

**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**

**BELAGAVI - 590018.**

**A MINI - PROJECT REPORT ON**

**“MUSEUM MANAGEMENT SYSTEM”**

***submitted in partial fulfillment of requirements for the fifth semester***

**DATABASE MANAGEMENT SYSTEM LABORATORY**

***for the course of***

**BACHELOR OF ENGINEERING**

***in***

**COMPUTER SCIENCE**

***submitted by***

**MAYANK KUMAR SHAW** **KUMAR PRINCE**

**(1ST17CS090)** **(1ST17CS084)**

***under the guidance of***

**Mrs. Bilwashree H**

**Asst.Prof, Dept. of CSE**

**&**

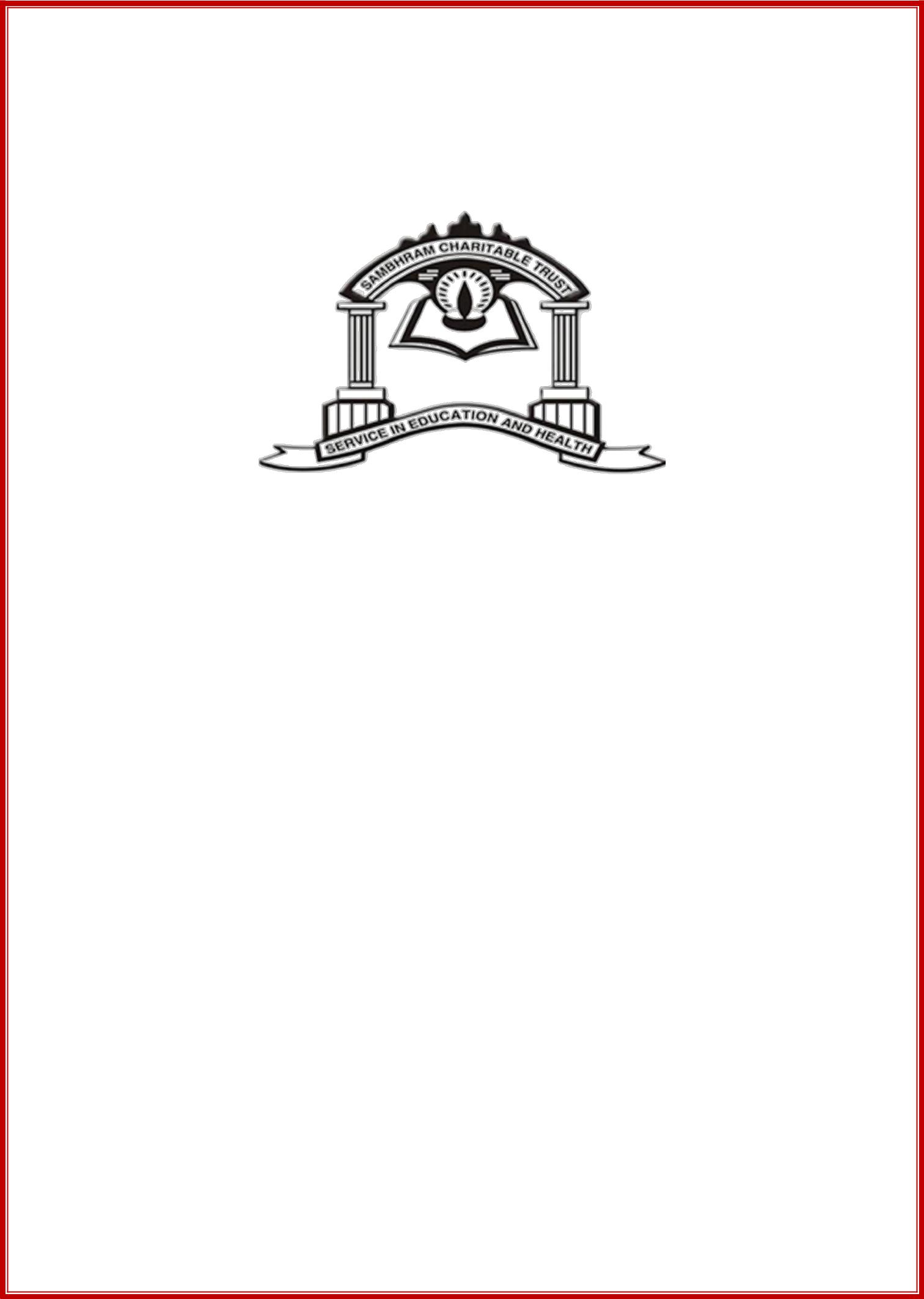
**Mrs. Divyashree N**

**Asst.Prof, Dept.of CSE**

**Department of Computer Science & Engineering**

**SAMBHRAM INSTITUTE OF TECHNOLOGY**

**BENGALURU 560097.**



**SAMBHRAM INSTITUTE OF TECHNOLOGY**

**M.S. PALYA, JALAHALLI EAST, BENGALURU - 560 097.**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**CERTIFICATE**

This is to certify that the mini-project entitled **“Museum Management System”** has been carried out by **MAYANK KUMAR SHAW (1ST17CS090)** and **KUMAR PRINCE (1ST17CS084),** bonafide students of **Sambhram Institute of Technology** in partial fulfilment of requirements for the fifth semester **Database Management System Laboratory,** prescribed by the **Visvesvaraya Technological University (VTU)**, Belagavi during the academic year **2019-2020**. It is also hereby, ensured that all corrections/suggestions indicated for internal assessment have been incorporated while submitting this report. This report has been approved as it satisfies the academic requirements with respect to the project work prescribed for the said course.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Mrs. Divyashree N**

**Asst.Professor, Dept. Of CSE**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Dr. T John Peter**

**Head of Dept. of CSE, SaIT**

**External Viva**

**Names of the examiners**

**Signatures with date**

1. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
2. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ACKNOWLEDGEMENT**

While presenting this mini-project entitled “**Museum Management System”** entrusted to us, we feel that it is our duty to acknowledge the guidance and help provided for its success.

We take this occasion to thank God, almighty for blessing us with his grace and taking our endeavour to a successful culmination.

We are grateful to our institution, **Sambhram Institute of Technology** for providing us, a congenial atmosphere to carry out this project with all possible amount of ease.

We thank **Dr. H.G Chandrakanth,** Principal, Sambhram Institue of Technology, and **Dr. T John Peter,** HOD, Department of Computer Science & Engineering, for their constant support.

We extend our thanks to our esteemed guides, **Mrs. Bilwashree H** and **Mrs. Divyashree N** Assistant Professors, Department of Computer Science &Engineering, for providing us with the right guidance and advice at the crucial junctures and for showing us the right way.

We are also indebted to our parents and friends for their continued moral and material support throughout the course of this project and helping us in showcasing the project’s presentation.

