**1. ABSTRACT**

The Theme Park Ticketing System is a simple PHP/MySQL project that helps manages and generates the visitors' tickets in a theme or amusement park. The system can generate a various ticket at once in different types and all of the tickets has a unique ticket number. The theme park management or administrator can set up different pricing for the rides, entrance, or promos that the park offers to the visitors. The system separates the adult and child fee for every pricing setup. The administrator user has the permission to manage all the data in the system especially the rides list and pricing list of the theme or amusement park. The system also generates a sales report for the number of tickets sold in the management's choice of the date range. This ticketing system also a simple calculator that helps the staff or admin calculate the total amount to pay and change of the customer or visitor. The ticket format is just simple, it only displays the ticket number, pay type, ticket type, and the date sold. This system provides a user-friendly and streamlined experience for both customers and park staff, helping to improve efficiency and revenue growth for the park.

**1.1 MAIN OBJECTIVE:**

The main objective is to develop a secure online ticket booking system to improve and enable booking of tickets easier to avoid long queues.

**1.2 SPECIFIC OBJECTIVES:**

* To identify requirements for the Online Ticket Booking System
* To design the system that will make online booking of tickets easier and fast.
* To test, validate and implement the system.

**1.3 MODULE DESCRIPTION:**

**Login Page:**

The login page is the first point of access for users of the Theme Park Ticketing System. It is where system users will input their credentials, such as their username and password, to gain access to the data and functionalities of the system according to their roles or permissions. The login page will have a simple and user-friendly design, with fields for entering the username and password, and a button to submit the information.

There will be a link to a “Forgot Password” page, in case the user forgets their password. The system will also have a feature that allows users to change their password after logging in.

The login page will be designed with security in mind, using encryption and other security measures to protect user information and prevent unauthorized access to the system. Once a user’s credentials have been verified, they will be directed to the appropriate section of the system based on their role or permission level.

The login page is the gatekeeper of the system that ensures that only authorized users can access the sensitive information and functionalities of the system, this is crucial for the security and integrity of the system.

**Dashboard Page:**

After logging into the Theme Park Ticketing System, users will be redirected to the default page, also known as the dashboard page. This page serves as the starting point for users to access the various features and functionalities of the system. The content and layout of this page will vary depending on the user’s role or permission level.

Admin users will be presented with a simple summary of key information such as total ticket sales, number of visitors, and revenue generated. They will also have quick access to important functions such as managing ticket pricing, creating new rides, and viewing reports.

On the other hand, Staff users will be presented with a simple welcome greeting and a list of the main functions they have access to such as scanning tickets, generating tickets, and creating a sales report.

Both Admin and Staff users will have access to different functionalities and features of the system but with different levels of permissions. This is to ensure that each user can only access the information and functionality that is relevant to their role in the park. The default page or dashboard page is designed to provide a clear and intuitive overview of the system’s key information and functionalities, making it easy for users to navigate and access the tools they need to effectively manage and operate the theme park.

**New Rides Page:**

The “Add New Ride” page is where the admin user can submit information about new rides that are being added to the theme park. This page will be accessible from the admin dashboard and will have a user-friendly design that allows for easy data entry.

The page will include fields for entering information such as the ride name, ride type, minimum height requirement, and a description of the ride. The admin user will also be able to upload images of the ride and set pricing for the ride.

The admin user will have the ability to set different prices for adults and children, as well as any additional fees such as express pass or fast lane. The admin user will also be able to set the ride’s availability, choosing from options such as always open, closed for maintenance or closed for the season.

The page will also include a preview button that allows the admin user to see how the ride will appear on the system’s website before submitting the information. Once the information is submitted, the new ride will be immediately available for purchase on the system’s website.

The “Add New Ride” page is a powerful tool that allows the admin user to quickly and easily update the theme park’s ride offerings, which can help attract more visitors and increase revenue.

**Rides List Page:**

The “Manage Rides” page is where the admin user can view and manage all the rides in the theme or amusement park. This page will be accessible from the admin dashboard and will be only visible to the admin users.

The page will have a list view of all the rides in the park, including the ride name, ride type, minimum height requirement, and a description of the ride. The admin user will also be able to see the ride’s pricing and availability.

The admin user will be able to perform various actions on the rides such as editing, deleting, and marking the ride as closed for maintenance or closed for the season. The admin user will also be able to upload images of the rides and change the ride’s pricing and availability.

The page will also include a search bar and sorting options to help the admin user quickly find the ride they are looking for. Additionally, the page will have a pagination feature that will allow the admin user to navigate through large numbers of rides easily.

The “Manage Rides” page gives the admin user a centralized location to view and manage all the rides in the theme or amusement park, this will help them to keep the park’s offerings up-to-date and attract more visitors. It also allows the admin user to monitor the status of each ride, ensuring that they are in good working order and ready for visitors.

**New Pricing Page:**

The “Manage Pricing” page is where the admin user can set up new pricing for new rides or promotional tickets. This page will be accessible from the admin dashboard and will be only visible to the admin users.

The page will have a list view of all the pricing options currently available in the system. The admin user will be able to add new pricing options by filling out a form that includes fields for the pricing name, adult price, child price and other additional fees such as express pass or fast lane.

The admin user will also be able to edit or delete existing pricing options. They will be able to assign these pricing options to specific rides or make them available as promotional tickets.

The page will also include a search bar and sorting options to help the admin user quickly find the pricing they are looking for. Additionally, the page will have a pagination feature that will allow the admin user to navigate through large numbers of pricing options easily.

The “Manage Pricing” page gives the admin user the flexibility to set up different pricing options for the rides and tickets, which can help to increase revenue and attract more visitors to the park. It also allows the admin user to make changes to the pricing options quickly and easily, which can help to keep the park’s offerings up-to-date and respond to market conditions.

**Pricing List Page:**

The “Manage Ticket Pricing” page is where the admin user can view and manage all the ticket pricing in the theme or amusement park. This page will be accessible from the admin dashboard and will be only visible to the admin users. The page will have a list view of all the ticket pricing options currently available in the system, including the pricing name, adult price, child price and any additional fees such as parking or meal vouchers. The admin user will also be able to see the number of tickets sold for each pricing option.

The admin user will be able to perform various actions on the ticket pricing such as editing, deleting, and creating new pricing options. They will also be able to assign these pricing options to specific dates or make them available as promotional tickets. The page will also include a search bar and sorting options to help the admin user quickly find the pricing they are looking for. Additionally, the page will have a pagination feature that will allow the admin user to navigate through large numbers of pricing options easily.

The “Manage Ticket Pricing” page gives the admin user a centralized location to view and manage all the ticket pricing options in the theme or amusement park. This will help them to keep the park’s offerings up-to-date and attract more visitors by adjusting pricing options as needed. It also allows the admin user to monitor the sales of each pricing option, ensuring that they are generating the desired revenue for the park.

**New Ticket Page:**

The page where the system users generate a ticket/tickets for the customer/visitor.

**Ticket List Page:**

This would likely be referred to as a “ticket sales management page” or a “ticket sales dashboard.” It would likely be accessible through a ticketing system or event management platform, and would allow users to view and manage a list of tickets that have been sold for a particular event or venue.

**Sales Report Page:**

The page where the printable list of the sold tickets on a selected range of date is shown along with he total amount of sale of the theme or amusement park.

**1.4 EXISTING SYSTEM:**

The weaknesses were identified in the Existing System:

* The system was time consuming and slow due to the manual data capture methods and tools involved because it required storage of records on paper which still needed to be compiled.
* The system had duplication of records (tickets) due to poor coordination between the agents involved in the process. In that at times they do not inform their workmates about the number of tickets bought due to ignorance or fatigue.
* The system was highly prone to loss of data since paper based filing of data has the weakness of being misplaced if not well managed. Further still fire was a great threat to the information kept since very many files are paper based and no fire proof and filing cabinets exists.
* The system did not cater for illiterate agents who felt left out. This is because they could not participate in the running and management of the various activities and records.

