CHAPTER 1

INTRODUCTION

1.1 PROJECT OVERVIEW

The project titled "**PRAGATHI**" aims at developing a system which provides an efficient way of managing the activities of Kudumbhashree in different Gramapanchayath .The existing system is a complete manual system since it has no official site or web applications .There are five main modules in this system .One is administrator module which view entire activities and grand permission for them .Second module is Community Development Society, CDS module which includes verification of loan applications and awareness classes. The third module is Area Development Society which includes marking of attendance and manage loan repayment . The fourth module is the Neighbour Hood Group, NHG module which includes registration of members and apply for loan and jobs offered by users . The fifth module is the users, they can give small job opportunities for members in each unit.

1.2 PROJECT SPECIFICATION

The "PRAGATHI" is a good web application that satisfies the requirements of different clients.The scope of this project is limited with in a panchayath.It designed in userfriendly and the duties of the panchayath members regarding a particular panchayath are done efficiently and added more funtionality in depth..

The system includes 5 modules. They are:

1. Admin Module

Admin must have a login into this system. He has the overall control of the system. Admin can add cds members. Admin can View details of exhibitions, cds, ads and unit members.

2. CDS Module

CDS can add ads and exhibition details. View details of members.

3. ADS Module

ADS can add unit members. View exhibition details.

4. Unit Member

View exhibition details.

5.Users

Users can register and view exhibition information.

CHAPTER 2

SYSTEM STUDY

2.1 INTRODUCTION

System analysis is a process of gathering and interpreting facts, diagnosing problems and the information to recommend improvements on the system. It is a problem solving activity that requires intensive communication between the system users and system developers. System analysis or study is an important phase of any system development process. The system is studied to the minute's detail and analyzed. The system analyst plays the role of the interrogator and dwells deep into the working of the present system. The system is viewed as a whole and the input to the system are identified. The outputs from the organizations are traced to the various processes. System analysis is concerned with becoming aware of the problem, identifying the relevant and decisional variables, analyzing and synthesizing the various factors and determining an optimal or at least a satisfactory solution or program of action.

A detailed study of the process must be made by various techniques like interviews, questionnaires etc. The data collected by these sources must be scrutinized to arrive to a conclusion. The conclusion is an understanding of how the system functions. This system is called the existing system. Now the existing system is subjected to close study and problem areas are identified. The designer now functions as a problem solver and tries to sort out the difficulties that the enterprise faces. The solutions are given as proposals. The proposal is then weighed with the existing system analytically and the best one is selected. The proposal is presented to the user for an endorsement by the user. The proposal is reviewed on user request and suitable changes are made. This is loop that ends as soon as the user is satisfied with proposal.

Preliminary study is the process of gathering and interpreting facts, using the information for further studies on the system. Preliminary study is problem solving activity that requires intensive communication between the system users and system developers. It does various feasibility studies. In these studies, a rough figure of the system activities can be obtained, from which the decision about the strategies to be followed for effective system study and analysis can be taken.

2.2 EXISTING SYSTEM

The existing system is a complete manual system since it has no official site or web application. There are many paper works to do for the operations like financial data, the members registration and so on.

The system does not support modern technology and more time consuming. Peoples like ADS and CDS to act as intermediates at each level and there may arise problems due to their intervention.

The existing system is a manual system. It is loaded with discrepancies. All these occur due to the pen and paper work done throughout the system. Manual operations derive inefficient process and information with average accuracy.

2.3 DRAWBACKS OF EXISTING SYSTEM

- Since there is no particular site, all the activities are managed through paper works.
- The chances of error in calculation (Thrift & Credit) may occur because of the manual intervention.
- A person (ADS) wanted to act as a mediator between the administrator and the user. This causes the loss of time and need a lot of human efforts.
- The data of each user is not secure**.**
- New changes cannot be easily implemented.

2.4 PROPOSED SYSTEM

The proposed system named ”PRAGATHI” is an integrated web application that handles various actions and activities of KudumbaShree units. Through links that displays in the home page the user can access different services of the website. The proposed system is for managing the activities of various needs of the panchayat, each unit and the members. The system provides the information about each units and financial transaction through internet. Each user can get their details and can make changes with their own username and password.