***Our thanks to all teaching, non-teaching staffs and others who have contributed bits, bytes and words to accomplish this project*.**

**MAYANK KUMAR SHAW**

**KUMAR PRINCE**

**ABSTRACT**

As the name specifies “MUSEUM MANAGEMENT SYSTEM” is a software developed for managing various activities in the museum. For the past few years the interest of people in various historic and cultural activities is increasing rapidly. Thereby the number of museums are also increasing for the accommodation of the various cultures. And hence there is a lot of strain on the person who are running the museum and software’s are not usually used in this context. This particular project deals with the problems on managing a museum and avoids the problems which occur when carried manually.

**TABLE OF CONTENTS**

**SI NO CHAPTER NAME PAGE NO**

**CHAPTER 1** **INTRODUCTION TO DATABASE** 1

|  |  |  |
| --- | --- | --- |
| 1.1 | Database system environment | 3 |
| 1.2 | Advantages of using DBMS approach | 5 |
| 1.3 | Architecture of database | 6 |
| **CHAPTER 2 USER REQUIREMENTS DEFINITION** | |  |
| 2.1 | Overview | 8 |
| 2.1.1 | Objectives | 8 |
| 2.2 | Functionalities of the application | 9 |
| 2.2.1 | User classes and characteristics | 9 |
| . 2.3 | Description of the relations | 10 |

|  |  |  |
| --- | --- | --- |
| **CHAPTER 3** | **SYSTEM REQUIREMENTS SPECIFICATION** | |
| 3.1 | Functional requirements | 15 |
| 3.1.1 | Valid assumptions and dependencies | 15 |
| 3.1.2 | Data requirement | 16 |
| 3.1.3 | External interface requirement | 16 |
| 3.1.4 | Operational requirement | 17 |
| 3.2 | Non-functional requirements | 17 |
| 3.2.1 | Performance requirement | 17 |
| 3.2.2 | Product and usability requirements | 18 |
| 3.2.3 | Requirement attributes | 18 |
| 3.3 | Software requirements specification | 19 |
| 3.3.1 | Specifications of software for app dev. | 19 |
| 3.4 | Hardware requirements specification | 19 |

|  |  |  |
| --- | --- | --- |
| **CHAPTER 4 DESIGN AND IMPLEMENTATION** | |  |
| 4.1 | ER Model and Schema Diagram | 20 |
| 4.2 | Code snippets for front end | 22 |
| 4.3 | Database code snippets | 23 |
| 4.4 | Database connectivity using PHP | 24 |

|  |  |  |
| --- | --- | --- |
| **CHAPTER 5** | **RESULT AND ANALYSIS** |  |
| 5.1 | User Front End | 28 |
| 5.2 | Back End Database | 29 |
| 5.3 | Record details | 29 |
|  |  |  |

**CONCLUSION** 30

**BIBLIOGRAPHY** 31

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIG NO** | **DESCRIPTION** | **PAGE NO** |
| 1.1 | Database System Environment | 3 |
| 1.2 | Basic Client Server Architecture | 6 |
| 4.1 | ER Diagram for Museum management | 20 |
| 4.2 | Schema Diagram for Museum management | 21 |
| 4.3 | Connection with PHP | 24 |
| 5.1 | Front page | 28 |
| 5.2 | Visitor Page | 28 |
| 5.3 | Database back end | 29 |
| 5.4 | Admin Record Details | 29 |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **TABLE NO** | **DESCRIPTION** | **PAGE NO** |
| 2.1 | Admin Records | 10 |
| 2.2 | Department Records | 10 |
| 2.3 | Staff Records | 11 |
| 2.4 | Object Records | 11 |
| 2.5 | Model Records | 12 |
| 2.6 | Statue Records | 12 |
| 2.7 | Painting Records | 13 |
| 2.8 | Museum Events Record | 13 |
| 2.9 | Store Items Record | 14 |

**CHAPTER 1**

**INTRODUCTION TO DATABASE**

A **database** is a collection of related data. By data, we mean known facts that can be recorded and that have implicit meaning. For example, consider the names, telephone numbers, and addresses of the people you know. You may have recorded this data in an indexed address book or you may have stored it on a hard drive, using a personal computer and software such as Microsoft Access or Excel. This collection of related data with an implicit meaning is a database.

The preceding definition of database is quite general; for example, we may consider the collection of words that make up this page of text to be related data and hence to constitute a database. However, the common use of the term database is usually more restricted. A database has the following implicit properties:

* A database represents some aspect of the real world, sometimes called the **miniworld** or the universe of discourse (**UOD**). Changes to the miniworld arereflected in the database.
* A database is a **logically coherent collection of data** with some inherent meaning. A random assortment of data cannot correctly be referred to as a database.
* A database is designed, built, and populated with data for a specific purpose. It has an intended group of users and some preconceived applications in

which these users are interested.

A database may be generated and maintained manually or it may be computerized. For example, a library card catalog is a database that may be created and maintained manually. A computerized database may be created and maintained either by a group of application programs written specifically for that task or by a database management system. We are only concerned with computerized databases in this report.

A database management system (DBMS) is a collection of programs that enables users to create and maintain a database. The DBMS is a general-purpose software system that facilitates the processes of defining, constructing, manipulating, and sharing databases among various users and applications.





**SaIT, CSE** **Page 1**

**MUSEUM MANAGEMENT SYSTEM**

****

* **Defining** a database involves specifying the data types, structures, andconstraints of the data to be stored in the database. The database definition or descriptive information is also stored by the DBMS in the form of a database catalog or dictionary; it is called meta-data.
* **Constructing** the database is the process of storing the data on some storagemedium that is controlled by the DBMS.
* **Manipulating** a database includes functions such as querying the database toretrieve specific data, updating the database to reflect changes in the miniworld, and generating reports from the data.
* **Sharing a** database allows multiple users and programs to access the

database simultaneously.

An application program accesses the database by sending queries or requests for data to the DBMS. A query typically causes some data to be retrieved; a transaction may cause some data to be read and some data to be written into the database.

Other important functions provided by the DBMS include protecting the database and maintaining it over a long period of time. Protection includes system protection against hardware or software malfunction (or crashes) and security protection against unauthorized or malicious access. A typical large database may have a life cycle of many years, so the DBMS must be able to maintain the database system by allowing the system to evolve as requirements change over time.

It is not absolutely necessary to use general-purpose DBMS software to implement a computerized database.

We could write our own set of programs to create and maintain the database, in effect creating our own special-purpose DBMS software. In either case—whether we use a general-purpose DBMS or not—we usually have to deploy a considerable amount of complex software. In fact, most DBMSs are very complex software systems.

To complete our initial definitions, we will call the database and DBMS software together a database system.

