**BIKE SERVICE AND SPARES MANAGEMENT**

**ABSTRACT**

The Bike Service and Spares Management System is a sophisticated web-based application tailored to streamline the intricate processes involved in managing bike servicing and spare parts inventory within a bike service center environment. Leveraging a combination of HTML, CSS, JavaScript, Bootstrap, PHP, and MySQL technologies, this system delivers a seamless user experience coupled with robust management functionalities.

At its core, the system boasts a secure user authentication mechanism, catering to both administrators and customers. Administrators are granted comprehensive access to all management functionalities, enabling them to oversee operations effectively. On the other hand, customers can effortlessly log in to access their service history, place new service requests, and monitor the progress of existing ones.

One of the standout features of the system is its intuitive service booking module. Customers can easily schedule service appointments by selecting preferred date and time slots, specifying service requirements, and providing relevant bike details. This streamlined process not only enhances customer convenience but also optimizes service center scheduling and resource allocation.

Furthermore, the system facilitates real-time service tracking, allowing customers to stay informed about the status of their service requests. They receive timely updates regarding service progress, estimated completion time, and any additional requirements, fostering transparency and customer satisfaction throughout the service journey.

In addition to service management, the system offers robust spare parts management capabilities. Administrators can efficiently oversee spare parts inventory, including adding new parts, updating existing ones, and monitoring stock levels. This ensures timely replenishment and optimal utilization of spare parts, minimizing service center downtime and maximizing operational efficiency.

Upon completion of service, the system generates detailed invoices for customers, encompassing service charges, spare parts used, and any additional fees. This streamlined invoicing process enhances financial transparency and facilitates smooth transactions between the service center and its customers.

To empower administrators with actionable insights, the system provides comprehensive reporting and analytics tools. Administrators can access detailed reports on service performance, revenue analysis, and trend identification, enabling informed decision-making and strategic planning.

With its responsive design built on the Bootstrap framework, the application ensures compatibility across various devices and screen sizes. This responsiveness enhances accessibility and usability, allowing users to interact with the system seamlessly across desktops, tablets, and smartphones.

Overall, the Bike Service and Spares Management System represents a comprehensive solution tailored to the unique needs of bike service centers. Its intuitive interface, robust functionality, and emphasis on customer satisfaction make it an indispensable tool for modern bike service businesses aiming to optimize operations, streamline processes, and drive business growth.

Top of Form

**INTRODUCTION**

In the dynamic world of bike servicing, the efficient management of service appointments, spare parts inventory, and customer communications is pivotal for the success of service centers. Recognizing these challenges, the Bike Service and Spares Management System emerges as a transformative solution designed to revolutionize the way bike service centers operate.

This web-based application harnesses the combined power of HTML, CSS, JavaScript, Bootstrap, PHP, and MySQL technologies to offer a comprehensive platform that addresses the diverse needs of bike service centers. With its user-friendly interface and robust functionalities, the system aims to streamline operations, enhance customer satisfaction, and drive business growth.

At its essence, the system prioritizes user experience and security. Through secure user authentication mechanisms, it ensures that only authorized personnel can access sensitive data, safeguarding the integrity of information within the system.

For customers, the system simplifies the service booking process, allowing them to schedule appointments conveniently and track the progress of their service requests in real-time. This transparency fosters trust and confidence in the service center, ultimately enhancing customer loyalty.

Meanwhile, administrators benefit from powerful management tools that facilitate efficient spare parts inventory management and comprehensive reporting. By optimizing inventory levels and analyzing service performance metrics, administrators can make data-driven decisions to improve operational efficiency and profitability.

Furthermore, the system's responsive design ensures seamless accessibility across various devices, empowering users to interact with the platform anytime, anywhere. Whether on desktops, tablets, or smartphones, users can experience the same level of functionality and performance.

In essence, the Bike Service and Spares Management System represents a paradigm shift in how bike service centers operate. By embracing technology and innovation, it equips service centers with the tools they need to thrive in today's competitive landscape, delivering exceptional service experiences and driving sustainable business success.

**OBJECTIVES**

1. **Efficient Service Management**: The primary objective of the Bike Service and Spares Management System is to streamline service management processes within bike service centers. This includes optimizing appointment scheduling, tracking service requests, and ensuring timely completion of services.
2. **Spare Parts Inventory Optimization**: Another key objective is to facilitate efficient management of spare parts inventory. The system aims to optimize stock levels, track usage, and ensure timely replenishment of spare parts to minimize downtime and maximize service center productivity.
3. **Enhanced Customer Experience**: The system is designed to enhance the overall customer experience by providing convenient service booking options, real-time service tracking, and transparent communication channels. By prioritizing customer satisfaction, the system aims to foster long-term customer loyalty.
4. **Data-driven Decision Making**: A crucial objective of the system is to empower administrators with comprehensive reporting and analytics tools. By analyzing service performance metrics, revenue trends, and customer feedback, administrators can make informed decisions to improve operational efficiency and strategic planning.
5. **Security and Data Integrity**: Ensuring the security and integrity of data is paramount. The system aims to implement robust security measures to safeguard sensitive information and prevent unauthorized access, thereby maintaining trust and confidentiality.
6. **Scalability and Adaptability**: As bike service centers evolve and grow, the system aims to remain scalable and adaptable to accommodate changing needs and technological advancements. This ensures that the system can continue to meet the requirements of service centers of varying sizes and complexities.
7. **Seamless Accessibility**: The system aims to provide seamless accessibility across different devices and platforms. By adopting a responsive design approach, users can interact with the system effortlessly, whether on desktop computers, tablets, or mobile devices.

By aligning with these objectives, the Bike Service and Spares Management System endeavors to revolutionize the way bike service centers operate, enabling them to deliver exceptional service experiences, drive operational efficiency, and achieve sustainable growth in a competitive market environment.

Top of Form

**EXISTING SYSTEM**

Before the implementation of the proposed Bike Service and Spares Management System, many bike service centers often rely on traditional manual methods or outdated software systems to manage their operations. While these existing systems may have served their purpose to some extent, they often come with several disadvantages that hinder efficiency, productivity, and customer satisfaction.

1. **Manual Processes**: Many bike service centers still rely heavily on manual processes for service appointment scheduling, spare parts inventory management, and customer communication. This manual approach is prone to errors, delays, and inefficiencies, leading to suboptimal service delivery.
2. **Limited Accessibility**: Traditional systems may lack accessibility features, making it challenging for administrators to manage operations remotely or for customers to interact with the service center outside of business hours. This limitation can hinder convenience and customer satisfaction.
3. **Poor Communication Channels**: Inadequate communication channels between the service center and customers can lead to misunderstandings, delays in service updates, and overall dissatisfaction. Without efficient communication tools, customers may feel disconnected from the service process.
4. **Inefficient Inventory Management**: Manual inventory management processes can result in inaccuracies in spare parts tracking, leading to stockouts or overstocking of parts. This inefficiency can impact service center productivity and profitability.
5. **Lack of Data Analysis**: Traditional systems may lack robust reporting and analytics capabilities, making it difficult for administrators to gain insights into service performance, revenue trends, and customer behavior. Without data-driven decision-making tools, service centers may struggle to optimize operations and identify areas for improvement.
6. **Security Concerns**: Older software systems may lack modern security features, leaving sensitive customer and business data vulnerable to breaches or unauthorized access. Inadequate security measures can undermine customer trust and expose the service center to regulatory compliance risks.
7. **Limited Scalability**: Legacy systems may lack scalability, making it challenging for service centers to adapt to growing business demands or incorporate new technologies. This limitation can impede business growth and innovation.

In summary, the existing systems used by many bike service centers suffer from various disadvantages, including manual processes, limited accessibility, poor communication channels, inefficient inventory management, lack of data analysis capabilities, security concerns, and limited scalability. These shortcomings underscore the need for a modern and comprehensive solution like the proposed Bike Service and Spares Management System to address these challenges and drive operational excellence in bike service center operations.

**PROPOSED SYSTEM**

The proposed Bike Service and Spares Management System aims to overcome the limitations of existing systems by leveraging modern technologies and innovative features. This comprehensive solution offers numerous advantages that enhance operational efficiency, customer satisfaction, and business growth for bike service centers.