Every KudumbaShree units consist of one ADS who act an intermediate between the panchayat and the unit. The main feature of this system is that the panchayat can easily communicate with the head of each unit directly, by avoiding the ADS intervention.

2.5 ADVANTAGES OF PROPOSED SYSTEM

The system is very simple in design and to implement. The system requires very low system resources and the system will work in almost all configurations. It has got following features:

- **Userfriendly:-**

The system developed can be used effectively to establish effective communication among
The members of each unit as well as the panchayat.

- **Better security: -**

For data to remain secure measures must be taken to prevent unauthorized access. Security means that data are protected from various forms of destruction. The system security problem can be divided into four related issues: security, integrity, privacy and confidentiality. Username and password requirement to sign in ensures security. It will also provide data security as we are using the secured databases for maintaining the documents.

- **Ensure data accuracy: -**

The proposed system eliminates the manual errors while entering the details of the users during the registration.

- **Better service: -**

The product will avoid the burden of hard copy storage. We can also conserve the time and human resources for doing the same task. The data can be maintained for longer period with no loss of data.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1FEASIBILITY STUDY

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that spend on it. Feasibility study lets the developer foresee the future of the project and the usefulness. A feasibility study of a system proposal is according to its workability, which is the impact on the organization, ability to meet their user needs and effective use of resources. Thus, when a new application is proposed it normally goes through a feasibility study before it is approved for development.

The document provides the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as Technical, Economic and Operational feasibilities. The following are its features: -

3.1.1 Economical Feasibility

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors, which affect the development of a new system, is the cost it would require.

The following are some of the important financial questions asked during preliminary investigation:

- ➢ The costs conduct a full system investigation.
- ➢ The cost of the hardware and software.
- ➢ The benefits in the form of reduced costs or fewer costly errors.

The proposed system is developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication of the system is economically possible for development.

The cost of project, PRAGATHI was divided according to the system used, its development cost and cost for hosting the project According to all the calculations the project was developed in a low cost. As it is completely developed using open source software the only cost was spent for hosting the project which is affordable.

3.1.2 Technical Feasibility

The system must be evaluated from the technical point of view first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of input, output, programs and procedures. Having identified an outline system, the investigation must go on to suggest the type of equipment, required method developing the system, of running the system once it has been designed.

Technical issues raised during the investigation are:

- Does the existing technology sufficient for the suggested one?
- Can the system expand if developed?

The project should be developed such that the necessary functions and performance are achieved within the constraints. The project requires High Resolution Scanning device and utilizes Cryptographic techniques. Through the technology may become obsolete after some period of time, due to the fact that newer version of same software supports older versions, the system may still be used. So there are minimal constraints involved with this project. The system has been developed using PHP in front end and MySQL in server in back end, the project is technically feasible for development. The system has been developed using PHP in front end and MySQL in server in back end, the project is technically feasible for development. The System used was also of good performance of Processor Intel i3 core; RAM 4GB and, Hard disk 1TB

3.1.3 Behavioral Feasibility

The proposed system includes the following questions:

- Is there sufficient support for the users?
- Will the proposed system cause harm?

The project would be beneficial because it satisfies the objectives when developed and installed. All behavioral aspects are considered carefully and conclude that the project is behaviorally feasible.

PRAGATHI, GUI is simple so that users can easily use it. PRAGATHI is simple enough so that no training is needed.

3.2 SYSTEM SPECIFICATION

3.2.1 Hardware Specification

Processor	-	intel core i3
RAM	-	4 GB
Hard disk	-	1 TB

3.2.2 Software Specification

Front End	-	HTML, CSS
Backend	-	MYSQL
Client on PC	-	Windows 7 and above.
Technologies used	-	JS, HTML5, AJAX, J Query, PHP, CSS

3.3 SOFTWARE DESCRIPTION

3.3.1 PHP

PHP is a server side scripting language designed for web development but used as a general purpose programming language.PHP is now installed on more than 244 million websites and 2.1 million web servers. Originally created by Rasmus Ledorf in 1995, the reference implementation of PHP is now produced by the PHP group. While PHP originally stood for personal Home page ,it now stands for PHP:HypertextPreprocessor, a recursive acronym.PHP code is interpreted by a web server with a PHP processor module which generates the resulting web page.PHP commands can be embedded directly into a HTML source document rather than calling an external file to process data. It has also evolved to include a command-line interface capability and can be used in standalone incompatible with the GNU General Public License (GPL) due to restrictions on the usage of the term PHP.PHP can be deployed on most web servers and also as a standalone shell on almost every operating system and platform, free of charge.