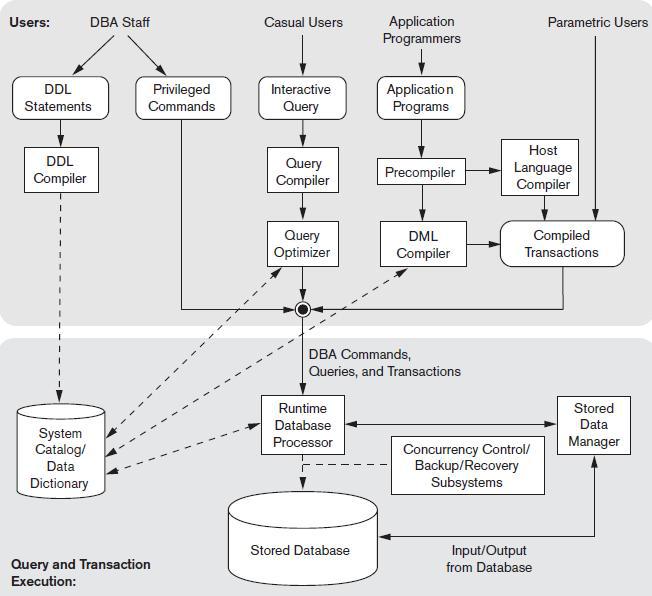


**SaIT, CSE** **Page 2**

**MUSEUM MANAGEMENT SYSTEM**

****

**1.1 DATABASE SYSTEM ENVIRONMENT**

****

**Fig 1.1: The database system environment**

The database and the DBMS catalog are usually stored on disk. Access to the disk is controlled primarily by the operating system (OS), which schedules disk read/write.

Many DBMSs have their own buffer management module to schedule disk read/write, because this has a considerable effect on performance. Reducing disk read/write improves performance considerably. A higher-level stored data manager module of the DBMS controls access to DBMS information that is stored on disk, whether it is part of the database or the catalog.

* The **DBA staff**, casual users work with interactive interfaces to formulate queries.





**SaIT, CSE** **Page 3**

**MUSEUM MANAGEMENT SYSTEM**

****

* **Application programmers** create programs using some host programminglanguages, and parametric users do data entry work by supplying parameters to predefined transactions. The DBA staff works on defining the database and tuning it by making changes to its definition using the DDL and other privileged commands.
* The **DDL compiler** processes schema definitions, specified in the DDL, and stores descriptions of the schemas (meta-data) in the DBMS catalog. The catalog includes information such as the names and sizes of files, names and data types of data items, storage details of each file, mapping information among schemas, and constraints. In addition, the catalog stores many other types of information that are needed by the DBMS modules, which can then look up the catalog information as needed.
* **Casual users** and persons with occasional need for information from thedatabase interact using some form of interface, which we call the interactive query interface. These queries are parsed and validated for correctness of the query syntax, the names of files and data elements, and so on by a query compiler that compiles them into an internal form. This internal query is subjected to query optimization.
* The **query optimizer** is concerned with the rearrangement and possible reordering of operations, elimination of redundancies, and use of correct algorithms and indexes during execution. It consults the **system catalog** for statistical and other physical information about the stored data and generates executable code that performs the necessary operations for the query and makes calls on the runtime processor.
* The **runtime database processor** executes (1) the privileged commands, (2) the executable query plans, and (3) the canned transactions with runtime parameters. It works with the system catalog and may update it with statistics. It also works with the stored data manager, which in turn uses basic

operating system services for carrying out low-level input/output (read/write) operations between the disk and main memory. It is now common to have the client program that accesses the DBMS running on a separate computer from the computer on which the database resides.





**SaIT, CSE** **Page 4**

**MUSEUM MANAGEMENT SYSTEM**

****

* The former is called the client computer running a DBMS client software and the latter is called the database server. In some cases, the client accesses a middle computer, called the application server, which in turn accesses the database server.

**1.2 ADVANTAGES OF USING THE DBMS APPROACH**

Due to its centralized nature, the database system can overcome the disadvantages of the file system-based system.

* **Data independency**: Application program should not be exposed to detailsof data representation and storage DBMS provides the abstract view that hides these details.
* **Efficient data access**: DBMS utilizes a variety of sophisticated techniquesto store and retrieve data efficiently.
* **Data integrity and security:** Data is accessed through DBMS, it can

enforce integrity constraints. E.g.: Inserting salary information for an employee.

* **Data Administration**: When users share data, centralizing the data is animportant task, Experience professionals can minimize data redundancy and perform fine tuning which reduces retrieval time.
* **Concurrent access and Crash recovery**: DBMS schedules concurrentaccess to the data. DBMS protects user from the effects of system failure.
* **Reduced application development time**: DBMS supports importantfunctions that are common to many applications.



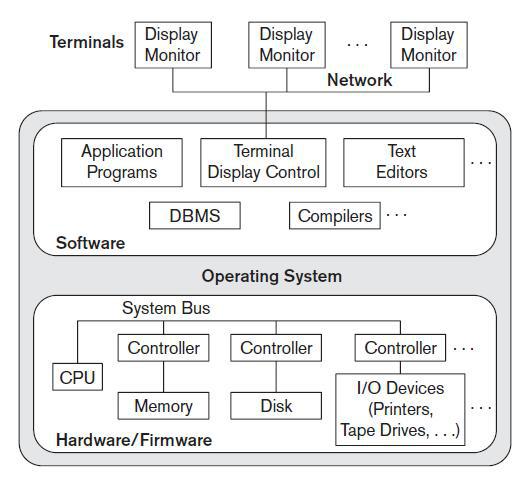


**SaIT, CSE** **Page 5**

**MUSEUM MANAGEMENT SYSTEM**

****

**1.3 ARCHITECTURE OF DATABASE**

****

**Fig1.2: A physical, centralized and a basic Client - Server architecture**

The idea is to define **specialized servers** with specific functionalities. For example, it is possible to connect a number of PCs or small workstations as clients to a **file server** that maintains the files of the client machines. Another machine can be designated as a **printer server** by being connected to various printers; all print requests by the clients are forwarded to this machine.

Thus, the resources provided by specialized servers can be accessed by many client machines. The **client machines** provide the user with the appropriate interfaces to utilize these servers, as well as with local processing power to run local applications. This concept can be carried over to other software packages, with specialized programs such as a CAD (computer-aided design) package being stored on specific server machines and being made accessible to multiple clients.





**SaIT, CSE** **Page 6**

**MUSEUM MANAGEMENT SYSTEM**

****

A **client** in this framework is typically a user machine that provides user interface capabilities and local processing. When a client requires access to additional functionality such as database access that does not exist at that machine, it connects to a server that provides the needed functionality.

A **server** is a system containing both hardware and software that can provide services to the client machines, such as file access, printing, archiving, or database access.





**SaIT, CSE** **Page 7**

**CHAPTER 2**

**USER REQUIREMENTS DEFINITION**

**2.1 OVERVIEW**

* MUSEUM MANAGEMENT SYSTEM is a software application which automatically performs the regulatory functions of a museum such as maintaining the records related to antiques, users(employee), departments, paintings, statues, antique detail info, museum events and all necessary requirements for it to manage day to day operations as described in the abstract section earlier.
* The database follows a typical event flow seen in such a system. The database mainly is from an administrator’s perspective, but it also allows the employees to manage their self information along with the visitors to access at a basic level of the system which provides them detailed information about the contents of the museum.
* Since a museum is the center for all the antiques stored, all the records in the database resolve around insert, display, remove and update activities.