1. **Streamlined Service Management**: The system provides an intuitive interface for managing service appointments, enabling administrators to efficiently schedule, track, and manage service requests. This streamlines service operations, reduces administrative overhead, and ensures timely completion of services.
2. **Efficient Spare Parts Inventory Management**: With robust inventory management functionalities, the system optimizes spare parts inventory levels, tracks usage, and facilitates timely replenishment. This minimizes stockouts, reduces inventory holding costs, and ensures that the service center has the necessary parts available to complete service requests promptly.
3. **Enhanced Customer Experience**: The system offers convenient service booking options and real-time service tracking capabilities for customers, fostering transparency and trust. Customers can easily schedule appointments, monitor service progress, and receive timely updates, resulting in a superior service experience and increased satisfaction.
4. **Improved Communication Channels**: The system incorporates communication features such as automated notifications and messaging functionalities, facilitating seamless communication between the service center and customers. This enhances engagement, reduces misunderstandings, and ensures that customers are informed throughout the service process.
5. **Advanced Reporting and Analytics**: By providing comprehensive reporting and analytics tools, the system empowers administrators to gain insights into service performance, revenue trends, and customer behavior. This enables data-driven decision-making, identifies areas for improvement, and drives continuous optimization of service center operations.
6. **Enhanced Security Measures**: The system implements modern security features, including encryption, access controls, and data integrity mechanisms, to protect sensitive customer and business data. This ensures compliance with regulatory requirements, mitigates security risks, and instills confidence in customers regarding the privacy and security of their information.
7. **Scalability and Flexibility**: Built on scalable technologies, the system is designed to accommodate the evolving needs of bike service centers, whether they are small independent shops or large chains. This scalability enables service centers to adapt to changing business requirements, incorporate new features, and expand their operations seamlessly.
8. **Responsive Design for Accessibility**: With a responsive design approach, the system ensures accessibility across various devices and platforms, allowing users to interact with the platform anytime, anywhere. This enhances convenience for administrators and customers, facilitating seamless engagement with the system.

In conclusion, the proposed Bike Service and Spares Management System offers a comprehensive and modern solution that addresses the challenges faced by bike service centers. By leveraging advanced technologies and innovative features, the system delivers numerous advantages, including streamlined service management, efficient inventory management, enhanced customer experience, improved communication channels, advanced reporting and analytics capabilities, enhanced security measures, scalability, and accessibility. These advantages empower service centers to optimize operations, drive customer satisfaction, and achieve sustainable business growth in today's competitive landscape.

Top of Form

**MODULE DESCRIPTION**

1. **User Authentication Module**: This module manages user authentication, allowing administrators and customers to securely log in to the system. It includes functionality for user registration, login, password management, and access control.
2. **Service Booking Module**: The service booking module enables customers to schedule service appointments conveniently. It provides features for selecting service types, choosing preferred dates and times, specifying service requirements, and submitting booking requests.
3. **Service Tracking Module**: This module allows customers to track the progress of their service requests in real-time. It provides status updates, estimated completion times, and notifications for any changes or additional requirements during the service process.
4. **Spare Parts Management Module**: The spare parts management module facilitates efficient management of spare parts inventory. It includes features for adding new parts, updating existing ones, tracking stock levels, and generating alerts for low inventory or stockouts.
5. **Invoice Generation Module**: The invoice generation module automates the process of generating invoices for completed services. It calculates service charges, includes details of spare parts used, and generates itemized invoices for customers to review and download.
6. **Reporting and Analytics Module**: This module provides administrators with comprehensive reporting and analytics tools to monitor service performance, track revenue, analyze trends, and make informed business decisions. It includes pre-defined reports as well as customizable reporting options.
7. **Security Module**: The security module ensures the confidentiality, integrity, and security of sensitive data within the system. It implements encryption, access controls, and other security measures to protect user information, prevent unauthorized access, and comply with data protection regulations.
8. **Responsive Design Module**: The responsive design module ensures that the system is accessible and functional across various devices and screen sizes. It adapts the user interface and layout to provide a consistent and optimal user experience on desktops, tablets, and smartphones.
9. **Admin Dashboard Module**: The admin dashboard module provides administrators with a centralized interface to manage all aspects of the system. It includes features for user management, service scheduling, inventory tracking, reporting, and configuration settings.

Each module plays a crucial role in the overall functionality and effectiveness of the Bike Service and Spares Management System, enabling service centers to optimize operations, enhance customer satisfaction, and drive business success.

**SYSTEM SPECIFICATION**

**HARDWARE SPECIFICATION**

|  |  |
| --- | --- |
| System | HP 15s |
| Processor | Ryzen 5 2.1 GHz |
| Storage | 512 GB SSD |
| RAM | 16 GB |
| Monitor | Integrated Monitor |
| Mouse | Integrated Trackpad |
| Keyboard | Integrated Keyboard |

**OPERATING SYSTEM**

|  |  |
| --- | --- |
| Operating System | Windows 11 |
| Front End | HTML, Bootstrap |
| Back End | PHP Version 8, MySQL Version 8 |
| Server | XAMPP |

**SOFTWARE SPECIFICATION**

**XAMPP:**

XAMPP is an [open-source](https://en.wikipedia.org/wiki/Free_software) [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) [web server](https://en.wikipedia.org/wiki/Web_server) [solution stack](https://en.wikipedia.org/wiki/Solution_stack) package developed by Apache Friends, consisting mainly of the [Apache HTTP Server](https://en.wikipedia.org/wiki/Apache_HTTP_Server), [Maria DB](https://en.wikipedia.org/wiki/MariaDB) [database](https://en.wikipedia.org/wiki/Database), and [interpreters](https://en.wikipedia.org/wiki/Interpreter_%28computing%29) for scripts written in the [PHP](https://en.wikipedia.org/wiki/PHP) and [Perl](https://en.wikipedia.org/wiki/Perl) [programming languages](https://en.wikipedia.org/wiki/Programming_language). XAMPP stands for Cross-Platform (X), Apache (A), Maria DB (M), PHP (P), and Perl (P). It is a simple, lightweight Apache distribution that makes it extremely easy for developers to create a local web server for testing and deployment purposes.

Everything needed to set up a web server – server application (Apache), database (Maria DB), and scripting language (PHP) – is included in an extractable file. XAMPP is also cross-platform, which means it works equally well on Linux, Mac, and Windows.

XAMPP's designers intended it for use only as a development tool, to allow website designers and programmers to test their work on their computers without any access to the Internet.

**CROSS-PLATFORM**

Cross-platform software is a type of software application that works on multiple operating systems or devices, which are often referred to as platforms. A platform means an operating system such as Windows, Mac OS, Android, or iOS. When a software application works on more than one platform, the user can utilize the software on a wider choice of devices and computers.

**BENEFITS OF CROSS-PLATFORM**

The benefit of a cross-platform software app or program is that you can use the same program whether you’re on a Windows PC or whether you’re logging in from your laptop or smartphone. The Microsoft Office suite of applications, which includes Word, Excel, and PowerPoint, is available on Windows, Mac OS, iOS (iPhone/iPad), and Android. While there are differences based on how the platforms work, you’ll have a similar experience within the application between all of your devices.

Having a similar experience across any platform means there’s a much smaller learning curve if one even exists at all, so you’ll be more productive and be able to use a software product you’re familiar with regardless of the operating system or device you choose. In addition, your files can be moved much more easily between your devices so you can use the software with whatever device you have with you at the time. And there’s a way to keep all of your work in sync across all of your devices, by using the cloud.

**EXAMPLES OF CROSS-PLATFORM**

**Unity 3D**

First, let’s talk about Unity3D. I think the game engine should be preferred by people who want to write mobile games.  
You can develop games on 17 platforms using multiple languages, including Linux. Of course, iOS, Android, and Windows Phone is also the most ideal game engine to develop games.

You can develop your application using C #, JS, and C ++.

Link to: [https://unity3d.com](https://unity3d.com/)

**Xamarin**

Xamarin Some time ago, it was purchased by Microsoft and is a perfect fit for developers using C #.

Because it is a C # language, it has a lot of documentation, and because of Microsoft support, Xamarin is the choice for C # developers.

In addition, you can do everything you can do in Objective-C, Swift, and Java with the Xamarin library.