3.3.2 MySQL

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation. The MySQL Web site provides the latest information about MySQL software.

- **MySQL is a database management system.**

  A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

- **MySQL databases are relational.**

  A relational database stores data in separate tables rather than putting all the data in one big storeroom. The database structures are organized into physical files optimized for speed. The logical model, with objects such as databases, tables, views, rows, and columns, offers a flexible programming environment. You set up rules governing the relationships between different data fields, such as one-to-one, one-to-many, unique, required or optional, and “pointers” between different tables. The database enforces these rules, so that with a well-designed database, your application never sees inconsistent, duplicate, orphan, out-of-date, or missing data. The SQL part of “MySQL” stands for “Structured Query Language”. SQL is the most common standardized language used to access databases. Depending on your programming environment, you might enter SQL directly (for example, to generate reports), embed SQL statements into code written in another language, or use a language-specific API that hides the SQL syntax. SQL is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and several versions exist. In this manual, “SQL92” refers to the standard released in 1992, “SQL: 1999” refers to the standard released in 1999, and “SQL: 2003” refers to the current version of the standard. We use the phrase “the SQL standard” to mean the current version of the SQL Standard at any time.

- **MySQL software is Open Source.**

  Open Source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. The MySQL software uses the GPL (GNU General Public License), to define what you may and may not do with the software in different situations. If you feel uncomfortable with the GPL or need to embed MySQL code into a commercial application, you can buy a commercially licensed version from us. See the MySQL Licensing Overview for more information.

- **The MySQL Database Server is very fast, reliable, scalable, and easy to use.**

  If that is what you are looking for, you should give it a try. MySQL Server can run comfortably on a desktop or laptop, alongside your other applications, web servers, and so on, requiring little or no attention. If you dedicate an entire machine to MySQL, you can adjust the settings to take advantage of all the memory, CPU power, and I/O capacity available.

- **MySQL Server works in client/server or embedded systems.**

  The MySQL Database Software is a client/server system that consists of a multi-threaded SQL server that supports different backends, several different client programs and libraries, administrative tools, and a wide range of application programming interfaces (APIs). We also provide MySQL Server as an embedded multi-threaded library that you can link into your application to get a smaller, faster, easier-to-manage standalone product.

CHAPTER 4

SYSTEM DESIGN

4.1 INTRODUCTION

Design is the first step into the development phase for any engineered product or system. Design is a creative process. A good design is the key to effective system. The term “design” is defined as “the process of applying various techniques and principles for the purpose of defining a process or a system in sufficient detail to permit its physical realization”. It may be defined as a process of applying various techniques and principles for the purpose of defining a device, a process or a system in sufficient detail to permit its physical realization. Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm that is used. The system design develops the architectural detail required to build a system or product. As in the case of any systematic approach, this software too has undergone the best possible design phase fine tuning all efficiency, performance and accuracy levels. The design phase is a transition from a user oriented document to a document to the programmers or database personnel. System design goes through two phases of development: Logical and Physical Design.