Some of the other independent categories may include users(employee) details, event’s details, object status…etc.

**2.1.1 OBJECTIVES**

* To automate the existing system by relating all of the records of the above mentioned operations in a logical manner such that the recorded transactions can be replaced and accepted without major changes and problems.
* To eliminate paper work by using a computerized system.
* To save time and money.
* To trap most of the data entry errors.



**SaIT, CSE** **Page 8**

**MUSEUM MANAGEMENT SYSTEM**

****

**2.2 FUNCTIONALITIES OF THE APPLICATION**

* Provision of quick access to the records maintained.
* Showing important details so that important decisions could be made easily.
* Generation of customized reports based on particular criteria. (For example: Retrieval may take place based on the name of the department, type of object stored, availability of the items… etc.)
* Provision of easy - to - use or a user friendly interface, with menus and forms for clear navigability.
* Data validation: minimize data redundancy. (unnecessary duplicate data are prevented from creeping into the database.)
* Provision of separate access to individual details and administrator such that only the administrator gets the rights to add, delete and modify the various services offered by the system.
* Provide the administrator with options to add as many as objects, events, employees corresponding to the subject or the content required.
* Providing the visitors to get minimal access to the information stored in the museum to provide them guidance about the details of the museum.
* Providing the employees to manage their general details.
* Provide effective and efficient querying techniques.

**2.2.1 USER CLASSES AND CHARACTERISTICS**

Any user with basic computer skills can make use of the application. The user should have only been briefed about the functionality of the system before he/she can start using the system.



**SaIT, CSE** **Page 9**

**MUSEUM MANAGEMENT SYSTEM**

****

**2.3 DESCRIPTION OF THE RELATIONS (TABLES)**

**(I)Admin**: This table is used to store details of the Admin working in the museum.

|  |  |  |
| --- | --- | --- |
| **Field name** | **Data type** | **Description** |
| **ad-id** | **Varchar(10)** | **Admin id** |
| **ad-name** | **Varchar(20)** | **Admin name** |
| **phone** | **Varchar(10)** | **Phone no. of admin** |
| **Gender** | **Varchar(1)** | **Admin gender** |
| **Email Id** | **varchar(30)** | **Email Id of Admin** |
| **Password** | **varchar(50)** | **Password of Admin** |
| **DOB** | **date** | **Admin Birth date** |

**Table 2.1: Admin records**

**(II) Department**: This table is used to store the details of the department.

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Type** | **Description** |
| **Did** | **Varchar(10)** | **Department id** |
| **Dname** | **Varchar(20)** | **Department name** |
| **Floor** | **Varchar(20)** | **Department Floor** |

**Table 2.2: Department Record**

****

**SaIT, CSE** **Page 10**

**MUSEUM MANAGEMENT SYSTEM**

****

**(III)Staff:** This table contains the details of the various employees working in the museum.

|  |  |  |
| --- | --- | --- |
| **Name** | **Datatype** | **Description** |
| **staffid** | **Varchar(10)** | **Employee ID** |
| **Did** | **Varchar(10)** | **Department ID** |
| **Name** | **Varchar(20)** | **Employee name** |
| **email** | **Varchar(40)** | **Employee email** |
| **password** | **Varchar(50)** | **Employee password** |
| **country** | **Varchar(20)** | **Residing country** |
| **phone** | **Varchar(10)** | **Employee phone** |
| **DOB** | **date** | **Employee birth date** |
| **gender** | **Varchar(1)** | **Employee gender** |
| **job** | **Varchar(20)** | **Job description** |
| **entry-time** | **datetime** | **Entry time record** |
| **exit-time** | **datetime** | **Exit time record** |

**Table 2.3: Staff Records**

**(IV) Object:** This table contains the details of all the antique objects stored in museum.

|  |  |  |
| --- | --- | --- |
| **Name** | **Data type** | **description** |
| **id** | **Varchar(10)** | **object ID** |
| **Did** | **Varchar(10)** | **Residing department ID** |
| **artist-name** | **Integer** | **Artist name** |
| **year** | **integer** | **Year of origin** |
| **title** | **varchar(20)** | **Object name** |
| **type** | **varchar(20)** | **Object type** |

**Table 2.4: Object records**

****

**SaIT, CSE Page 11**

**MUSEUM MANAGEMENT SYSTEM**

****

**(V) Model**: This table contains the details of various models.

|  |  |  |
| --- | --- | --- |
| **Name** | **Data type** | **description** |
| **Id** | **Varchar(10)** | **object ID** |
| **Did** | **Varchar(10)** | **Department ID** |
| **artist-name** | **Varchar(20)** | **Artist name** |
| **year** | **integer** | **Year of origin** |
| **title** | **varchar(20)** | **Object title** |
| **type** | **Varchar(10)** | **Object type** |
| **size** | **Varchar(10)** | **Object size** |

**Table 2.5: Model records**

**(VI) Statue**: This table contains the details of various statues.

|  |  |  |
| --- | --- | --- |
| **Name** | **Data type** | **description** |
| **id** | **Varchar(10)** | **object ID** |
| **Did** | **Varchar(10)** | **Residing department ID** |
| **artist-name** | **Integer** | **Artist name** |
| **year** | **integer** | **Year of origin** |
| **title** | **varchar(20)** | **Object name** |
| **type** | **varchar(20)** | **Object type** |
| **material** | **varchar(20)** | **Material type** |
| **style** | **varchar(20)** | **Statue style** |
| **height-in-m** | **integer** | **Height** |
| **weight-in-kg** | **weight** | **Weight** |

**Table 2.6: Statue records**

****

**SaIT, CSE Page 12**

**MUSEUM MANAGEMENT SYSTEM**

****

**(VII) Painting**: This table contains the details of various paintings.

|  |  |  |
| --- | --- | --- |
| **Name** | **Data type** | **description** |
| **id** | **Varchar(10)** | **object ID** |
| **Did** | **Varchar(10)** | **Residing department ID** |
| **artist-name** | **Integer** | **Artist name** |
| **year** | **integer** | **Year of origin** |
| **title** | **varchar(20)** | **Object name** |
| **type** | **varchar(20)** | **Object type** |
| **material-type** | **varchar(20)** | **Material type** |
| **style** | **varchar(20)** | **Statue style** |

**Table 2.7: Painting records**

**(VIII) Museum Events**: This table is used to store the details of all museum events.

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Type** | **Description** |
| **event-name** | **Varchar(20)** | **Event name** |
| **Did** | **Varchar(10)** | **Department ID** |
| **date-start** | **date** | **Event start date** |
| **date-end** | **date** | **Event end date** |
| **people-involved** | **Varchar(20)** | **People involved** |

**Table 2.8 Museum Events Record**

****

**SaIT, CSE Page 13**

**MUSEUM MANAGEMENT SYSTEM**

****

**(IX) Store Items**: This table is used to store the details of all store items.

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Type** | **Description** |
| **item-name** | **Varchar(20)** | **Item name** |
| **item-id** | **Varchar(10)** | **Item ID** |
| **price** | **integer** | **Item price** |

**Table 2.9: Store Items Record**

****

**SaIT, CSE Page 14**

**CHAPTER 3**

**SYSTEM REQUIREMENTS SPECIFICATION**

**3.1 FUNCTIONAL REQUIREMENTS**

Functional requirements are statements of services the system should provide, how the system should react to particular inputs, and how the system should behave in particular situations. In some cases, the functional requirements may also explicitly state what the system should not do.