**1.5 PROPOSED SYSTEM:**

The Advantages of the Proposed system are as follows :

* One of the major benefits that e ticketing will offer is that it does not require a paper ticket/record as a confirmation of your booking.
* The confirmation of your booking deception in the organizer computerized ticketing system in an electronic format.
* You need not to carry a paper ticket while traveling and hence there is no fear of losing the ticket or forgetting it behind since they exist in electronic format.
* Helps minimize transport costs by just logging on the net and performing all the necessary transactions and security together with providing flexibility and convenience.
* This type of making reservations reduces the risk of you losing your paper tickets since print outs of your e-tickets can be taken anytime and anywhere.
* No queue for buying tickets and customers can access more information and find other deals online.

**2. REQUIREMENT SPECIFICATION**

**2.1 HARDWARE REQUIREMENT:**

The hardware specification is necessary to support the proposed system, which have been identified, ordered, delivered, installed and tested at the time of installation.

* Processor : Inteli3
* Hard Disk : 40GB
* RAM : 4GB
* Mouse : Optical
* Printer : HP Laser

**2.2 SOFTWARE REQUIREMENT**

The software specification is the one says about the development environment of the package.

* Operating System : Windows 10
* User Interface : HTML, CSS
* Client-side Scripting : JavaScript
* Programming Language : PHP
* Database : MY SQL
* Server Deployment : Apache

**2.3 APPLICATION SPECIFICATION:**

**Introduction to PHP**

The original release of PHP was designed and created by Rasmus Lerdorf back in the middle of the 90s as a way of making various common web tasks easier and less repetitive. Back then, the main goal was to have the minimum amount of logic as was possible in order to achieve results, and this led to PHP being HTML-centric – that is, PHP code was embedded inside HTML. PHP 3 also finally made the language extensible – something that was serious lacking from prior versions With PHP 3, the language had gained limited object-oriented support, and this only added extra fuel to the fire of PHP's growth.

* PHP is a server side scripting language, like ASP.
* PHP stands for Hypertext Pre Processor.
* PHP scripts are executed on the server.
* PHP supports many databases(MYSQL, Informix, Oracle, Sybase, Solid, Postgre SQL, Generic ODBC, etc.).
* PHP is open source software (OSS).
* PHP is free to download and use.

The main features of the PHP scripting language include the following:

**Open Source**

PHP is an open source language and is freely available for use. The community of open source PHP developers provides technical support and is constantly improving updating the core PHP functionalities. The PHP Extension and Application Repository system provides and maintains a library of PHP code packages that are available for use. The packages can include functions such as authentication, caching, encryption, error handling etc.

**Compatibility**

PHP provides high compatibility with leading operating systems and web servers such as enabling it to be easily deployed across several different platforms.

**File Handling**

PHP be used to read text and generate files in various formats such as PDF and XML. Using the file manipulation functions, files and documents can be uploaded and stored on the server. The uploaded documents can be accessed and manipulated through PHP code.

**Improved Performance**

The PHP complier includes features to optimize and improve the quality of compiled code by reducing the size execution time of the code thereby leading to improved performance.

**Debuggers**

Several debuggers are available with PHP enabling developers to identify and analyze the code for potential bugs and bottlenecks

**Sessions**

PHP provides extensive session and cookie management features and functions enabling the creation and development of personalized web pages.

**Graphics**

PHP can be used to generate images and graphics dynamically. Using the image functionalities available with PHP, the header information of images can be accessed and manipulated. The GD library of PHP includes a host of features and functionalities that can be used to create images in various formats such as gif, jpeg and png.

**Extensible**

The source code of PHP can be modified to include custom created extensions and components thereby increasing its extensible.

**Features of MYSQL**

The following list shows the most important properties of MYSQL.

**Client/Server Architecture**

MYSQL is a client/server system. There is a database server (MYSQL) and arbitrarily many clients (application programs), which communicate with the server; that is, they query data, save changes, etc. The clients can run on the same computer as the server or on another computer (communication via a local network or the Internet).

**SQL compatibility**

MYSQL supports as its database language as its name suggests – SQL (Structured Query Language). SQL is a standardized language for querying and updating data and for the administration of a database.

**Sub selects**

MYSQL is capable of processing a query in the form SELECT \* FROM table1 WHERE x IN (SELECT y FROM table2) (There are also numerous syntax variants for sub SELECTs.)

**Views**

Views relate to an SQL query that is viewed as a distinct database object and makes possible a particular view of the database. MYSQL has supported views since version 5.0.

**Stored procedures**

Stored procedures are generally used to simplify certain steps, such as inserting or deleting a data record. For client programmers this has the advantage that they do not have to process the tables directly, but can rely on SPs. Like views, SPs help in the administration of large database projects. SPs can also increase efficiency.

**Triggers**

Triggers are SQL commands that are automatically executed by the server in certain database operations (INSERT, UPDATE, and DELETE).

**Unicode**

MY SQL has supported all conceivable character sets since version 4.1, including Latin-1, Latin-2, and Unicode (either in the variant UTF8 or UCS2).

**Full-text search**

Full-text search simplifies and accelerates the search for words within a text field.

**Replication**

Replication allows the contents of a database to be copied (replicated) onto a number of computers. In practice, this is done for two reasons: to increase protection against system failure (so that if one computer goes down, another can be put into service) and to improve the speed of database sql queries.

**Transactions**

In the context of a database system, a transaction means the execution of several database operations as a block. The database system ensures that either all of the operations are correctly executed or none of them. This holds even if in the middle of a transaction there is a power failure, the computer crashes, or some other disaster occurs.

**Foreign key constraints**

Foreign key constrains are rules that ensure that there are no cross references in linked tables that lead to nowhere. MY SQL supports foreign key constraints for tables.

**Programming languages**

Programming languages are quite a number of APIs (application programming interfaces) and libraries for the development of MY SQL applications. For client programming you can use, among others, the languages C, C++, Java, Perl, PHP, Python, and Tcl.

**Platform independence**

Platform independence is not only client applications that run under a variety of operating systems; MYSQL itself (that is, the server) can be executed under a number of operating systems. The most important are Apple Macintosh OS X, Linux, Microsoft Windows, and the UNIX variants, such as Open BSD, Net BSD, SGI Iris, and Sun Solaris. MySQL is a [relational database management system](http://en.wikipedia.org/wiki/Relational_database_management_system) (RDBMS), and ships with no [GUI](http://en.wikipedia.org/wiki/Graphical_user_interface) tools to administer MySQL databases or manage data contained within the databases. Users may use the included [command line](http://en.wikipedia.org/wiki/Command_line) tools, or use MySQL "front-ends", desktop software and web applications that create and manage MySQL databases, build database structures, back up data, inspect status, and work with data records. The official set of MySQL front-end tools, [MySQL Workbench](http://en.wikipedia.org/wiki/MySQL_Workbench) is actively developed by Oracle, and is freely available for use.[[](http://en.wikipedia.org/wiki/MySQL#cite_note-26)

MySQL ships with many [command line](http://en.wikipedia.org/wiki/Command_line) tools, from which the main interface is 'MySQL' client. Third-parties have also developed tools to manage, optimize, monitor and backup a MySQL server, some listed below. All these tools work on \*NIX type operating systems, and some of them also on Windows.

Windows

Windows Professional gives all the benefits of Windows Home Edition, plus additional remote access, security, performance, manageability and multi-lingual features that make the operating system to suit for businesses of all sizes and user who demand the most out of their computing experience.

**Features**

* New user interface makes it easy to find details according to user needs.
* Network Setup Wizard easily connects & shares the computers and devices.
* Windows Messenger the ultimate communications & collaboration tool with instant messaging, voice and video conferencing, and application sharing.
* Windows Media Player for Windowssingle place for finding, playing, organizing, and storing digital media.
* Help & Support Center easy to recover from problems and get help and support when needed.
* File and Folder Management Windows provides several new ways to arrange and identify files when viewing them in folders such as My Documents.