4.2 UML DIAGRAM

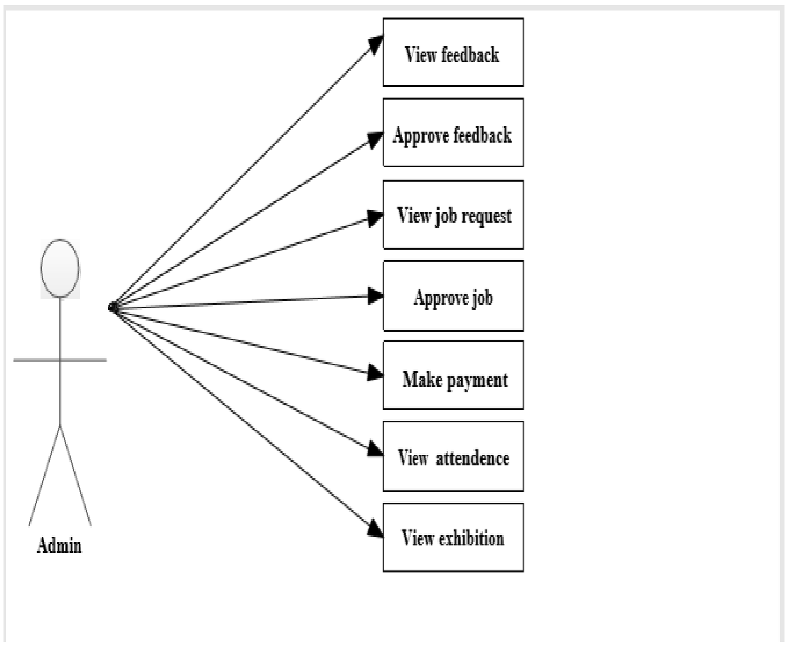
A use case diagram is a graphic depiction of the interactions among the elements of a system. A use case is a methodology used in system analysis to identify, clarify, and organize system requirements. In this context, the term "system" refers to something being developed or operated, such as a mail-order product sales and service Web site. Use case diagrams are employed in UML (Unified Modeling Language), a standard notation forthe modeling of real-world objects and systems.

System objectives can include planning overall requirements, validating a hardware design, testing and debugging a software product under development, creating an online help reference, or performing a consumer-service-oriented task. For example, use cases in a product sales environment would include item ordering, catalog updating, payment processing, and customer relations. A use case diagram contains four components.

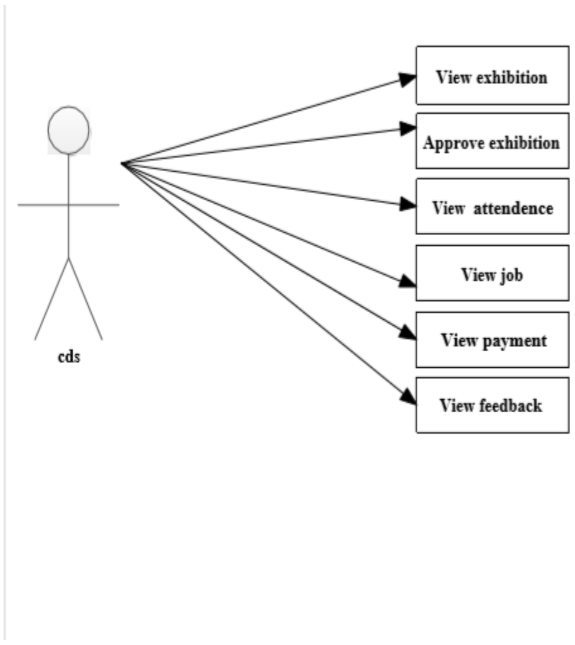
- The boundary, which defines the system of interest in relation to the world around it.
- The actors, usually individuals involved with the system defined according to their roles.
- The use cases, which are the specific roles are played by the actors within and around the system.
- The relationships between and among the actors and the use cases.