**3.1.1 Valid assumptions and dependencies**

* The **assumptions** are:
  + The source code pertaining to each module developed should be error free.
  + The system should be user friendly.
  + The information, modifications and updates to the database should be available to all the administrators of the museum.
  + The system should have feasible storage capacity and should provide faster access to the data.
  + The system should provide retrieval facility and also facilities for quick and understandable transactions.
  + Users should be registered by admin in order to procure administrative membership.
  + Valid credentials shall only be keyed-in without which, authentication becomes unsuccessful.
* The **dependencies** are:
  + The specific hardware and software through which the product operates.
  + Thus on the basis of the list of the requirements specification, the application would be developed to a fully functional one.
  + The end-users have to possess proper knowledge and understanding in order to work with it.
  + The system should have a general report generator.
  + All updates pertaining to all modules are to be recorded and changes should happen in the referenced modules concurrently.



**SaIT, CSE** **Page 15**

**MUSEUM MANAGEMENT SYSTEM**

****

**3.1.2 Data Requirement**

The input consists of query to the database and the output consists of solutions for the query. The output also includes the user receiving the details of their accounts.

In this project, the inputs will be the queries as fired by the users like creating an account, adding information on the basis of various criteria, updating the information pertaining to individual records, deleting individual records, searching and retrieving records, etc.

Now, the solutions for the queries as the respective outputs will be visible when the users request the server through the GUI.

**3.1.3 External Interface Requirement**

**GUI**

The application that is developed provides a good graphical user interface for the user or the administrator who operates the system, performs the required tasks such as insert, update, delete and search on the basis of various criteria.

**General features of the GUI**

* It allows the user to generate quick reports and export data to different formats.
* Provides stock verification and search facility based on different criteria.
* All the modules designed are collectively integrated to form the Menu Driven Interface (MDI).
* The design of the interface is simple and all the modules within it, follow a standard template.
* The user interface must be able to interact with the user management module and a part of the interface must be dedicated to the login/logout module.
* **Login interface or the User Authentication Window**

In case the user is not yet registered, registration can be done through this window. Once the registration is successful, authenticity can be established through the user authentication window which on successful authentication redirects the user to the menu driven interface.



**SaIT, CSE** **Page 16**

**MUSEUM MANAGEMENT SYSTEM**

****

* **Menu - oriented views**

The menu may further show the categories based on which the whole system is supposed to be administered.

* **Control Panel**

This panel will allow the user to update/delete and refresh the contents or records based on a particular criterion.

**3.1.4 Operational Requirement**

The product will operate in Windows environment. The only requirement to use this product would be the installation of supporting software packages like MYSQL Server, APACHE server a greater version, XAMPP CONTROL PANEL VERSION 3.2.2, for viewing the pdf using Microsoft word.

**3.2 NON - FUNCTIONAL REQUIREMENTS**

These are constraints on the services or functions offered by the system. They include timing constraints, constraints on the development process, and constraints imposed by standards. Non-functional requirements often apply to the system as a whole, rather than individual system features or services.

**3.2.1 Performance requirement**

The proposed system that we are going to develop may be used as a chief performance system within the different campuses of the museum, through secured interaction with the museum staffs.

Therefore, it is expected that the database would perform functionally, all the requirements that are specified by the museum.

* The performance of the system should be fast and accurate.



**SaIT, CSE** **Page 17**

**MUSEUM MANAGEMENT SYSTEM**

****

* The MUSEUM MANAGEMENT SYSTEM shall handle expected and non-expected errors in ways that prevent loss in information and long downtime period. Thus it should have inbuilt error testing to identify and handle exceptions.
* The system should be able to handle large amount of data.

**3.2.2 Product and usability requirements**

* **Availability**: The system is available 100% for the user and can be used on a24 x 7 basis.
* **Accuracy**: The system shall accurately provide real time information takinginto consideration various concurrency issues.
* **Reliability**: The system has to be 100% reliable due to the importance ofdata and the damages that can be caused by incorrect or incomplete data.
* **Maintainability**: Changes (addition of new members, database changes)must be verified once per day at least.
* **Portability:** The system should also be portable.

**3.2.3 Requirements attributes**

* There may be multiple administrators creating the project, so, all of them would have rights to create changes to the system.
* The project should be open source.
* The quality of the database is maintained in such a way so that it can be very friendly to all the users of the database.
* The users should be able to easily install the software on the system.



**SaIT, CSE** **Page 18**

**MUSEUM MANAGEMENT SYSTEM**

****

**3.3 SOFTWARE REQUIREMENTS SPECIFICATION**

The application is developed using PHP as the front end which is supported by Apache server and MySQL Server as the back end for accessing and connecting the front end to the database.

PHP is a powerful but simple language aimed primarily at developers creating applications by using the Apache server, inherits many of the best features of HTML and php, but few of the inconsistencies and anachronisms, resulting in a cleaner and more logical language.

SQL (Structured Query Language) is used for defining, manipulating, controlling, storing and viewing the information present in a database.

**3.3.1 Specifications of the software used for application development**

The following software are required to develop the application:

* **Front end: HTML**
* **Back end: MYSQL**
* **Database used:** MS SQL Server Database.
* **Database file used:** MS SQL Server Database file (.mdf)
* XAMPP CONTROL PANNEL V3.2.2
* **Operating system:** Windows 10 (x64),

**3.4 HARDWARE REQUIREMENTS SPECIFICATION**

The following hardware requirements are needed to develop the application:

* Computer that has a 1.6 GHz or faster processor (2 GHz recommended).
* 1 GB (32-bit) or 2 GB (64-bit) RAM (add 512 MB if running in a virtual machine).
* 10 GB of available hard disk space.
* 5400 RPM hard disk drive.