Link to: [https://xamarin.com](https://xamarin.com/)

**React Native**

React Native is an open-source JavaScript library developed by the new generation of React — Facebook, which was open to Github in 2013. Native application creation means writing applications only for a specific operating system. React Native helps developers reuse their code over the web and on mobile. Developers will not have to create the same app from scratch for iOS and Android. They will be able to reuse the code in each operating system. The great thing about React Native is that there is little difference between a finished application in Objective-C or Java and an application built using React Native. Android and iOS code development environments are very different. So it takes time to remove the application to two different platforms. However, with React Native, only one developer can write on different mobile operating systems.

**APACHE:**

The Apache HTTP Server, colloquially called Apache is a [free and open-source](https://en.wikipedia.org/wiki/Free_and_open-source) [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) [web server](https://en.wikipedia.org/wiki/Web_server) software, released under the terms of [Apache License](https://en.wikipedia.org/wiki/Apache_License) 2.0. Apache is developed and maintained by an open community of developers under the auspices of the [Apache Software Foundation](https://en.wikipedia.org/wiki/Apache_Software_Foundation).

The vast majority of Apache HTTP Server instances run on a [Linux distribution](https://en.wikipedia.org/wiki/Linux_distribution), but current versions also run on [Microsoft Windows](https://en.wikipedia.org/wiki/Microsoft_Windows), [OpenVMS](https://en.wikipedia.org/wiki/OpenVMS),  and a wide variety of [Unix-like](https://en.wikipedia.org/wiki/Unix-like) systems. Past versions also ran on [NetWare](https://en.wikipedia.org/wiki/NetWare), [OS/2](https://en.wikipedia.org/wiki/OS/2), and other operating systems,  including ports to mainframes.

Originally based on the HTTP server, the development of Apache began in early 1995 after work on the NCSA code stalled. Apache played a key role in the initial growth of the [World Wide Web](https://en.wikipedia.org/wiki/World_Wide_Web), quickly overtaking NCSA HTTP as the dominant [HTTP](https://en.wikipedia.org/wiki/HTTP) server. In 2009, it became the first web server software to serve more than 100 million [websites](https://en.wikipedia.org/wiki/Website). As of January 2021, [Netcraft](https://en.wikipedia.org/wiki/Netcraft) estimated that Apache served 24.63% of the million busiest websites, while [Nginx](https://en.wikipedia.org/wiki/Nginx) served 23.21% and Microsoft is in third place at 6.85% (for some of Netcraft's other stats Nginx is ahead of Apache), while according to W3Techs, Apache is ranked first at 35.0% and Nginx second at 33.0% and Cloudflare Server third at 17.3%.

**LANGUAGE SPECIFICATION**

**PHP**

**INTRODUCTION OF PHP**

PHP started as a small open-source project that evolved as more and more people found out how useful it was. Rasmus Lerdorf unleashed the first version of PHP way back in 1994.

* PHP is a recursive acronym for "PHP: Hypertext Preprocessor".
* PHP is a server-side scripting language that is embedded in HTML. It is used to manage dynamic content, databases, session tracking, and even build entire e-commerce sites.
* It is integrated with several popular databases, including MySQL, PostgreSQL, Oracle, Sybase, Informix, and Microsoft SQL Server.
* PHP is pleasingly zippy in its execution, especially when compiled as an Apache module on the Unix side. The MySQL server, once started, executes even very complex queries with huge result sets in record-setting time.
* PHP supports a large number of major protocols such as POP3, IMAP, and LDAP. PHP4 added support for Java and distributed object architectures (COM and CORBA), making n-tier development a possibility for the first time.
* PHP is forgiving: PHP language tries to be as forgiving as possible.
* PHP Syntax is the same as C language.

**What is a PHP File?**

* PHP files can contain text, HTML, CSS, JavaScript, and PHP code.
* PHP code is executed on the server, and the result is returned to the browser as plain HTML.
* PHP files have the extension ".php".

**What Can PHP Do?**

* PHP can generate dynamic page content and it can create, open, read, write, delete, and close files on the server and it can collect form data.
* PHP can send and receive cookies it can add, delete, and modify data in your database and it can be used to control user-access and encrypt data.

**Why PHP?**

* PHP runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.).
* PHP is compatible with almost all servers used today (Apache, IIS, etc.).
* PHP supports a wide range of databases.
* PHP is free.
* PHP is easy to learn and runs efficiently on the server side.

**What is a Database?**

* A database is a separate application that stores a collection of data. Each database has one or more distinct APIs for creating, accessing, managing, searching, and replicating the data it holds.
* Other kinds of data stores can be used, such as files on the file system or large hash tables in memory but data fetching and writing would not be so fast and easy with those types of systems.
* Nowadays, we use relational database management systems (RDBMS) to store and manage huge volumes of data. This is called a relational database because all the data is stored in different tables and relations are established using primary keys or other keys known as foreign keys.

**MySQL Database**

* MySQL is released under an open-source license. So you have nothing to pay to use it. MySQL is a very powerful program in its own right. It handles a large subset of the functionality of the most expensive and powerful database packages.
* MySQL uses a standard form of the well-known SQL data language. MySQL works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc. MySQL works very quickly and works well even with large data sets.
* MySQL is very friendly to PHP, the most appreciated language for web development. MySQL supports large databases, up to 50 million rows or more in a table.
* The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB). MySQL is customizable.
* The open-source GPL license allows programmers to modify the MySQL software to fit their specific environments.

**TABLE CREATION**

* Name of the table
* Names of fields
* Definitions for each field
* Field Attribute **NOT NULL** is being used because we do not want this field to be NULL. So if the user tries to create a record with a NULL value, then MySQL will raise an error.
* Field Attribute **AUTO\_INCREMENT** tells MySQL to go ahead and add the next available number to the id field.
* Keyword **PRIMARY KEY** is used to define a column as the primary key. You can use multiple columns separated by a comma to define a primary key.

**ADMINISTRATIVE MYSQL COMMAND**

* **USE DATABASE NAME**: This will be used to select a particular database in the MySQL work area.
* **SHOW DATABASES:** Lists the databases that are accessible by the MySQL DBMS.
* **SHOW TABLES:** Shows the tables in the database once a database has been selected with the use command.
* **SHOW COLUMNS FROM Table name:** Shows the attributes, types of attributes, key information, whether NULL is permitted, defaults, and other information for a table.
* **SHOW INDEX FROM Table name:** Presents the details of all indexes on the table, including the PRIMARY KEY

**CREATING TABLES USING PHP SCRIPT:**

To create a new table in any existing database you would need to use PHP function **mysqli\_query()**.

**Dropping Tables Using PHP Script:**

Drop an existing table in any database, you would need to use the PHP function **mysqli\_query()**.

**INSERTING DATA USING PHP SCRIPT:**

**CREATE**

Create table statement is used to create a table in MySQL.

**SELECT**

The SELECT statement is used to select data from one or more tables.

**UPDATE**

The UPDATE statement is used to update existing records in a table:

**DELETE**

The DELETE statement is used to delete records from a table:

**DATABASE DESIGN:**

The data in the system has to be stored and retrieved from the database. Designing the database is part of system design. Data elements and data structures to be stored have been identified at the analysis stage.

They are structured and put together to design the data storage and retrieval system. A database is a collection of interrelated data stored with minimum redundancy to serve many users quickly and efficiently.

The general objective is to make database access easy, quick, inexpensive, and flexible for the user. Relationships are established between the data items and unnecessary data items are removed.

Normalization is done to get an internal consistency of data and to have minimum redundancy and maximum stability. This ensures minimizing data storage required, minimizing chances of data inconsistencies, and optimizing for updates.

**INPUT DESIGN**

The Input design is the main feature of the system. Input design determines the format and validation criteria for data entering the system. Inputs originate with end-users; human factors play a significant role in input design. The input design is designed to control the input, avoid delay, and errors in data, avoid extra steps, to keep the process simple. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps, and keeping the process simple. The input is designed in such a way that it provides security and ease of use while retaining privacy.

The following are the general principles, that are considered in designing inputs,

* + - Enter only variable data
    - Do not input data that can be calculated
    - List of values
    - Sequence entry

**OUTPUT DESIGN**

Designing the output is more important than working up with a few layout charts and reports. The outputs are designed based on the issue encountered. It will also take care of who will receive the output, what for it is produced how many details are needed, when it is needed, and by what method.

The outputs designed in this system are easy to use and useful for their jobs. The outputs are simple to read and interpret. The outputs obtained from this system are designed by using a few guidelines, which are given below. The information should be clear and accurate, yet concise and restricted to relevant data. Reports should have titles, data, and descriptive headings for columns of data, numbered pages, and so on.