4.2.1 Usecase diagram

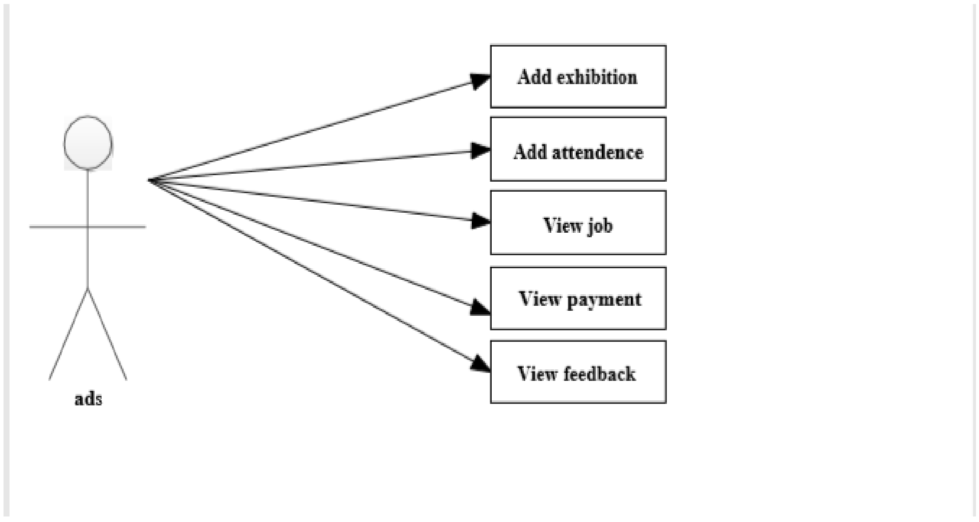
- Usecase diagram for Admin :



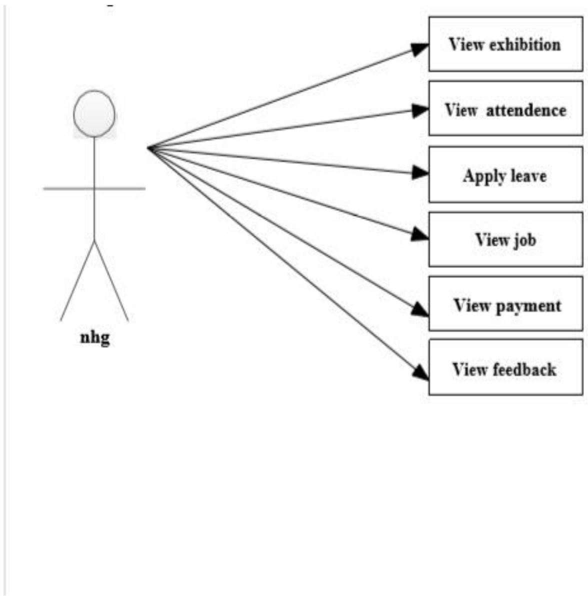
- Usecase diagram for cds:



- Usecase diagram for ads:

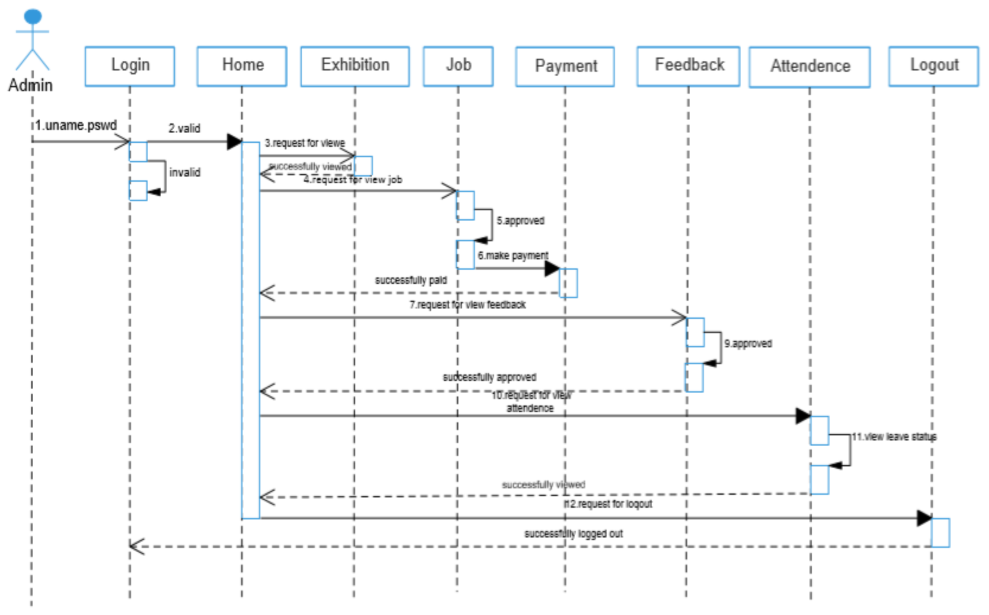


- Usecase diagram for nhg(unit member):