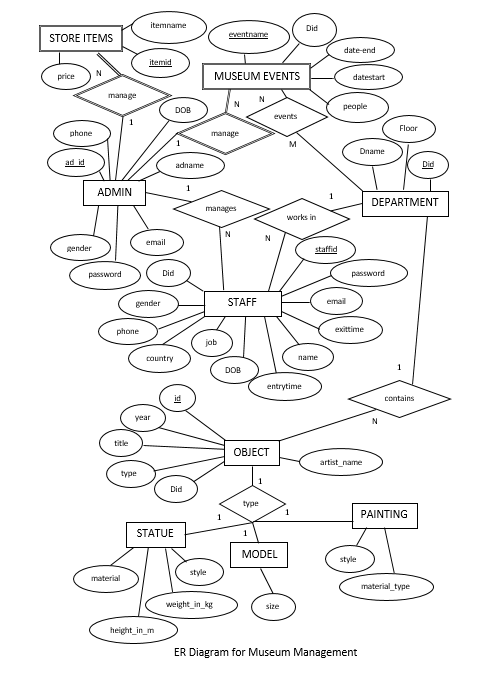


**SaIT, CSE** **Page 19**

**CHAPTER 4**

**DESIGN AND IMPLEMENTATION**

**ER MODEL**



**Fig 4.1: Logical ER diagram for museum management system**

****

**SaIT, CSE Page 20**

**MUSEUM MANAGEMENT SYSTEM**

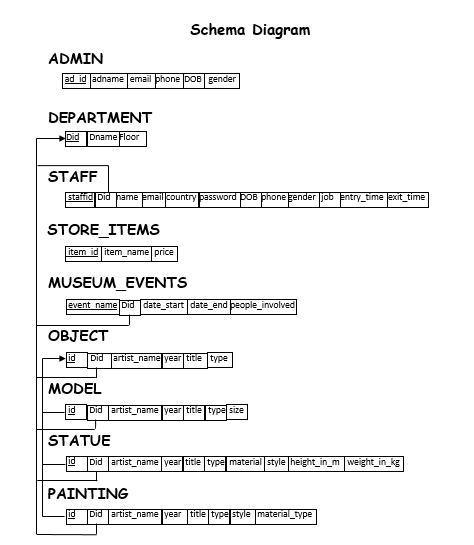
****

The ER model describes data as *entities*, *relationships*, and *attributes*. The basic object that the ER model represents is an **entity**, which is a *thing* in the real world with an independent existence.

Here each entity has one or more attributes. Fig shows the entities with their respective attributes, relationships, with cardinality ratios which further mean something about the quality of participation that may be offered by the entities for their relationships between various other entities.

**SCHEMA DIAGRAM**

The schema diagram describes the inter-relation between the participating attributes of the tables and provides a clear view of the understanding of the connections and functioning of the database structure.

****

**Fig 4.2: Schema Diagram**



**SaIT, CSE Page 21**

**MUSEUM MANAGEMENT SYSTEM**

****

**4.2 CODES FOR FRONTEND**

<?php

include('login.php');

if(isset($\_SESSION['login\_id'])){

header("loation: admin\_menu.php");

}

?>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8" />

<meta http-equiv="x-ua-compatible" content="ie=edge" />

<meta name="viewport" content="width=device-width, initial-scale=1" />

<title>Museum login form</title>

<link href="css/index\_style.css" rel="stylesheet" type="text/css"/>

<link rel="stylesheet" href="fonts/fontawesome-free-5.10.2-web/css/all.min.css"/> <h1 style="font-family:courier; color: black; font-size: 6vh">MUSEUM MANAGEMENT SYSTEM</h1>

</head>

<body class="floor-fade" style="background:url('images/gall.jpg') no-repeat; background-size:cover;">

<header >

<div class="wrapper">

<div class="logo">

<img src="images/images.png" alt="">

</div>

<ul class="nav-area" style="font-family:courier; color:black">

<li><a href="about.php">About</a></li>

<li><a href="visitor\_page.php">Visitor</a></li>

<li><a href="emp\_login.php">Employee</a></li>

<li><a href="index.php">Admin</a></li>

</ul>

</div>

</header>

<div class="login\_box">

<form method="POST" action="">

<h1>Login</h1>

<div class="textbox">

<i class="fas fa-user" aria-hidden="true"></i>

<input type="text" placeholder="Enter Admin ID" name="ad\_id" value="" />

</div>

<div class="textbox">

<i class="fas fa-unlock" aria-hidden="true"></i>

<input type="password" placeholder="Password" name="pass\_word" value="" />

</div>

<a href="">



**SaIT, CSE Page 22**

**MUSEUM MANAGEMENT SYSTEM**

****

<input class="btn" type="submit" name="submit" value="Sign In" />

</a>

<a href="admin\_forgot\_password.php" style="color: white">Forgot Password?</a>

<br/>

<span><?php echo $error; ?></span>

</form>

</div>

</body>

</html>

**4.3: DATABASE CODE SNIPPETS**

<?php

session\_start(); //starting session

$error = ''; //variable to show error message

if(isset($\_POST['submit'])){

if(empty($\_POST['ad\_id']) || empty($\_POST['pass\_word'])){

$error = "ID or Password is invalid";

}

else{

//define admin id and password

$ad\_id = $\_POST['ad\_id'];

$pass\_word = $\_POST['pass\_word'];

$conn = mysqli\_connect("localhost", "root", "", "museumsys");

//query to fetch info of registered users

$query = "SELECT ad\_id, pass\_word FROM admin WHERE ad\_id = ? AND pass\_word = ? LIMIT 1";

//to protect sql injection for security

$stmt = $conn->prepare($query);

$stmt->bind\_param("ss", $ad\_id, $pass\_word);

$stmt->execute();

$stmt->bind\_result($ad\_id, $pass\_word);

$stmt->store\_result();

if($stmt->fetch()){

$\_SESSION['login\_id'] = $ad\_id;

header("location: admin\_menu.php");

}

else{

$error = "UserId or Password is invalid";

}

mysqli\_close($conn);

}

}

?>

****

**SaIT, CSE Page 23**

**MUSEUM MANAGEMENT SYSTEM**

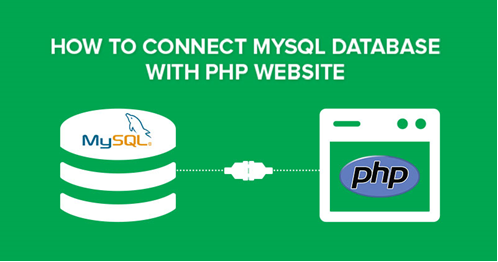
****

**4.4 DATABASE CONNECTIVITY USING PHP**

In order to establish connection to the database, the following directives are needed. They are:

In the first instalment of this MYSQL series, I introduced database and database management system. I also presented a brief overview of a popular DBMS, M





**Fig 4.3: CONNECT MYSQL DATABASE WITH PHP**

**Step1: CHANGING ADMIN PASSWORD**

First, let me tell you what PHPMyAdmin is. It is a control panel from where you can manage your database that you have created. Open your browser and go to localhost/PHPMyAdmin or click “Admin” in XAMPP UI.