**SYSTEM TESTING**

System testing is the process of exercising software with the intent of finding and ultimately correcting errors. This fundamental philosophy does not change for web applications, because Web-based systems and applications reside on a network and interoperate with many different operating systems, browsers, hardware platforms, and communication protocols; the search for errors represents a significant challenge for web applications.

The distributed nature of client/server environments, the performance issues associated with transaction processing, the potential presence of several different hardware platforms, the complexities of network communication, the need to serve multiple clients from a centralized database, and the requirements imposed on the server all combine to make testing of client\server architectures.

Testing issues

* Client GUI considerations
* Target environment and platform diversity considerations
* Distributed database considerations
* Distributed processing considerations

**TYPES OF TESTING**

1. Unit Testing

2. Integration Testing

3. Validation Testing

4. User Acceptance Testing

5. System Testing

**Unit Testing**

All modules were tested and individually as soon as they were completed were checked for their correct functionality. Unit testing is carried out by verifying and recovering errors within the boundary of the smallest unit or a module. In this testing step, each module was found to be working satisfactorily per the expected output of the module. In the package development, each module is tested separately after it has been completed and checked with valid data.

**Integration Testing**

The entire project was split into small programs; each of these single programs gives a frame as an output. These programs were tested individually; at last, all these programs were combined by creating another program where all these constructions were used. It causes a lot of problems by not functioning in an integrated manner.

The user interface testing is important since the user has to declare that the arrangements made in the frames are convenient and it is satisfied. When the frames are tested, the end user gives suggestions. Since they were much exposed to do the work manually.

**Validation Testing**

At the culmination of the black box testing software is completely assembled as a package. Interfacing errors have been uncovered and corrected and a final series of tests i.e., validation succeeds when the software functions in a manner that can be reasonably accepted by the customer.

**User Acceptance Testing**

User acceptance testing of the system is the key factor in the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective systems at the time of development and making changes whenever required. This is done concerning the input screen design and output screen design.

**System Testing**

This is to verify that all the system elements have been properly integrated and perform allocated functions. Testing is executing a program to test the logic changes made in it to find errors. Tests are also conducted to find discrepancies between the system and its original objective, current specifications, and documents.

**SYSTEM IMPLEMENTATION**

Implementation is the stage in the project where the theoretical design is turned into a working system. The most crucial stage is achieving a successful new system & and giving the user confidence that the new system will work efficiently & and effectively in the implementation stage.

The stage consists of

* + - Testing the developed program with simple data.
    - Detections and correction of errors.
    - Creating whether the system meets user requirements.
    - Testing whether the system.
    - Making necessary changes as desired by the user.
    - Training user personnel.

**Implementation Procedures**

The implementation phase is less creative than the system design. A system project may be dropped at any time before implementation, although it becomes more difficult when it goes to the design phase.

The final report to the implementation phase includes procedural flowcharts, record layouts, report layouts, and a workable plan for implementing the candidate system design into an operational one. Conversion is one aspect of implementation.

**System Maintenance**

Maintenance is the implementation of the review plan. As important as it is, many programmers and analysts are to perform or identify themselves with the maintenance effort. There are psychological, personality, and professional reasons for this. Analysts and programmers spend far more time maintaining programs than they do writing them. Maintenance accounts for 50-80 percent of total system development.

Maintenance is expensive. One way to reduce maintenance costs is through maintenance management and software modification audits.

* Maintenance is not as rewarding or exciting as developing systems. It is perceived as requiring neither skill nor experience.
* Users are not fully cognizant of the maintenance problem or its high cost.
* Few tools and techniques are available for maintenance.
* A good test plan is lacking.
* Standards, procedures, and guidelines are poorly defined and enforced.
* Programs are often maintained without care for structure and documentation.
* There are minimal standards for maintenance.
* Programmers expect that they will not be in their current commitment by the time their programs go into the maintenance cycle.

**SYSTEM DESIGN**

System design is "the process of studying a procedure or business to identify its goals, purposes and create systems and procedures that will efficiently achieve them". Another view sees system analysis as a problem-solving technique that breaks down a system into its component pieces for the study of how well those parts work and interact to accomplish their purpose.

The field of system analysis relates closely to requirements analysis or operations research. It is also "an explicit formal inquiry carried out to help a decision maker identify a better course of action and make a better decision than they might otherwise have made."

* **DESIGN NOTATION**

Design notations are used when planning and should be able to communicate the purpose of a program without the need for formal code. Commonly used design notations are:

* DFD
* ERD
* **DFD (DATA FLOW DIAGRAM):**

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an [information system](https://en.wikipedia.org/wiki/Information_system), modeling its process aspects. A DFD is often used as a preliminary step to create an overview of the system without going into great detail, which can later be elaborated. DFDs can also be used for the [visualization](https://en.wikipedia.org/wiki/Data_visualization) of [data processing](https://en.wikipedia.org/wiki/Data_processing) (structured design). A DFD shows what kind of information will be input to and output from the system, how the data will advance through the system, and where the data will be stored. It does not show information about the timing of the process or information about whether processes will operate in sequence or parallel, unlike a [flowchart](https://en.wikipedia.org/wiki/Flowchart) which also shows this information.

Data flow diagrams were popularized in the late 1970s, arising from the book Structured Design, by computing pioneers Ed Yourdon and Larry Constantine. They based it on the “data flow graph” computation models by David Martin and Gerald Estrin. The structured design concept took off in the software engineering field, and the DFD method took off with it. It became more popular in business circles, as it was applied to business analysis than in academic circles.

**DFD SYMBOLS**

The process that transforms data flow

Source or Destination of Data

Data Flow

Data source

**ENTITY RELATIONSHIP DIAGRAM**

The relation upon the system is structured through a conceptual ER-Diagram, which not only specifies the existential entities but also the standard relations through which the system exists and the cardinalities that are necessary for the system state to continue. The Entity Relationship Diagram (ERD) depicts the relationship between the data objects. The ERD is the notation that is used to conduct the data modeling activity The attributes of each data object noted in the ERD can be described resign a data object description.

The set of primary components that are identified by the ERD are

* + Data object
  + Relationships
  + Attributes
  + Various types of indicators

The primary purpose of the ERD is to represent data objects and their relationships.

**ER-DIAGRAM SYMBOL**

Entity

Relationship

Flow

**INPUT DESIGN**

The input design is the link between the information system and the user. It comprises the developing specifications and procedures for data preparation and those steps are necessary to put transaction data into a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps, and keeping the process simple. The input is designed in such a way that it provides security and ease of use while retaining privacy.

Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system. It is achieved by creating user-friendly screens for the data entry to handle large volumes of data.

The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulations can be performed. It also provides record viewing facilities. When the data is entered it will check for its validity. Data can be entered with the help of screens.

**DATABASE DESIGN**

The database is designed to manage large bodies of information. The management of data involves both the definitions of structures for the storage of information. In addition, the database system must provide for the safety of the information solved, despite system crashes or attempts at unauthorized access. For developing an efficient database users have to fulfill certain conditions such as controlled redundancy.

* Defining the data
* Inputting the data
* Locating the data
* Accessing the data
* Communicating the data

Revising the data

**Objectives of Database Design**

For designing a database design several objectives have to be met as follows:

* Ease of use
* Control of data integrity
* Control of redundancy
* Control of security
* Data independence (logical & physical)
* Data storage protection
* System performance

**OUTPUT DESIGN**

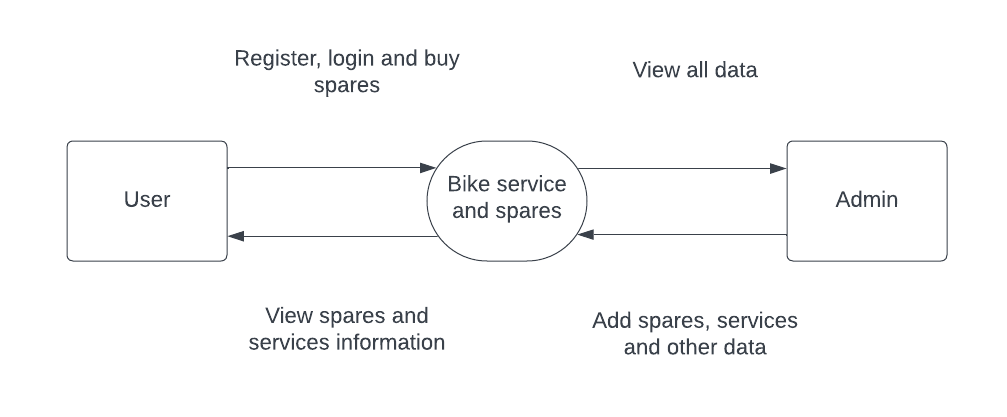
A quality output is one, which meets the requirements of the end user and presents the information. In any system results of processing are communicated to the users and other systems through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source of information to the user. Efficient and intelligent output design improves the system’s relationship to help user decision-making.