Windows added the ability for windows to use “Visual Styles” to change the user interface. However visual styles must be cryptographically signed by Microsoft to run. Luna is the name of the new visual style the ships with Windows and is enabled by default for machines with more than 64 MB of RAM. Luna refers only to one particular visual style, not to all of the new user interface features of Windows as a whole. Some users “patch” the uxtheme.dll file restricts the ability to use visual styles, created by the general public or the user, on Windows

**3. SYSTEM DESIGN AND DEVELOPMENT**

**3.1 FILE DESIGN**

**Flat File Database**

A flat file database is a database designed around a single table. The flat file design puts all database information in one table, or list, with fields to represent all parameters. A flat file may contain many fields, often, with duplicate data that are prone to data corruption. If data between two flat files have to be merged, it is needed to copy the relevant information from one file to the other. There is no automation between flat files.

If two or more flat files that contain client addresses, it is required to manually modify the address parameters in each file that contains that client’s information. Changing information in one file has no bearing on other files. Flat files offer the functionality to store information, manipulate fields, print or display formatted information and exchange information with others, through email and over the Internet. Some flat files may be attached to external files, such as text editors, to extend functionality and manage related information.

**Relational File Database**

A relational database, on the other hand, incorporates multiple tables with methods for the tables to work together. The relationships between table data can be collated, merged and displayed in database forms. Most relational databases offer functionality to share data:

* Across networks
* Over the Internet
* With laptops and other electronic devices, such as palm pilots
* With other software systems

Designing flat file databases is simple and requires little design knowledge. Flat files can be developed using just about any database engine. Flat files can be created in relational database engines by not taking advantage of relational design concepts. Designing a relational database takes more planning than flat file databases. With flat files, it is possible to add information, as you deem necessary. With relational databases, it is required to be careful to store data in tables such that the relationships make sense. Building a relational database is dependent upon the ability to establish a relational model. The model must fully describe how the data is organized, in terms of data structure, integrity, querying, manipulation and storage. Relational databases allow defining certain record fields, as keys or indexes, to perform search queries, join table records and establish integrity constraints.

Search queries are faster and more accurate when based on indexed values. Table records can be easily joined by the indexed values. Integrity constraints can be established to ensure that table relationships are valid. If the project is able to establish a one-to-many relationship in the data tables, relational database should be used because a flat file is not sufficient to handle the data processing needs. Relational databases offer more robust reporting with report generators that filter and display selected fields.

Relational databases offer the capability to building own reporting modules. Most relational databases also offer the capability to import and export data from other software. There are three primary relational database systems, proprietary, open source and embedded. Proprietary relational databases require the use of proprietary development languages, often times, to complement SQL. Microsoft Access, for example, combines Visual Basic with SQL. Open source databases, such as MySQL, are distributed freely to encourage user development.

Embedded, relational databases are packaged as part of other software packages, such as with tax-preparation software packages. The vendor supplies the database, and all manipulation tools, to control the database structure. These databases are, often times, accompanied with tools to provide audit trails of transactions. Proposed project uses relational database files implementing SQL queries in MS Access. So 1 to many relationships can be established between tables and the table data can be accessed soon even in the non-indexed locations.

**3.2 INPUT DESIGN**

The input design is the process of entering data to the system. The input design goal is to enter to the computer as accurate as possible. Here inputs are designed effectively so that errors made by the operations are minimized. The inputs to the system have been designed in such a way that manual forms and the inputs are coordinated where the data elements are common to the source document and to the input. The input is acceptable and understandable by the users who are using it. Once identified, appropriate input media are selected for processing.

The input design also determines the user to interact efficiently with the system. Input design is a part of overall system design that requires special attention because it is the common source for data processing error. The goal of designing input data is to make entry easy and free from errors.

The main objectives that are done during the input design are:

* Data is collected from the source
* Transfer of data to an input form is done
* Data is converted to a computer acceptable form
* The converted data is verified
* Data is checked for its accuracy
* Data is transmitted to the computer
* Validation of input data is done

**3.3 OUTPUT DESIGN**

Computer output is most important and direct source of information to the user. Efficient of intelligible output should improve the system relationship with the user and help in decision-making. Major forms of output are hard copy from the printer and soft copy from the CRT unit.

The output design was done so that results of processing could be communicated to the users. The various outputs have been designed in such a way that they represent the same format that the office and management used to.

Computer output is the most important and direct source of information to the user. Efficient, intelligible output design should improve the systems relationships with the user and help in decision making. A major form of output is the hardcopy from the printer.

* Scalability of the device according to the output format required
* The need of hard copy
* The response time taken
* The detail specification needed

**3.4 DATABASE DESIGN**

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 data base system must provide for the safety of the information solved, despite system crashes or due to attempts at unauthorized access. For developing an efficient database proposed project 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**

In database design several objectives are designed such as:

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

For achieving the above mentioned criteria we have to make use of various features that are available with the RDBMS by enforcing integrity constrains, it is possible to ensure data integrity and reduce data inconsistency to a great extent.

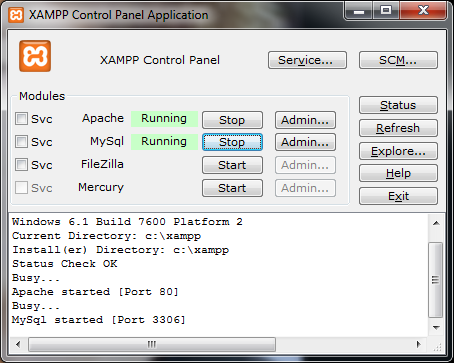
**Database Backups**

Another area that often gets lost in the layers of security is the critical area of database backup and recovery. As a part of whatever backup type is used, testing recovery of data is mandatory. Further, check pointing has been improved to be done more frequently, also easing the recovery process. With frequent check pointing, as well as transaction logging and making regular backup copies, backup and recovery of databases is made more straightforward.

Specifically, the InnoDB transactional model allows for commit, rollback and crash recovery. By adding locking capabilities for users, having many users access the database at the same time becomes faster and more stable. To ensure that InnoDB is available with your installation, configure your package with the ‘with-innodb’ flags. You will also want to specify InnoDB options in your ‘my.cnf’ file. Details on these set-up options can be found at the MySQL Documentation on InnoDB Start-up Options.

The following Fig 3.3 describes the Xampp Control Panel Application form which shows the start the MySQL connectivity.

**Fig 3.3 Xampp Control Panel Application**



**Privileges in MySQL Server**

The privileges can get very granular, so it is important that they are used in a well-planned fashion. The types of privileges include:

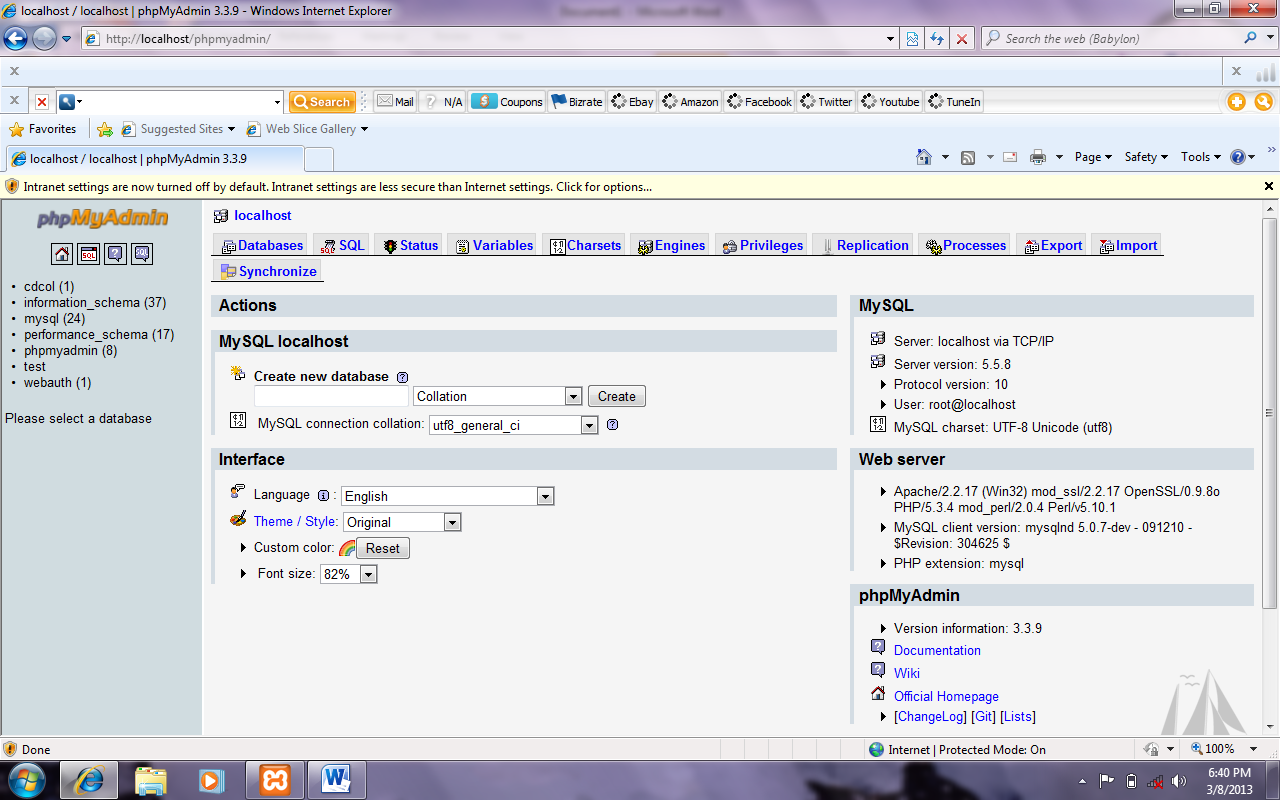
* Alter
* Delete
* Create
* Drop
* Execute
* Select
* Update

Once a database is completely set up, these privileges should be reviewed prior to going to any usage of the database to ensure that the privileges were set up appropriately.

## Create a Database in MySQL Server

The CREATE DATABASE statement is used to create a database table in MySQL.We must add the CREATE DATABASE statement to the mysql\_query () function to execute the command. The following form describes to create a database named "product details".

The Fig 3.3.1 describes how to create a new database in the My\_SQL database. It contains the database name and also the locations were the database is created and where it is accessed.

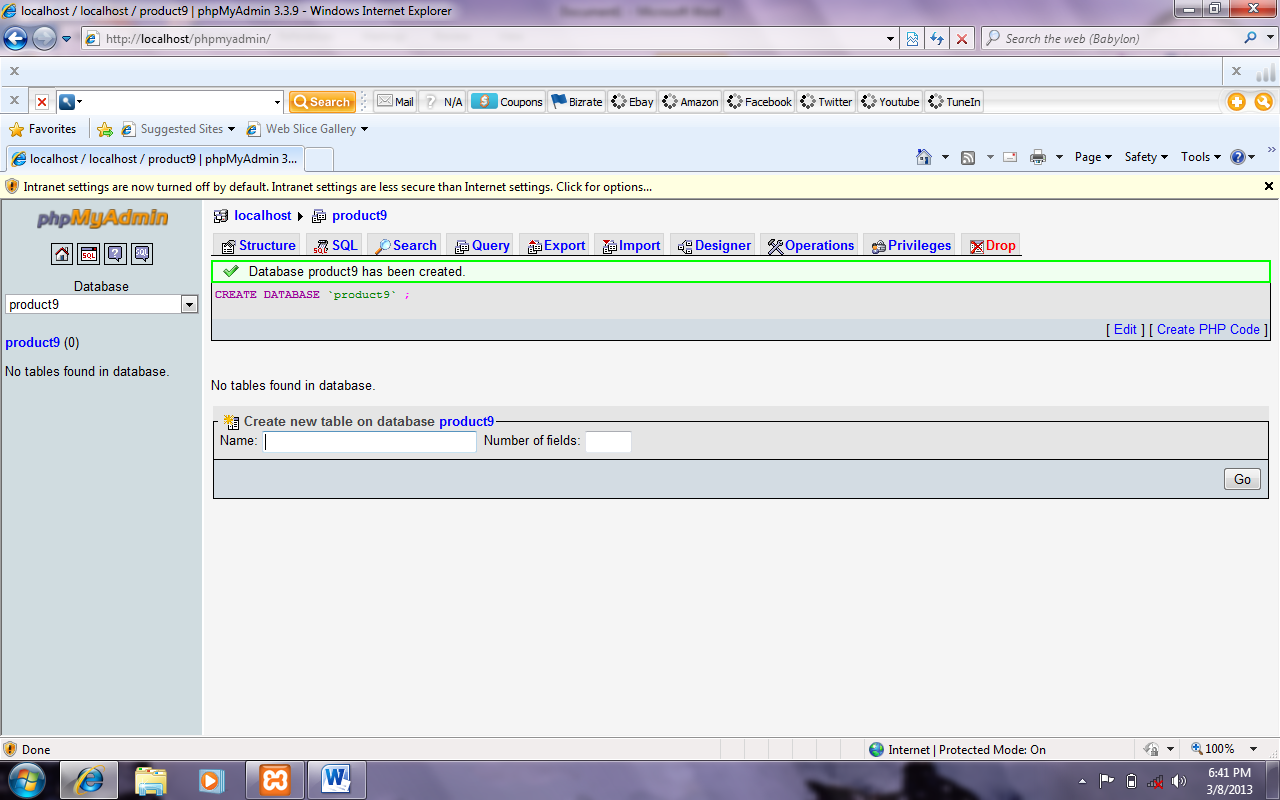


**Fig 3.3.1 Creating a New Database in MySQL**

## Create a Table in the MySQL Server

The CREATE TABLE statement is used to create a table in MySQL. We must add the CREATE TABLE statement to the mysql\_query () function to execute the command. The following figure describes how to create a tables in the database.

The following Fig 3.3.2 describes to create a new table in the database job scheduling. It includes the table name and the entities used in the table, its size and also its type.



**Fig 3.3.2 Create a New Table in the Database**

There are several GUI consoles available to make the management of MySQL database easier. For instance, MySQL Explorer allows several management processes to be done via an interface that runs on several windows platforms. The MySQL team also has a version in beta called MySQL Control Center.

**3.5 DATA FLOW DIAGRAM**

A **data flow diagram** (**DFD**) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. Often they are a preliminary step used to create an overview of the system which can later be elaborated. DFD can also be used for the visualization of data processing(structured design).

A DFD shows what kinds of data will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of processes, or information about whether processes will operate in sequence or in parallel (which is shown on a flowchart).

360px-DataFlowDiagram_Example

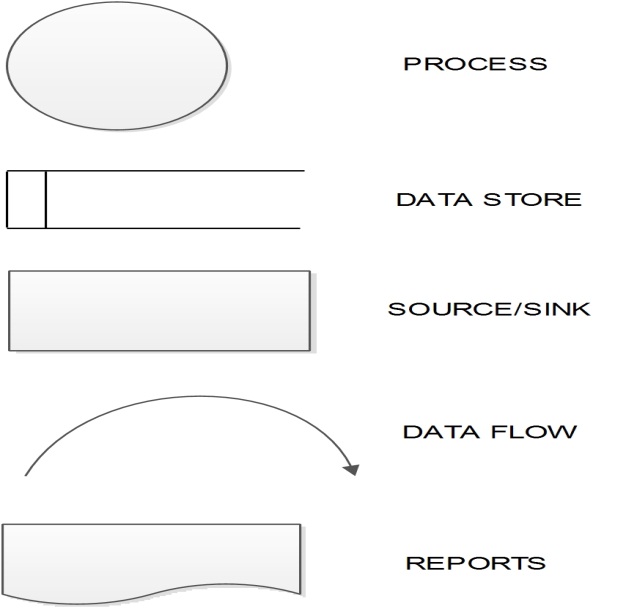
**Fig 1: Data flow diagram example**

It is common practice to draw the context-level data flow diagram first, which shows the interaction between the system and external agents which act as data sources and data sinks. On the context diagram the system’s interactions with the outside world are modeled purely in terms of data flows across the system boundary. The context diagram shows the entire system as a single process, and gives no clues as to its internal organization.