When you first installed XAMPP, it only created the username for it to be accessed, you now have to add a password to it by yourself. For this, you have to go to User account where the user is same as the one shown in this picture:

change-password

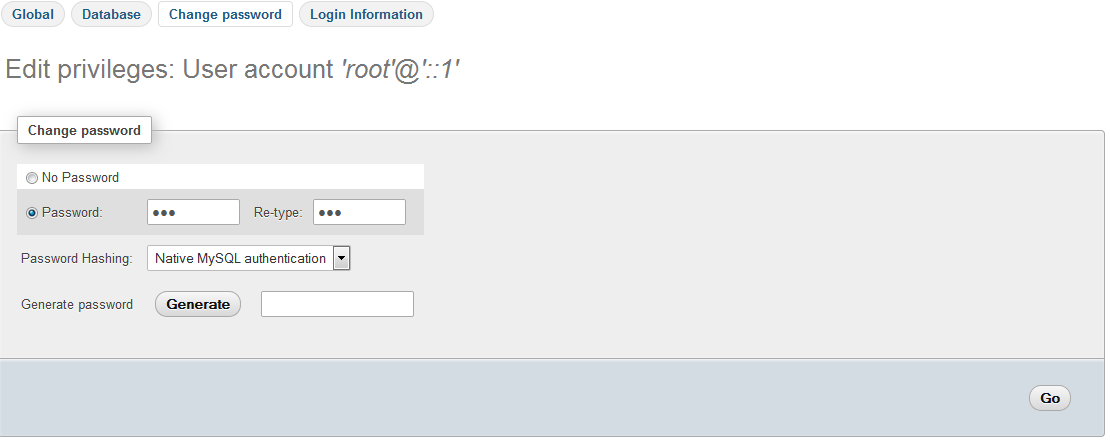
Now click edit privileges and go to change password, type your password there and save it. Remember this password as it will be use to connect to your database.

For our mini project we made our database open source using XAMPP.



**SaiT, CSE** **Page 24**

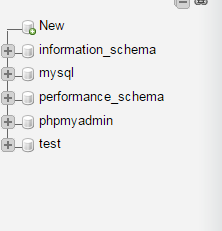
 **MUSEUM MANAGEMENT SYSTEM**



Note: It is not necessary to change password to access database on local host. It is a good practice and that is why we have used a password.

**Step 2: CREATE DATABASE**

Now return to homepage pf phpmyadmin. Click New button to create a new database.



In the new window, name your database as per your need, I am naming it “practice”. Now select collation as utf8\_general \_ci, as we are using it for learning purpose and it will handle all of our queries and data that will be covered in this tutorial series. Now click on create and your database will be created.

.

**SaIT, CSE Page 25**

**MUSEUM MANAGEMENT SYSTEM**





The newly created database will be empty now, as there are no tables in it. I will be covering that in the upcoming series where we will learn how to create tables and insert data in it. In this tutorial, we are going to connect this database to a localhost using PHP.

**STEPS 3: CREATE A FOLDER IN HTdocs**

Now, locate the folder where you installed XAMPP and htdocs folder (usually c:/xampp). Create a new folder inside c:/xampp/htdocs/ and name it. we will place web files in this folder. Why we have created folder in htdocs? XAMPP uses folders in htdocs to execute and run your PHP sites.

Note: If your using WAMP, then add your practice folder in c:/wamp/www folder.

**STEP 4: CREATE DATABASE CONNECTION FILE IN PHP**

Create a new php file and name it some name db\_connection.php and save it. Why am I creating a separate database connection file? Because if you created multiple files in which you want to insert data or select data from the database, you don’t need to write the code for database connection every time. You just have to include it by using PHP custom function **include** on the top of your code and call its function and use it. It also helps when you are moving your project location from one PC to another and you have to change the values on the single file and all the changes will be applied to all other files automatically. Write the following code in your database file.

**STEP 5: CREATE NEW PHP FILE TO CHECK YOUR DATABASE CONNECTION**

Create a new php file to connect to your database. Name it index.php and add this code in this file.

**STEP 6:** **RUN IT!**

Now open your browser and go to localhost/practice/index.php and you should see the message ‘**CONNECTED SUCCESSFULLY’ in** this screen.

**SaIT, CSE** **Page 26**

**CHAPTER 5**

**RESULT AND ANALYSIS**

The “MUSEUM MANAGEMENT SYSTEM” has been computed successfully and was tested successfully by taking into consideration the “test cases”.

It is user friendly and has required options, which can be utilized by the user to perform the desired operation.

The software or the so called application was developed using PHP, HTML as front end and phpMyAdmin SQL server as the backend in the Windows 10 environment. It is also hereby ensured that the goals are met by the software. The goals are :-

* Optimum utilization of resources.
* Efficient management of records.
* Simplification of records.
* Less processing time & quick retrieval of records.
* Characteristics of being portable and flexible for further enhancement.

Over all, project of ours being developed to help the employee and staff members of Company management to maintain them in an optimal way possible & also minimize human efforts.





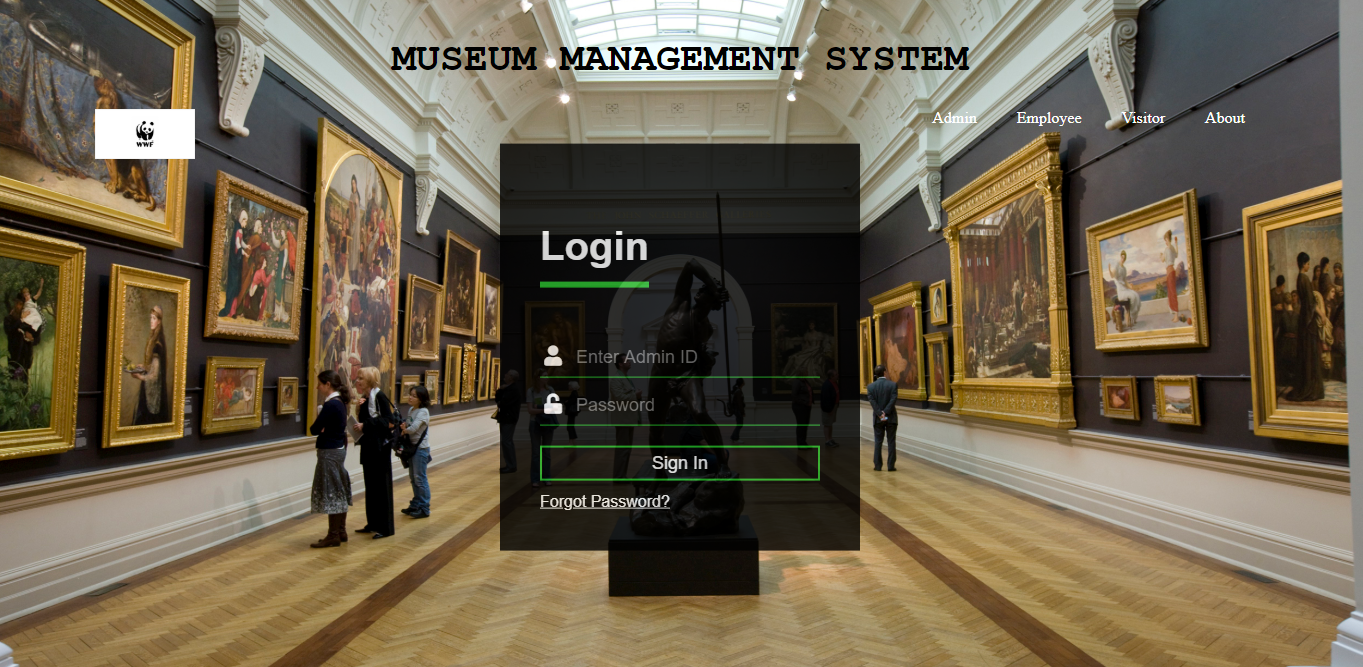


**SaIT, CSE Page 27**

**MUSEUM MANAGEMENT SYSTEM**

****

**5.1: USER FRONT END**



**Fig 5.1: Front page**

The user front end as designed for this project is shown above.