Output design generally refers to the results and information that are generated by the system for many end-users; output is the main reason for developing the system and the basis on which they evaluate the usefulness of the application.

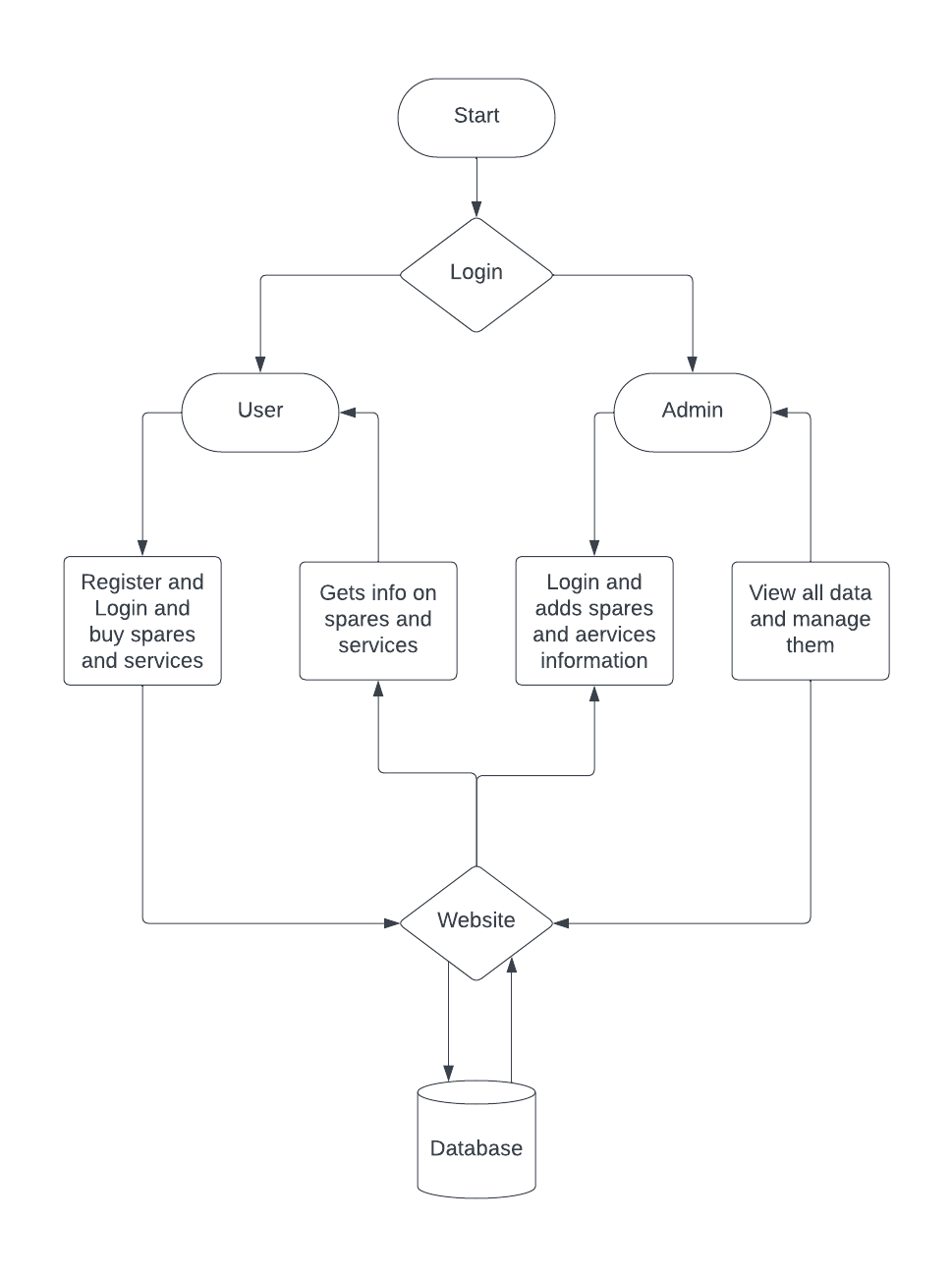
In this Online Repository System project output is to view customer details, employee lists, order tracking details, and attendance percentage results.

**SYSTEM FLOW DIAGRAM**

**DATA FLOW DIAGRAM**

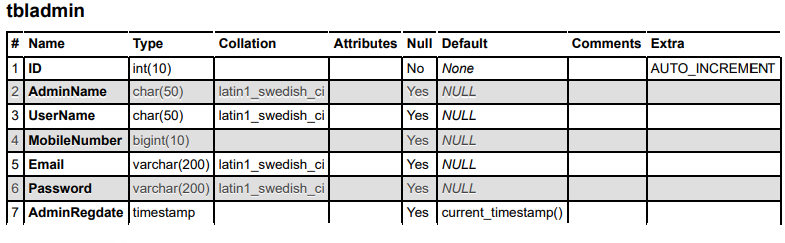


**ER DIAGRAM**

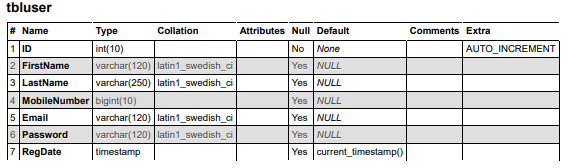


**DATABASE DESIGN**

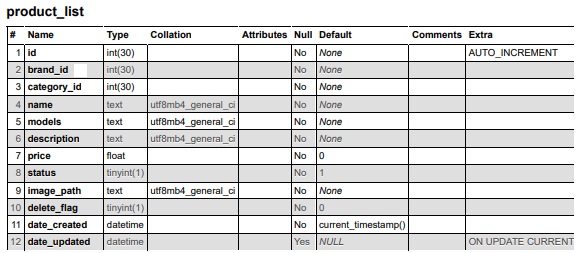
**Table name: tbladmin**

****

**Table name: tbluser**

****

**Table name: productlist**

****

**CONCLUSION**  
In conclusion, the proposed Bike Service and Spares Management System presents a comprehensive solution to address the challenges faced by bike service centers in managing their operations efficiently. By leveraging modern technologies and innovative features, the system offers numerous advantages, including streamlined service management, efficient inventory management, enhanced customer experience, improved communication channels, advanced reporting and analytics capabilities, enhanced security measures, scalability, and accessibility.