This context-level DFD is next “exploded”, to produce a Level 0 DFD that shows some of the detail of the system being modeled. The Level 0 DFD shows how the system is divided into sub-systems (processes), each of which deals with one or more of the data flows to or from an external agent, and which together provide all of the functionality of the system as a whole. It also identifies internal data stores that must be present in order for the system to do its job, and shows the flow of data between the various parts of the system.

Data flow diagrams were proposed by Larry Constantine, the original developer of structured design, based on Martin and Estrin’s “data flow graph” model of computation.

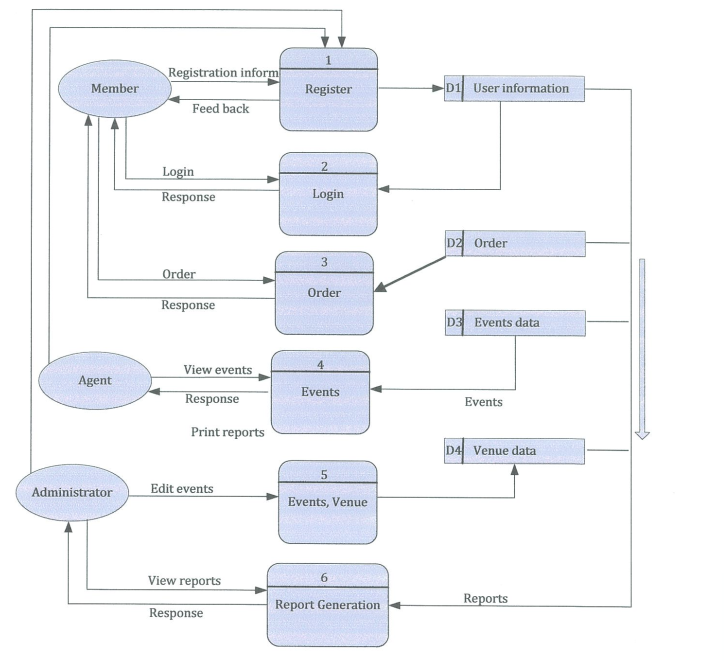
Data flow diagrams are one of the three essential perspectives of the structured-systems analysis and design method (SSADM). The sponsor of a project and the end users will need to be briefed and consulted throughout all stages of a system’s evolution. With a data flow diagram, users are able to visualize how the system will operate, what the system will accomplish, and how the system will be implemented. The old system’s data flow diagrams can be drawn up and compared with the new system’s data flow diagrams to draw comparisons to implement a more efficient system. Data flow diagrams can be used to provide the end user with a physical idea of where the data they input ultimately has an effect upon the structure of the whole system from order to dispatch to report. How any system is developed can be determined through a data flow diagram model.



**Fig 2: Data flow diagram**

In the course of developing a set of leveled data flow diagrams the analyst/designers is forced to address how the system may be decomposed into component sub-systems, and to identify the transaction data in the data model. Data flow diagrams can be used in both Analysis and Design phase of system development life cycle (SDLC).There are different notations to draw data flow diagrams (Yourdon & Coad and Gane&Sarson), defining different visual representations for processes, data stores, data flow, and external entities.

**DFD Diagram of Online Theme Park Ticket Booking System:**



**4. TESTING AND IMPLEMENTATION**

**4.1 SYSTEM TESTING**

The most important phase in system development life cycle is system testing. The number and nature of errors in a newly designed system depends on the system specification and the time frame given for the design.

A newly designed system should have all the subsystems working together, but in reality each subsystems work independently. During the phase, all the subsystems are gathered into one pool and tested to determine whether it meets the user requirements.

Testing is done in two level-Testing of individual modules and test the entire system. During the system testing, the system is used experimentally to ensure the software will run according to the specification ns and in the way the user expects. Each test case is designed with the intent of finding errors in the way the system will process.

Testing is vital to the success of the system. System testing makes a logical assumption that if all the parts of the system are correct; the goal will be successfully achieved non- testing leads to error that may not appear until months later. This creates two problems.

* The time lag between the cause and appearance of the problem.
* The effort of system error on files and records within the system.

The Testing Steps are:

* Unit testing
* Integrated testing
* Validation testing
* White Box testing

**4.2 TEST PLAN**

The system has been designed and developed according to the current requirements of the user. At the same time the system is very flexible and extensible, Hence, future enhancements, if needed can be made without much difficulty, so new applications can be developed and it be integrated with the existing one very easily.

The following future enhancements may be worthwhile to make the tool usable to a wider section of users.

* Currently only text-based reports are handled. It can be extended to include graphics and images. Also music and audio clips are considerable.
* User level authentication and authorization may be of use in certain circumstances.

**4.3 TESTING METHODS AND IMPLEMENTATION**

System implementation is the process of developing the system based on the user requirement that has to be enforced in any system while development. Security window prohibits unauthorized users entering the system. Implementation is the stage of the project when the system design is turned into fully working system. This stage consists of following steps.

* Testing the developed program with the sample data.
* Detecting and correction of internal errors.
* Testing the system to see if it meets the user requirements.
* Feeding the real time data.
* Making changes as desired by the user.
* Training user personal.

The purpose of System Implementation can be summarized as follows: making the new system available to a prepared set of users (the deployment), and positioning on-going support and maintenance of the system within the Performing Organization (the transition). At a finer level of detail, deploying the system consists of executing all steps necessary to educate the Consumers on the use of the new system, placing the newly developed system into production, confirming that all data required at the start of operations is available and accurate, and validating that business functions that interact with the system are functioning properly. A key difference between System Implementation and all other phases of the lifecycle is that all project activities up to this point have been performed in safe, protected, and secure environments, where project issues that arise have little or no impact on day-to-day business operations. Once the system goes live, however, this is no longer the case.

**Unit Testing**

Testing of individual programs or modules is known as unit testing. Unit testing is done both during documentation and testing phase. Unit testing focuses on verification of effort on the smallest of software design. Modules using the detailed design description as a guide, important control paths are tested to uncover errors within the boundary of the module. The relative complexity is test and errors detected as a result are limited by the constraints scope established for unit testing. Unit testing is always white box oriented and the step can be conducted in parallel for multiple modules.

**Integration Testing**

Integration testing is a systematic technique for constructing the program structure while at the same time conducting test to uncover errors associated with interfacing. The objective is to take unit - tested modules and build a program structure that has been dictated by design. Careful test planning is required to determine the extent and nature of system testing to be performed and to establish criteria by which the result will be evaluated.

**Validation Testing**

The most common web application security weakness is the failure to properly validate input coming from the client or environment before using it. The weakness leads to almost all of the major vulnerabilities in web applications, such as cross site scripting, SQL injection, interpreter injection. Data from an external entity or client should never be trusted, since it can be arbitrarily tampered with by an attacker. Validation doesn't just mean putting your pages through some web driven testers. It also means test-driving it with friends, relatives, co-workers, and strangers. Everyone has a different system and way of working, so ask for others to test-drive your styles or themes before you make them public.

**White Box Testing**

White box testing is a test case method that uses control structure and procedural design to drive test cases using white box testing method. Software engineer can test cases that:

* Exercise all logical decisions on their true or false sites.
* Guarantee that all independent paths with a module have been exercised at least once.
* Exercise internal data structure to ensure validity.
* Execute all loops at their boundaries and their operational bounds.

White box testing sometimes called as glass box testing is a test case design method that uses the control structures of the procedural design to derive test cases.

Using White Box testing methods, the software engineer can derive test case, that guarantee that all independent paths with in a module have been exercised at least once, exercise all logical decisions on the true and false sides, execute all loops at their boundaries and within their operational bounds, exercise internal data structures to ensure their validity. “Logic errors and incorrect assumptions are inversely proportional to the probability that a program path will be executed“. Black Box Testing: Black box testing, also called as behavioral testing, focuses on the functional requirements of the software. That is, black box testing enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements for a program.