🡪 This page is the initial login page or ‘index.php’ that is first page executed by the server in order to start the museum system.

🡪 Here the admin is able to login with his registered Admin ID and Password which would allow him to go into the next page of operations if logged in successfully.

🡪 Similarly, the employees can also login in into the system by using their registered ID and passwords.

🡪 This system also allows for visitors to visit the system to get general museum details via a specialized open menu just for them.



**Fig 5.2: visitor page**

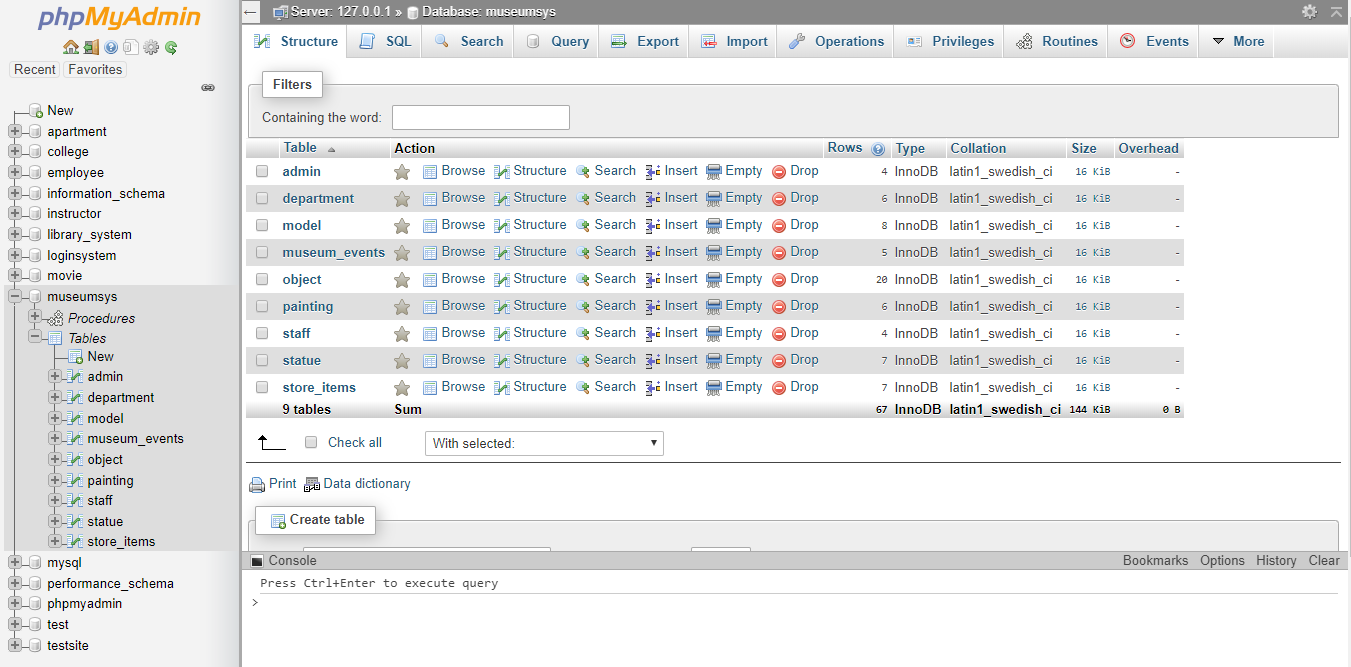


**SaIT, CSE Page 28**

**MUSEUM MANAGEMENT SYSTEM**

****

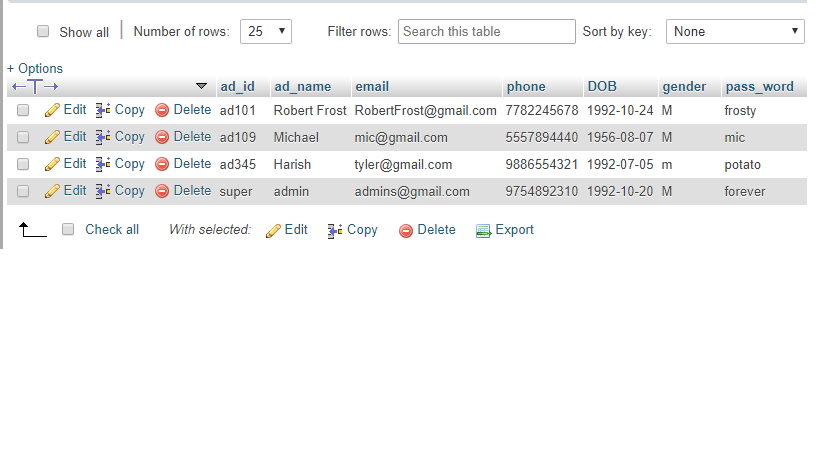
**5.2: BACK END DATABASE**



**Fig 5.3: database back end**

This is the database structure used for making this project. It consists of 9 tables each suitable for their own purpose.

**5.3: RECORD DETAILS**



**Fig 5.4: Admin record details**

* The given figure above shows the format in which the details are stored.
* The record system uses a super ID that allows for the super user operations preventing complete removal of the records.

..

**SaIT, CSE Page 29**

**CONCLUSION**

The MUSEUM MANAGEMENT SYSTEM is required for efficient working of the museum systems and allows for reducing cost, data redundancy and maintaining privacy of policy as specified by the requested museum.

The system provides attractive and user friendly approach for managing the quick essential operations of insertion, deletion, update and ordering of the records stored in the database, which is of the utmost importance.

The MUSEUM MANAGEMENT SYSTEM thus provides the required necessities that would allow for efficient use of the system.

..

**SaIT, CSE Page 30**

**BIBLIOGRAPHY**

The following project was made by the help of below mentioned sources--

[1] Learning PHP, MySQL, JavaScript, CSS & HTML5 by Robin Nixon

[2] Online sources like Google, Wikipedia, TutorialsPoint, GeeksForGeeks and StackOverflow.

[3] <https://www.tutorialrepublic.com/php-tutorial/php-mysql-crud-application.php> for php basics

[4] <https://www.w3schools.com/Css/> for CSS basics and tutorials

[5] <https://www.quora.com> for troubleshooting.

.

**SaIT, CSE Page 31**