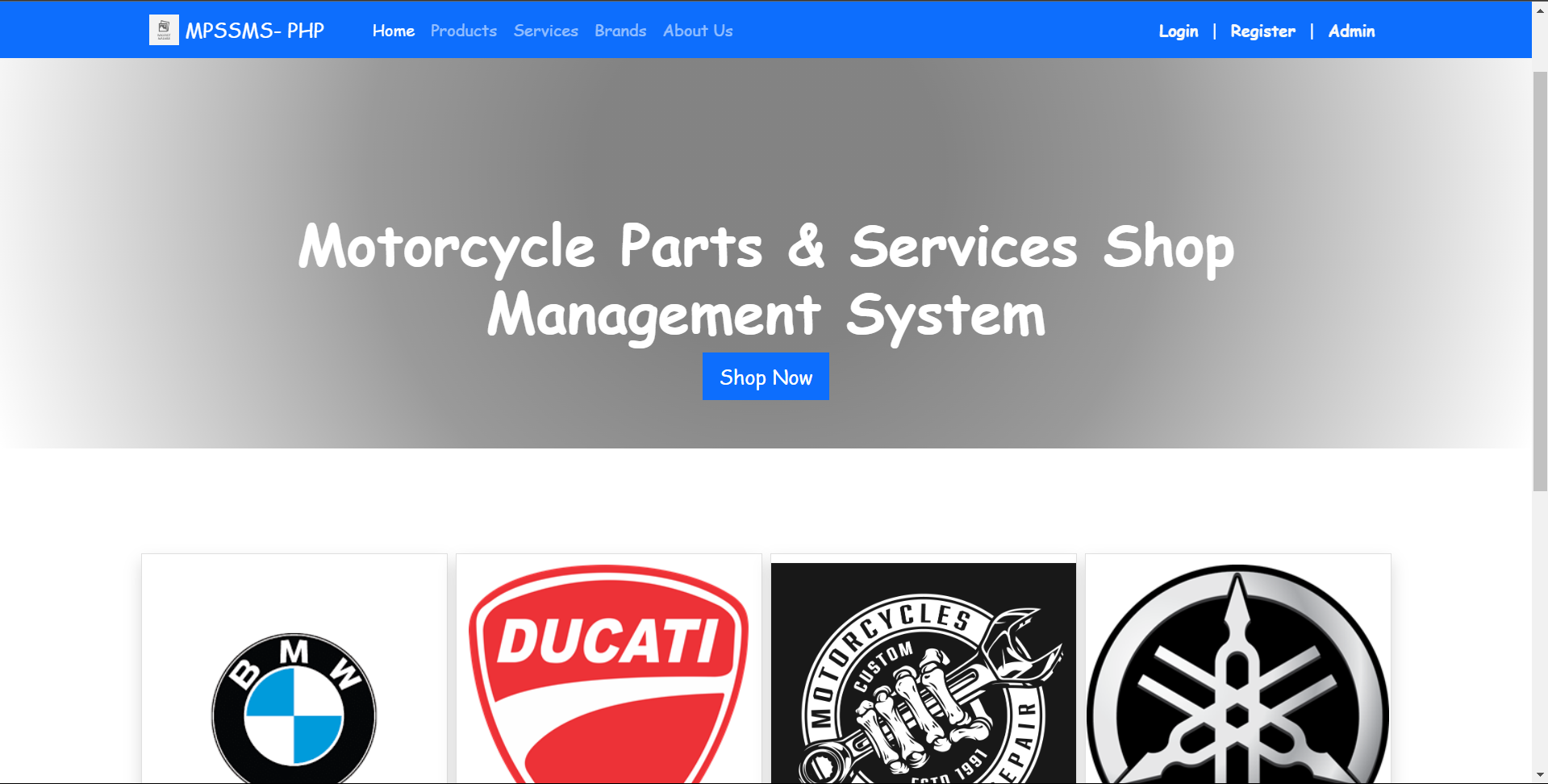
The modular architecture of the system ensures that each component functions cohesively to deliver a seamless and user-friendly experience for both administrators and customers. From user authentication and service booking to spare parts management and invoice generation, every aspect of the service center's operations is meticulously managed and optimized.

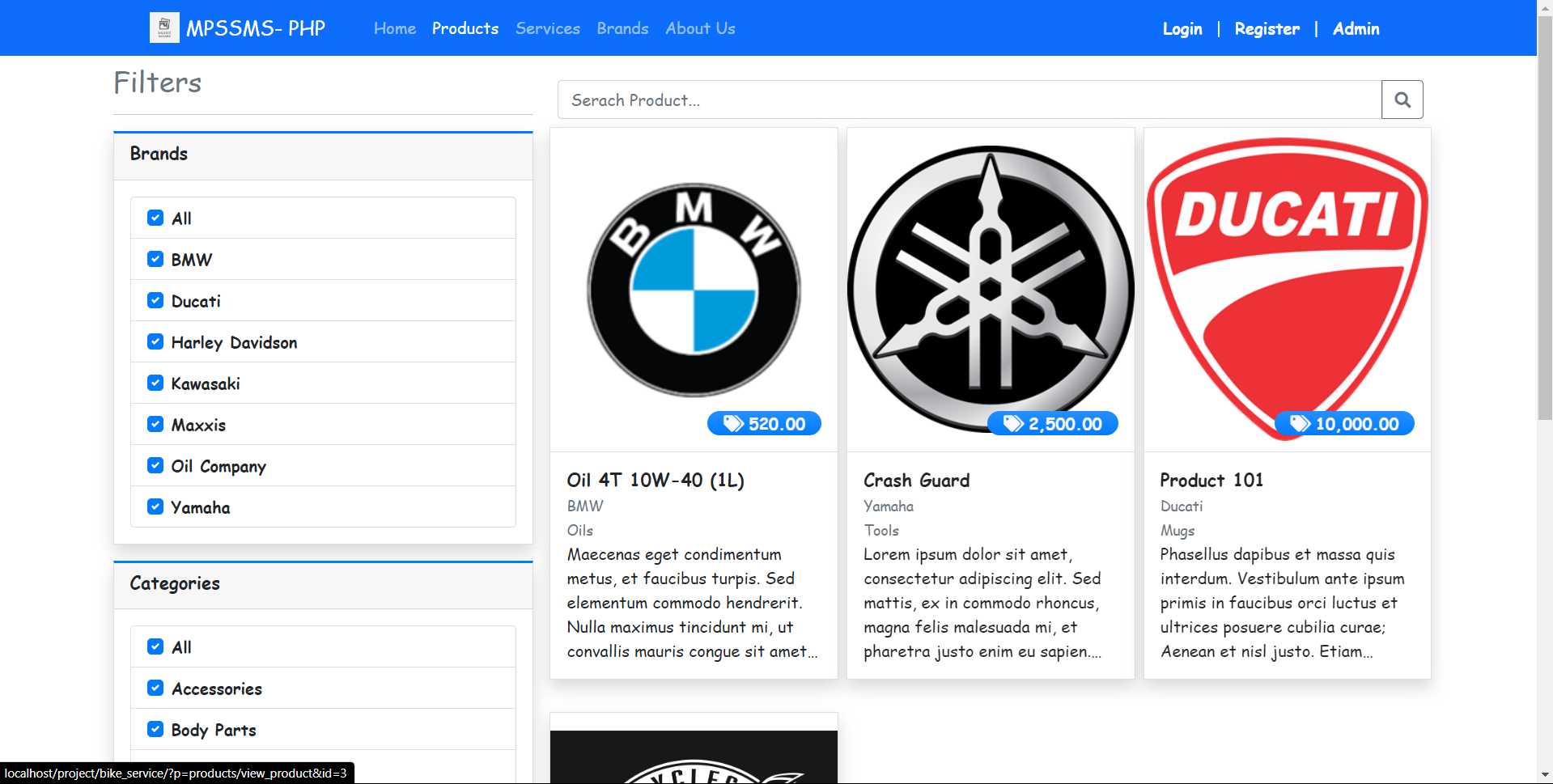
Moreover, the system's emphasis on security ensures the protection of sensitive customer and business data, fostering trust and confidence among users. With responsive design principles, the system ensures accessibility across various devices and platforms, enabling users to interact with the platform anytime, anywhere.