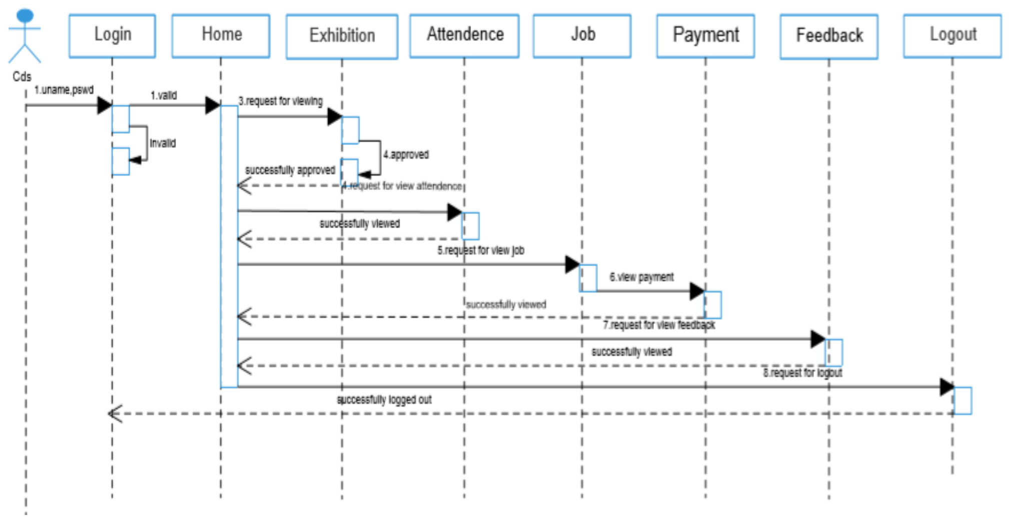


4.2.2 Sequential diagram

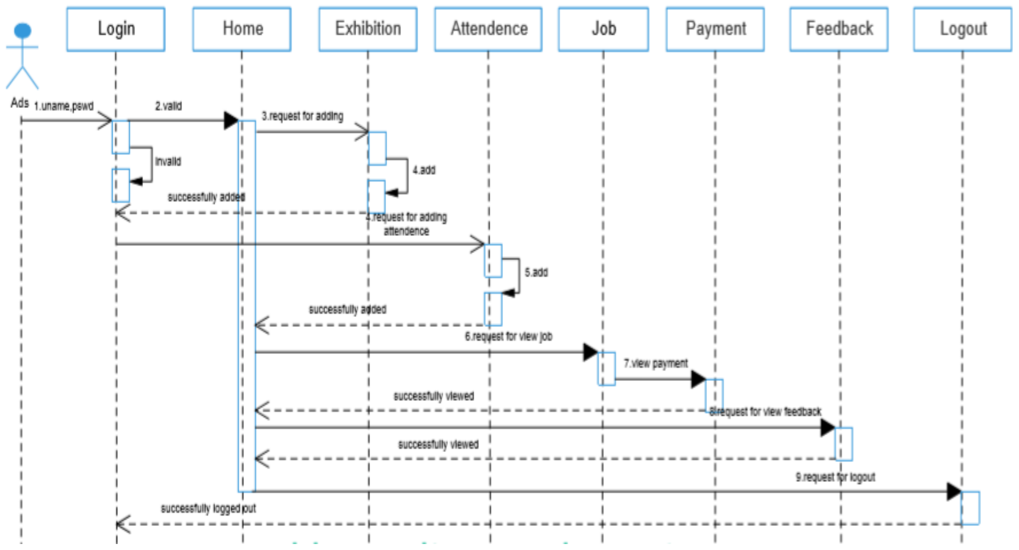
➢ **SEQUENCE DIAGRAM FOR ADMIN**



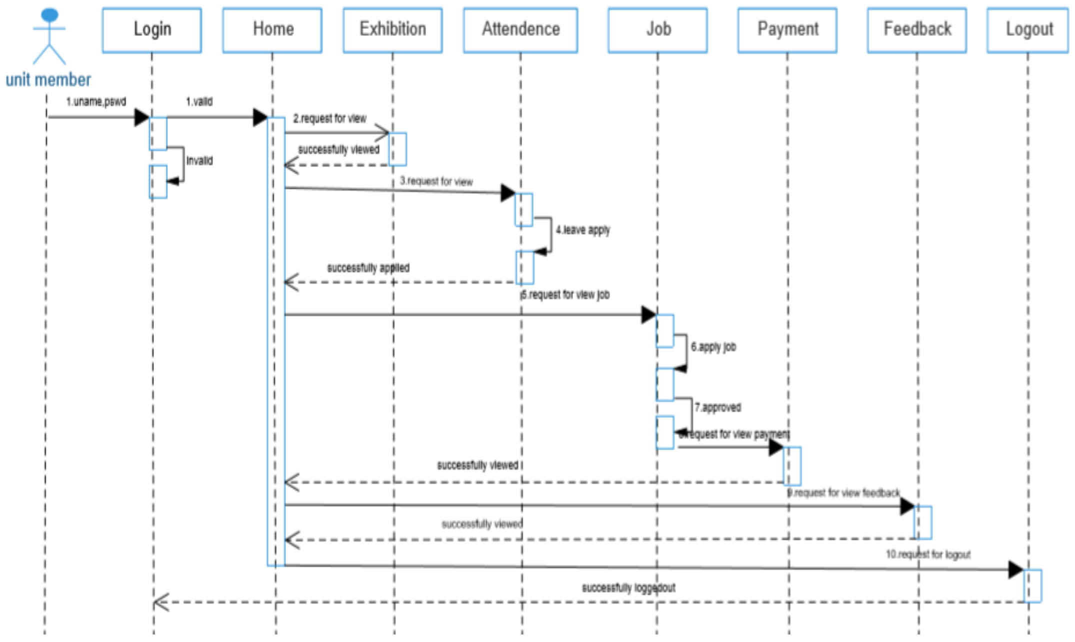