Black box testing focuses on the fundamental requirements on software and on input and output of the module. It enables the software engineers to derive set of input condition that will truly exercise all functional requirements of a program. Black box testing is rather a contemporary approach that is likely to uncover different class of errors.

It attempts to find out errors in the following category:

* Incorrect and missing functions
* Performance errors
* Initialization and termination errors

**6. CONCLUSION**

The entire project has been developed and deployed as per the requirements stated by the user, it is found to be bug free as per the testing standards that is implemented. Any specification-traced errors will be concentrated in the coming versions, which are planned to be developed in near future. The system at present does not take care off the money payment methods, as the consolidated constructs need SSL standards and are critically to be initiated in the first face, the application of the credit card transactions is applied as a developmental phase in the coming days. The system needs more elaborative technicality for its inception and evolution.

**BIBLIOGRAPHY**

[1] Androutsopoulos, J. Koutsias, K.V. Chandrinos, and C.D. Spyropoulos. An Experimental Comparison of Naive Bayesian and Keyword-Based Anti-Spam Filtering with Encrypted Personal E-mail Message.In Proc. SIGIR 2000, 2000.

[2] The Anti-phishing working group. http://www.antiphishing.org/.

[3] Neil Chou, Robert Ledesma, Yuka Teraguchi, Dan Boneh, and John C.Mitchell. Client-side defense against web-based identity theft. In Proc.NDSS 2004, 2004.

[4] Cynthia Dwork, Andrew Goldberg, and Moni Naor. On Memory-Bound Functions for Fighting Spam. In Proc. Crypto 2003, 2003.

[5] EarthLink. ScamBlocker. http://www.earthlink.net/software/free/toolbar/.

[6] David Geer. Security Technologies Go Phishing. IEEE Computer, 38(6):18–21, 2005.

[7] John Leyden.Trusted search software labels fraud site as ’safe’ http://www.theregister.co.uk/2005/09/27/untrusted search/.

[8] Microsoft.Sender ID Framework.http://www.microsoft.com/mscorp/safety/technologies/senderid/default.mspx.

[9] Netcraft. Net craft toolbar. http://toolbar.netcraft.com/.

[10] PhishGuard.com.Protect Against Internet Phishing Scams .http://www.phishguard.com/.

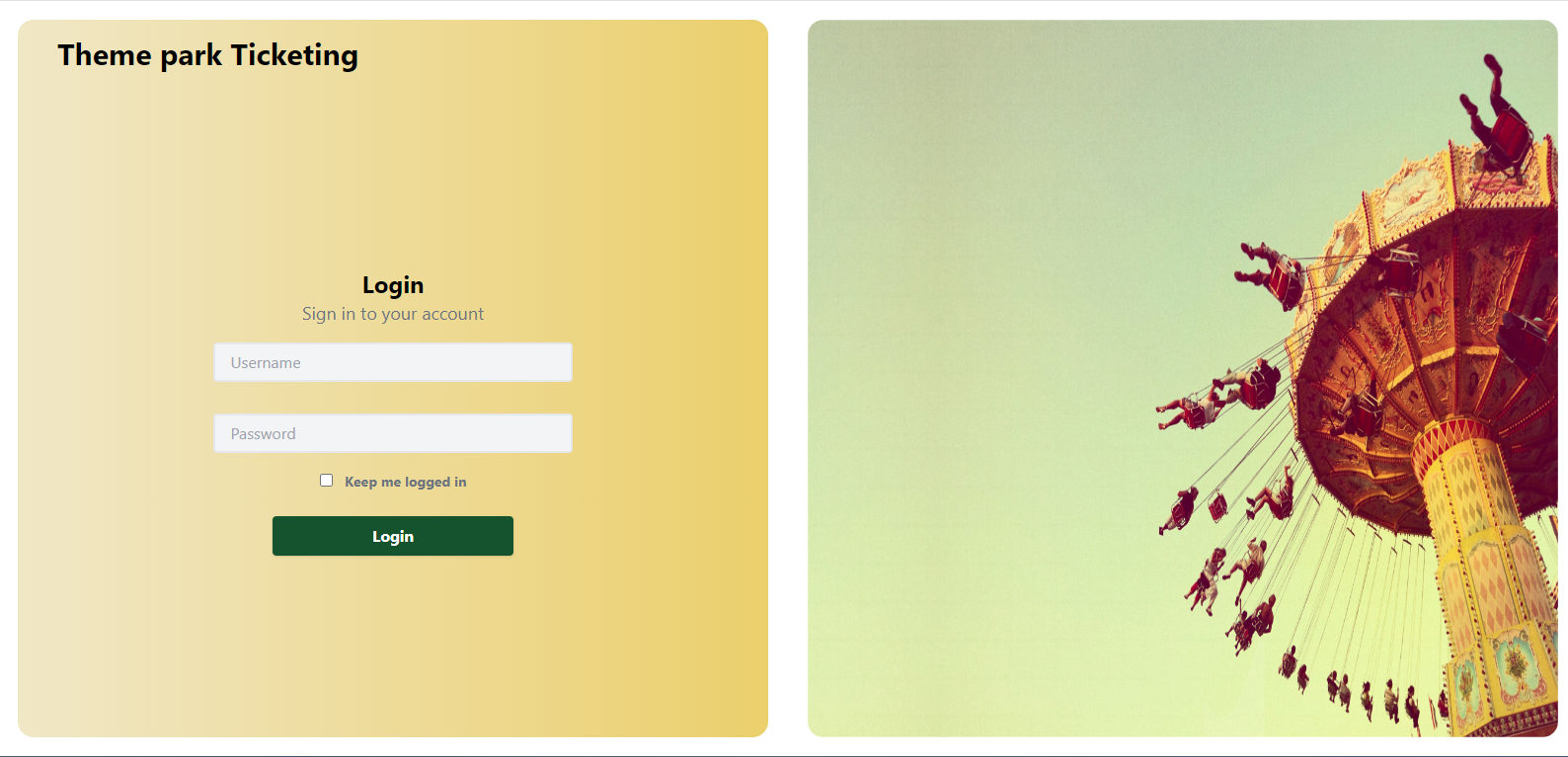
[11] Jonathan B. Postel. Simple Mail Transfer Protocol.RFC821:http://www.ietf.org/rfc/rfc0821.txt.

[12] Georgina Stanley. Internet Security-Gone phishing. http://www.cyota.com/news.asp?id=114.