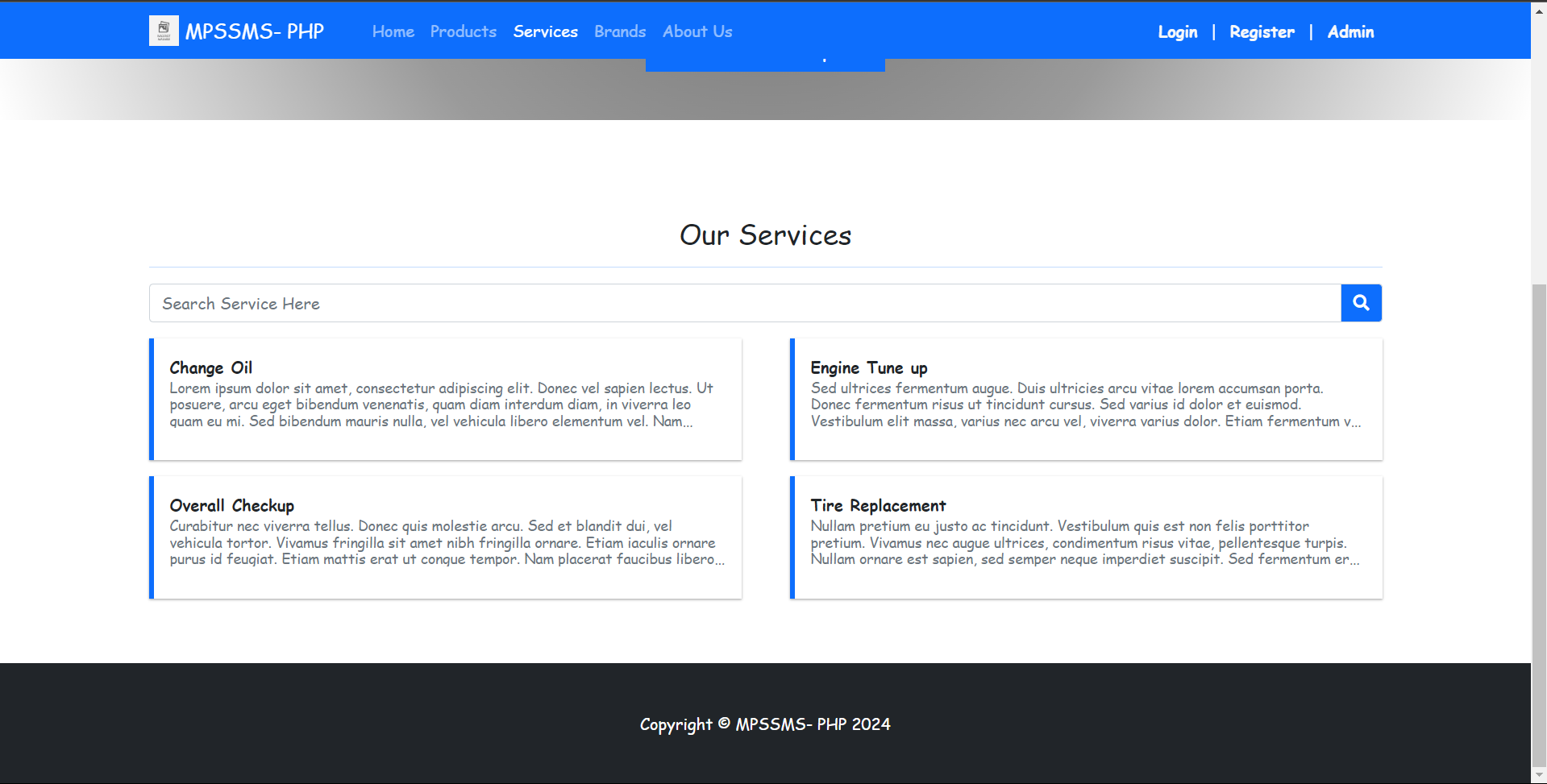
Overall, the Bike Service and Spares Management System represents a significant advancement in the realm of bike service center operations. By empowering service centers with the tools they need to streamline operations, enhance customer satisfaction, and drive business growth, the system paves the way for sustainable success in today's competitive market environment. Its implementation promises to revolutionize the way bike service centers operate, delivering exceptional service experiences and cementing their position as industry leaders.

Top of Form

**SCREENSHOTS**

****

****

****

**SAMPLE CODE**

**<style>**

**.prod-cart-img{**

**width:10em;**

**height:10em;**

**object-fit:scale-down;**

**object-position: center center;**

**}**

**</style>**

**<div class="content py-5 mt-3">**

**<div class="container">**

**<h3><b>My Shopping Cart</b></h3>**

**<hr>**

**<div class="card card-outline card-primary shadow rounded-0">**

**<div class="w-100" id="cart-list">**

**<?php**

**$total = 0;**

**$cart = $conn->query("SELECT c.\*,p.name, p.price, p.image\_path,b.name as brand, cc.category FROM `cart\_list` c inner join product\_list p on c.product\_id = p.id inner join brand\_list b on p.brand\_id = b.id inner join categories cc on p.category\_id = cc.id where c.client\_id = '{$\_settings->userdata('id')}' order by p.name asc");**

**while($row = $cart->fetch\_assoc()):**

**$total += ($row['quantity'] \* $row['price']);**

**?>**

**<div class="d-flex align-items-center w-100 border cart-item" data-id="<?= $row['id'] ?>">**

**<div class="col-auto flex-grow-1 flex-shrink-1 px-1 py-1">**

**<div class="d-flex align-items-center w-100 ">**

**<div class="col-auto">**

**<img src="<?= validate\_image($row['image\_path']) ?>" alt="Product Image" class="img-thumbnail prod-cart-img">**

**</div>**

**<div class="col-auto flex-grow-1 flex-shrink-1">**

**<a href="./?p=products/view\_product&id=<?= $row['product\_id'] ?>" class="h4 text-muted">**

**<p class="text-truncate-1 m-0"><?= $row['name'] ?></p>**

**</a>**

**<small><?= $row['brand'] ?></small><br>**

**<small><?= $row['category'] ?></small><br>**

**<div class="d-flex align-items-center w-100 mb-1">**

**<div class="input-group " style="width:8em">**

**<div class="input-group-prepend">**

**<button class="btn btn-sm btn-outline-secondary btn-minus" data-id='<?= $row['id'] ?>'><i class="fa fa-minus"></i></button>**

**</div>**

**<input type="text" value="<?= $row['quantity'] ?>" readonly class="form-control form-control-sm text-center">**

**<div class="input-group-append">**

**<button class="btn btn-sm btn-outline-secondary btn-plus" data-id='<?= $row['id'] ?>'><i class="fa fa-plus"></i></button>**

**</div>**

**</div>**

**<span class="ml-2">X <?= number\_format($row['price'],2) ?></span>**

**</div>**

**<button class="btn btn-sm btn-flat btn-outline-danger btn-remove" data-id="<?= $row['id'] ?>"><i class="fa fa-times"></i> Remove</button>**