➢ **SEQUENCE DIAGRAM FOR CDS**



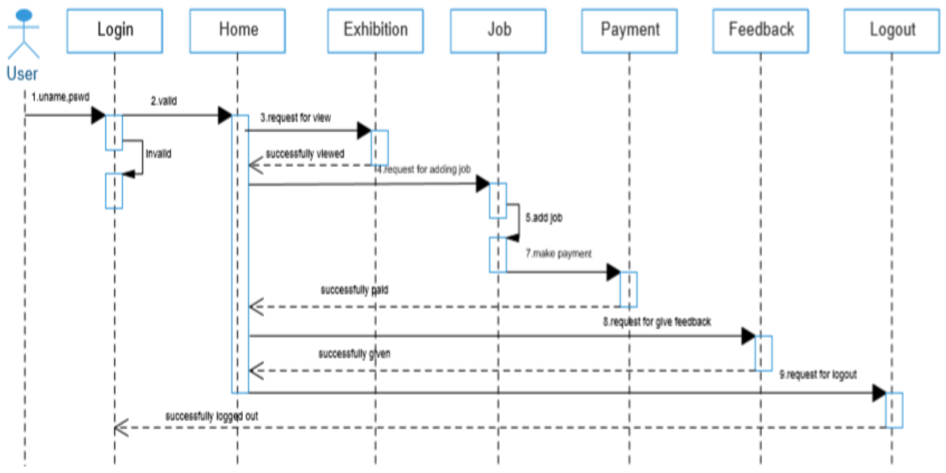
➢ SEQUENCE DIAGRAM FOR ADS



➢ SEQUENCE DIAGRAM FOR NHG (UNIT MEMBER)