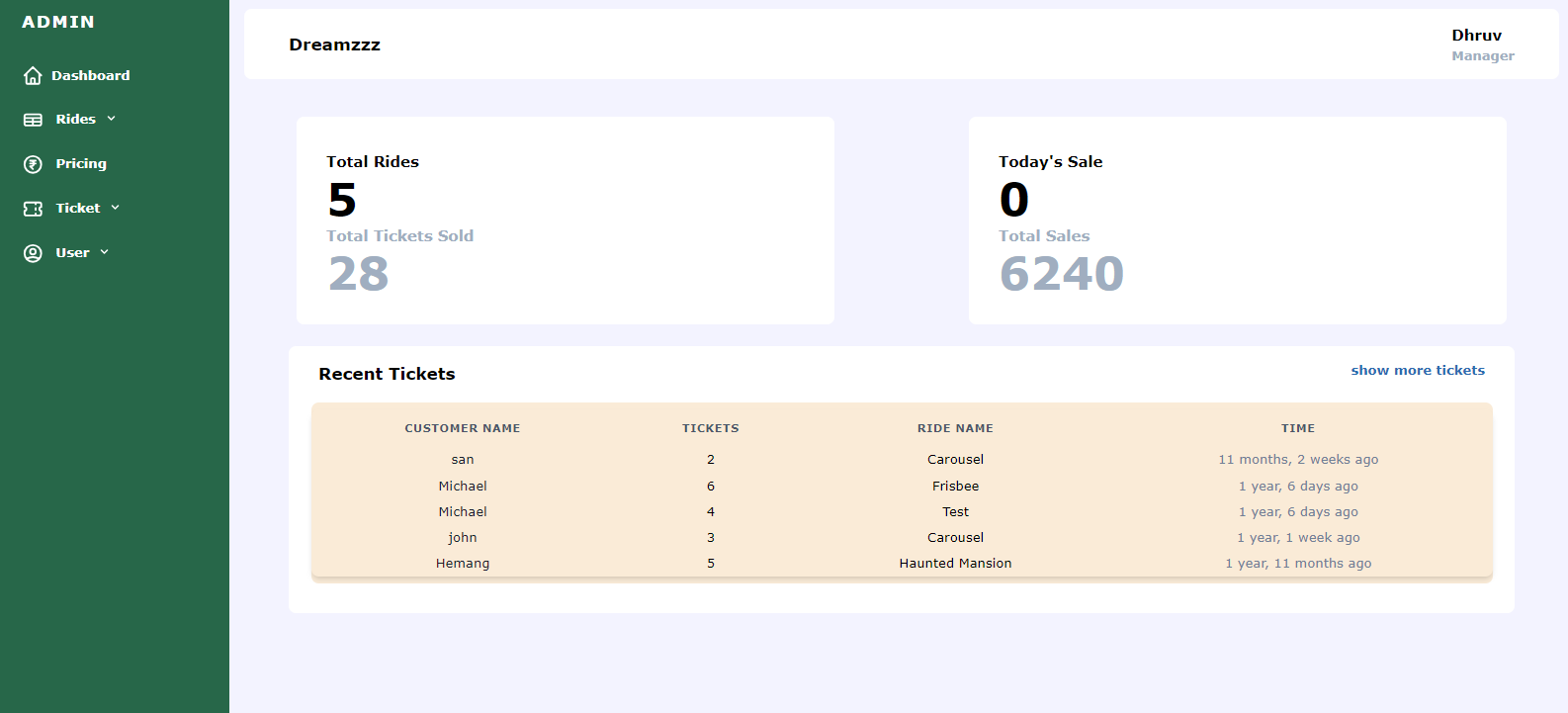
[13] MengWengWong. Sender ID SPF. http://www.openspf.org/whitepaper.pdf.