**</div>**

**</div>**

**</div>**

**<div class="col-auto text-right">**

**<h3><b><?= number\_format($row['quantity'] \* $row['price'],2) ?></b></h3>**

**</div>**

**</div>**

**<?php endwhile; ?>**

**<?php if($cart->num\_rows <= 0): ?>**

**<div class="d-flex align-items-center w-100 border justify-content-center">**

**<div class="col-12 flex-grow-1 flex-shrink-1 px-1 py-1">**

**<small class="text-muted">No Data</small>**

**</div>**

**</div>**

**<?php endif; ?>**

**<div class="d-flex align-items-center w-100 border">**

**<div class="col-auto flex-grow-1 flex-shrink-1 px-1 py-1">**

**<h3 class="text-center">TOTAL</h3>**

**</div>**

**<div class="col-auto text-right">**

**<h3><b><?= number\_format($total,2) ?></b></h3>**

**</div>**

**</div>**

**</div>**

**</div>**

**<div class="clear-fix my-2"></div>**

**<div class="text-right">**

**<button class="btn btn-flat btn-sm btn-dark" type="button" id="checkout">Checkout</button>**

**</div>**

**</div>**

**</div>**

**<script>**

**window.update\_quantity = function($cart\_id = 0, $quantity = ""){**

**start\_loader();**

**$.ajax({**

**url:\_base\_url\_+'classes/master.php?f=update\_cart\_quantity',**

**data:{cart\_id : $cart\_id, quantity : $quantity},**

**method:'POST',**

**dataType:'json',**

**error:err=>{**

**console.error(err)**

**alert\_toast('An error occurred.','error')**

**end\_loader()**

**},**

**success:function(resp){**

**if(resp.status == 'success'){**

**location.reload()**

**}else if(!!resp.msg){**

**alert\_toast(resp.msg,'error')**

**}else{**

**alert\_toast('An error occurred.','error')**

**}**

**end\_loader();**

**}**

**})**

**}**

**$(function(){**

**$('.btn-minus').click(function(){**

**update\_quantity($(this).attr('data-id'),"- 1")**

**})**

**$('.btn-plus').click(function(){**

**update\_quantity($(this).attr('data-id'),"+ 1")**

**})**

**$('.btn-remove').click(function(){**

**\_conf("Are you sure to remove this product from cart list?","remove\_from\_cart",[$(this).attr('data-id')])**

**})**

**$('#checkout').click(function(){**

**if($('#cart-list .cart-item').length > 0){**

**location.href="./?p=place\_order"**

**}else{**

**alert\_toast('Shopping cart is empty.','error')**

**}**

**})**

**})**

**function remove\_from\_cart($id){**

**start\_loader();**

**$.ajax({**

**url:\_base\_url\_+'classes/master.php?f=remove\_from\_cart',**

**data:{cart\_id : $id},**

**method:'POST',**

**dataType:'json',**

**error:err=>{**

**console.error(err)**

**alert\_toast('An error occurred.','error')**

**end\_loader()**

**},**

**success:function(resp){**

**if(resp.status == 'success'){**

**location.reload()**

**}else if(!!resp.msg){**

**alert\_toast(resp.msg,'error')**

**}else{**

**alert\_toast('An error occurred.','error')**

**}**

**end\_loader();**

**}**

**})**

**}**

**</script>**

**FUTURE SCOPE**

1. **Integration of IoT Devices**: The future of bike service management could involve the integration of IoT devices with bikes and service equipment. IoT sensors installed in bikes could collect real-time data on performance metrics, wear and tear, and maintenance needs, enabling predictive maintenance and remote diagnostics. Service centers can utilize this data to proactively schedule service appointments and optimize maintenance schedules, thereby reducing downtime and improving customer satisfaction.
2. **Augmented Reality (AR) for Service Assistance**: Implementing AR technology can revolutionize the way bike service technicians perform repairs and maintenance tasks. Technicians equipped with AR-enabled devices can overlay digital instructions, diagrams, and step-by-step guides onto the physical bike, providing visual aids and guidance for complex repair procedures. This enhances efficiency, accuracy, and training for service personnel.
3. **Mobile Service Units**: To enhance convenience and accessibility, bike service centers could deploy mobile service units equipped with tools, equipment, and trained technicians to perform on-site repairs and maintenance. Customers can schedule appointments through the service center's website or app, and the mobile unit can visit their location to address service needs, eliminating the need for customers to transport their bikes to the service center.
4. **Predictive Analytics for Spare Parts Management**: Leveraging predictive analytics algorithms, service centers can forecast demand for spare parts based on historical data, seasonal trends, and bike usage patterns. This enables service centers to maintain optimal inventory levels, minimize stockouts, and reduce holding costs associated with excess inventory. Integrating predictive analytics into the existing service management system can optimize spare parts procurement and inventory management processes.
5. **Blockchain for Supply Chain Management**: Implementing blockchain technology can enhance transparency, traceability, and security in the bike service and spares supply chain. Blockchain-based solutions can create immutable records of transactions, track the movement of spare parts from manufacturers to service centers, and verify the authenticity of spare parts. This increases trust among stakeholders and reduces the risk of counterfeit parts entering the supply chain.
6. **Enhanced Customer Engagement Platforms**: Investing in customer engagement platforms such as mobile apps or web portals can improve communication and interaction between service centers and customers. These platforms can offer features such as appointment scheduling, service status updates, service history tracking, and loyalty programs. By providing a seamless and personalized experience, service centers can enhance customer satisfaction and loyalty.
7. **Integration with Vehicle Diagnostics Systems**: Future bike service management systems can integrate with onboard vehicle diagnostics systems to retrieve real-time data on bike performance, engine diagnostics, and error codes. Service centers can access this data remotely, diagnose issues more accurately, and provide targeted recommendations for repairs and maintenance. Integrating vehicle diagnostics systems with the service management platform improves efficiency and reduces turnaround time for service appointments.
8. **Expanded E-commerce Capabilities for Spare Parts**: To cater to the growing demand for online shopping, bike service centers can expand their e-commerce capabilities by offering an online store for spare parts and accessories. Customers can browse, purchase, and schedule installation services for spare parts directly through the service center's website or app. This expands revenue opportunities for service centers and provides added convenience for customers.

**REFFERENCES**

BIBLIOGRAPHY

Comprehensive Web Development Textbook References

General Web Development:

• MDN Web Docs: (https://developer.mozilla.org/) - The authoritative source from Mozilla, offering in-depth documentation, tutorials, and references for various web technologies.

• W3Schools: (https://www.w3schools.com/) - A well-established website with interactive tutorials, references, and examples for a wide range of web development topics.

• The Odin Project: (https://theodinproject.com/) - A free, full-stack web development curriculum with a strong focus on practical projects.

• FreeCodeCamp: (https://www.freecodecamp.org/) - Another free coding platform with interactive lessons, projects, and a supportive community for web development learning.

Specific Technologies:

• HTML:

o HTML Dog: (https://www.htmldog.com/) - Offers interactive tutorials, challenges, and references for learning HTML.

• CSS:

o CSS-Tricks: (https://css-tricks.com/) - A popular website with articles, tips, tricks, and best practices for mastering CSS.

• JavaScript:

o JavaScript30: (https://javascript30.com/) - Provides 30 JavaScript coding challenges for practicing and improving your JavaScript skills.

o Eloquent JavaScript Website: (https://eloquentjavascript.net/) - Interactive tutorials and resources aligned with the book "Eloquent JavaScript" by Marijn Haverbeke.

• Bootstrap:

o Get Bootstrap: (https://getbootstrap.com/) - The official Bootstrap website with comprehensive documentation, examples, and tutorials for learning and using the framework.

o Start Bootstrap: (https://startbootstrap.com/) - Offers free Bootstrap templates to use as a starting point for your projects.

• PHP:

o PHP.net: (https://www.php.net/manual/en/index.php) - The official PHP website with comprehensive documentation, tutorials, and a reference manual.

o Laracasts: (https://laracasts.com/) - Features video tutorials and screencasts for learning PHP, Laravel (a popular PHP framework), and other web development topics.

• MySQL:

o MySQL Documentation: (https://dev.mysql.com/doc/) - Official documentation from MySQL, providing detailed information on using the database management system.

o SQLBolt: (https://sqlbolt.com/) - An interactive platform where you can practice writing and running SQL queries, the language used with MySQL.

Community Resources:

• Stack Overflow: (https://stackoverflow.com/) - A question-and-answer website for programmers, where you can search for solutions to your coding problems or ask questions related to web development.

• GitHub: (https://github.com/) - A version control system for code hosting and collaboration. GitHub also offers a wealth of open-source web development projects that you can explore and learn from.

Comprehensive Web Development Online References

General Web Development:

• MDN Web Docs: (https://developer.mozilla.org/) - The authoritative source from Mozilla, offering in-depth documentation, tutorials, and references for various web technologies.

• W3Schools: (https://www.w3schools.com/) - A well-established website with interactive tutorials, references, and examples for a wide range of web development topics.

• The Odin Project: (https://theodinproject.com/) - A free, full-stack web development curriculum with a strong focus on practical projects.

• FreeCodeCamp: (https://www.freecodecamp.org/) - Another free coding platform with interactive lessons, projects, and a supportive community for web development learning.

Specific Technologies:

• HTML:

o HTML Dog: (https://www.htmldog.com/) - Offers interactive tutorials, challenges, and references for learning HTML.

• CSS:

o CSS-Tricks: (https://css-tricks.com/) - A popular website with articles, tips, tricks, and best practices for mastering CSS.

• JavaScript:

o JavaScript30: (https://javascript30.com/) - Provides 30 JavaScript coding challenges for practicing and improving your JavaScript skills.

o Eloquent JavaScript Website: (https://eloquentjavascript.net/) - Interactive tutorials and resources aligned with the book "Eloquent JavaScript" by Marijn Haverbeke.

• Bootstrap:

o Get Bootstrap: (https://getbootstrap.com/) - The official Bootstrap website with comprehensive documentation, examples, and tutorials for learning and using the framework.

o Start Bootstrap: (https://startbootstrap.com/) - Offers free Bootstrap templates to use as a starting point for your projects.

• PHP:

o PHP.net: (https://www.php.net/manual/en/index.php) - The official PHP website with comprehensive documentation, tutorials, and a reference manual.

o Laracasts: (https://laracasts.com/) - Features video tutorials and screencasts for learning PHP, Laravel (a popular PHP framework), and other web development topics.

• MySQL:

o MySQL Documentation: (https://dev.mysql.com/doc/) - Official documentation from MySQL, providing detailed information on using the database management system.

o SQLBolt: (https://sqlbolt.com/) - An interactive platform where you can practice writing and running SQL queries, the language used with MySQL.

Community Resources:

• Stack Overflow: (https://stackoverflow.com/) - A question-and-answer website for programmers, where you can search for solutions to your coding problems or ask questions related to web development.

• GitHub: (https://github.com/) - A version control system for code hosting and collaboration. GitHub also offers a wealth of open-source web development projects that you can explore and learn from.