**Screen Shots**

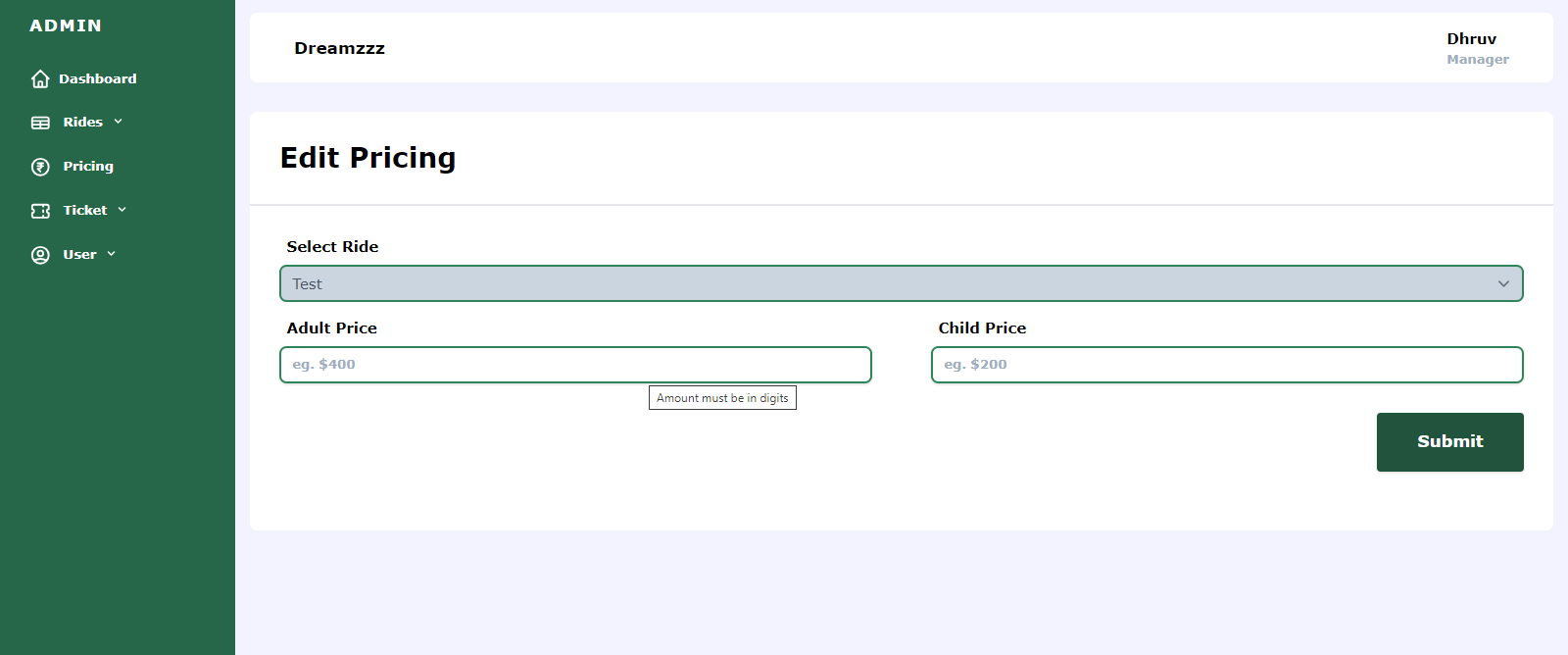
**Login Page**

****

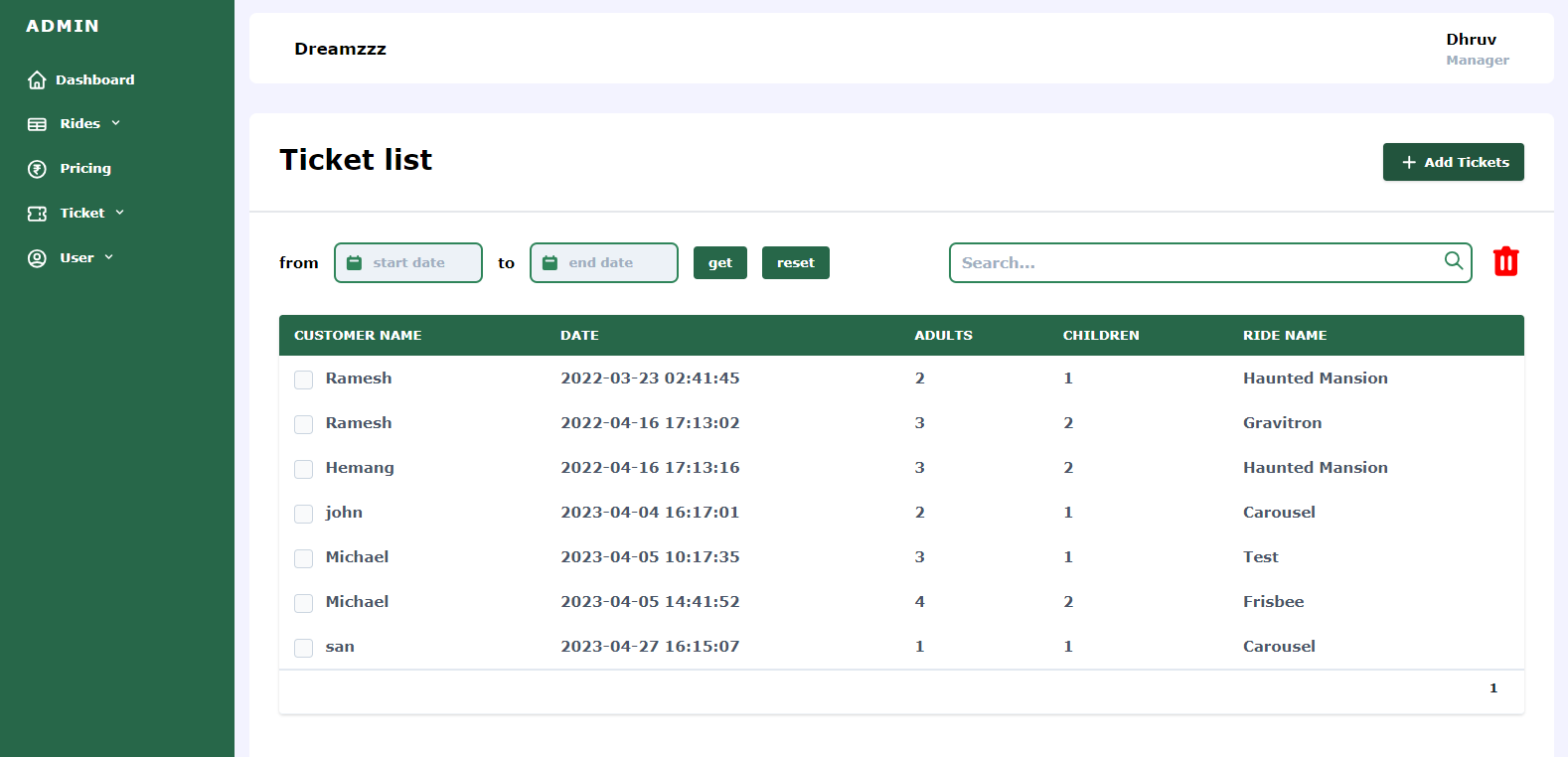
**Admin Dashboard:**

****

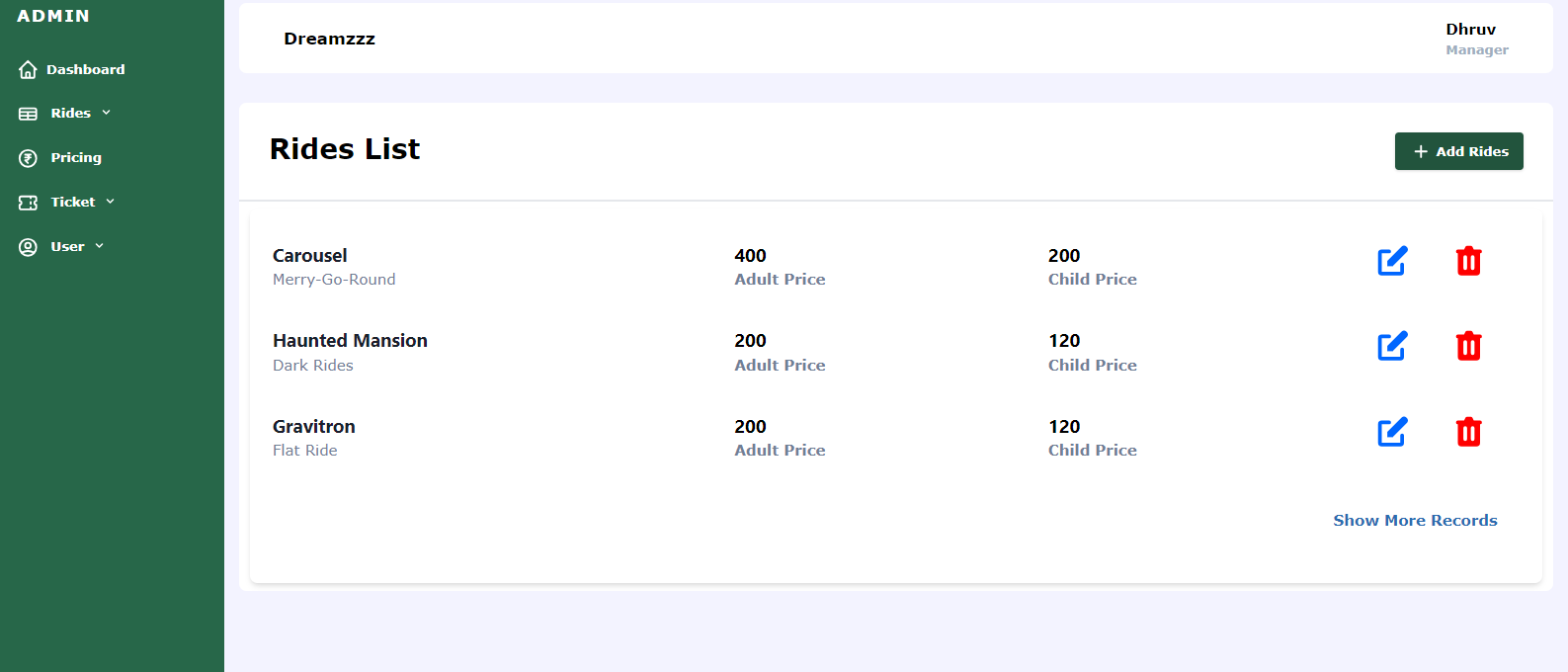
**Ticket Pricing:**

****

**Manage Ticket**

****

**Riders Management**

****

**Sample Code:**

<?php include 'util/session.php' ?>

<?php include 'util/isadmin.php' ?>

<!DOCTYPE html>

<html lang="en">

<head>

<?php include 'util/links.php' ?>

<title>Add Rides</title>

</head>

<body style="background: #F3F3FF">

<div class="md:flex">

<?php include 'components/sidebar.php'; ?>

<div class="grow" style="flex-grow: 1;">

<?php include 'components/navbar.php' ?>

<div class=" bg-white m-4 rounded-lg mt-8 font-bold">

<div class="border-b-2 p-4 lg:p-8">

<p class="text-2xl lg:text-3xl font-bold">Add Rides</p>

</div>

<div class="p-4 lg:p-8">

<form action="util/addride\_functionality.php" method="POST">

<div class="pb-4">

<p class=" pl-2 pb-2">Ride Name</p>

<input pattern="[A-Za-z ]{3,40}" title="Only Letters allowed and between size of 3 to 40." required type="text" name="ride\_name" placeholder="eg. Roller Coaster" class="text-sm w-full border-2 rounded-lg shadow appearance-none border-green-700 py-2 px-3 text-gray-700 focus:outline-none focus:border-green-700 " />

</div>

<div class="pb-4">

<p class=" pl-2 pb-2">Ride Description</p>

<input pattern="[A-Za-z ]{3,120}" title="Only Letters allowed and between size of 3 to 120." type="text" required name="ride\_description" placeholder="eg. Child's ride for fun" class="text-sm w-full border-2 rounded-lg shadow appearance-none border-green-700 py-2 px-3 text-gray-700 focus:outline-none focus:border-green-700 " />

</div>

<div class="lg:flex">

<div class="pb-4 flex-1 lg:pr-8">

<p class=" pl-2 pb-2">Adult Price</p>

<input type="text" pattern="[0-9]{0,11}" title="Amount must be in digits" required name="adult\_price" placeholder="eg. $400" class="text-sm w-full border-2 rounded-lg shadow appearance-none py-2 px-3 border-green-700 text-gray-700 focus:outline-none focus:border-green-700 " />

</div>

<div class="pb-4 flex-1 lg:pl-8">

<p class=" pl-2 pb-2">Child Price</p>

<input type="text" pattern="[0-9]{0,11}" title="Amount must be in digits" required name="child\_price" placeholder="eg. $200" class="text-sm w-full border-2 rounded-lg shadow appearance-none py-2 px-3 border-green-700 text-gray-700 focus:outline-none focus:border-green-700 " />

</div>

</div>

<div class="pb-4 flex justify-end">

<button class="shadow w-full lg:w-40 lg:h-16 text-lg bg-green-900 hover:bg-green-700 focus:shadow-outline focus:outline-none text-white font-bold py-2 px-4 mt-4 rounded" type="submit">

Submit

</button>

</div>

<div class="mt-8 flex justify-center">

<?php if (isset($\_GET['status'])) { ?>

<p class="text-sm text-green-600"> <?php echo $\_GET['status'] ?> </p>

<?php } ?>

<?php if (isset($\_GET['error'])) { ?>

<p class="text-sm text-red-600"> <?php echo $\_GET['error'] ?> </p>

<?php } ?>

</div>

</form>

</div>

</div>

</div>

</div>

</body>

</html>