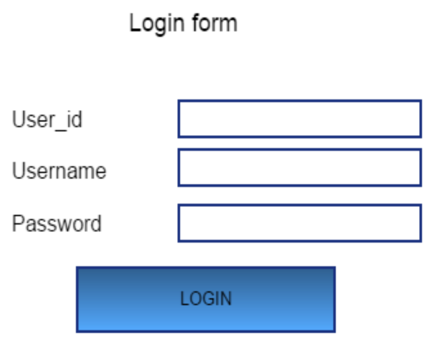
➢ SEQUENCE DIAGRAM FOR USERS



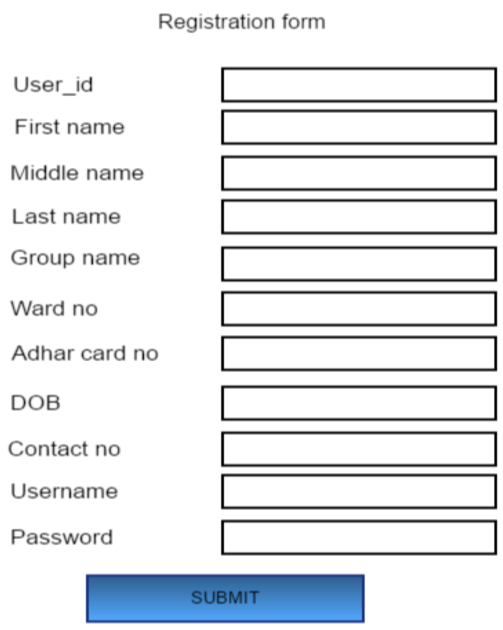
4.3 USER INTERFACE DESIGN

4.3.1-INPUT DESIGN

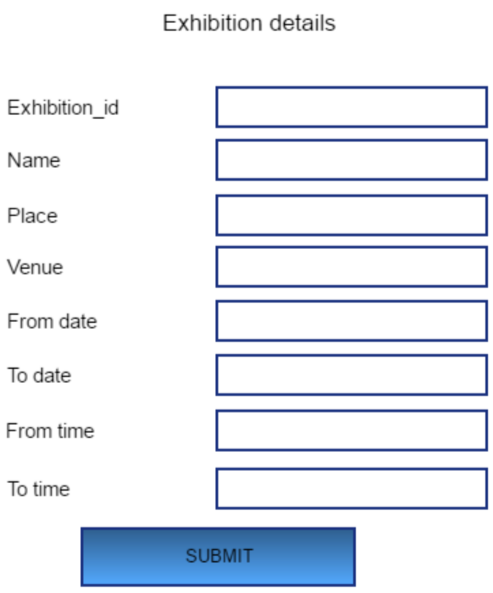
Login Form



Registration Form

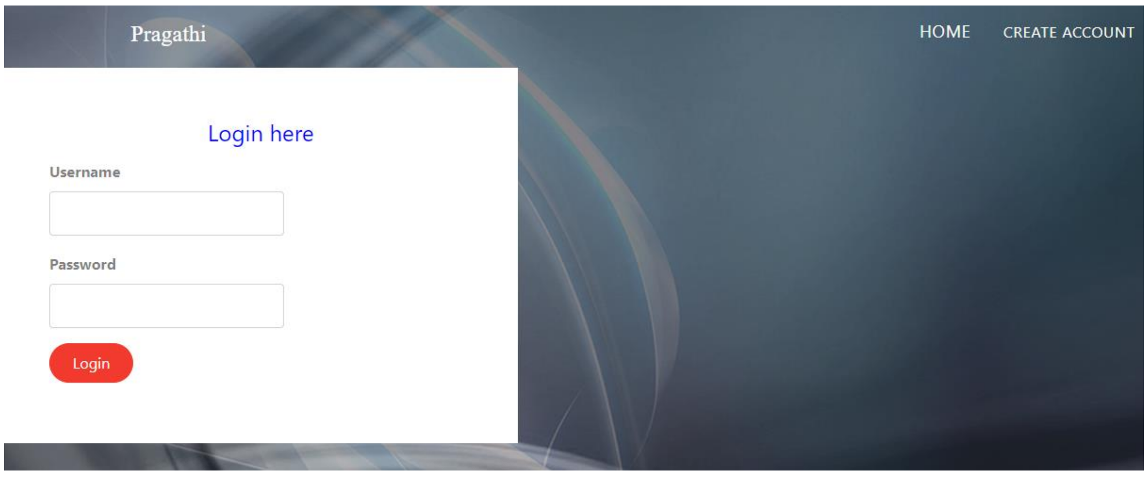


Exhibition Form

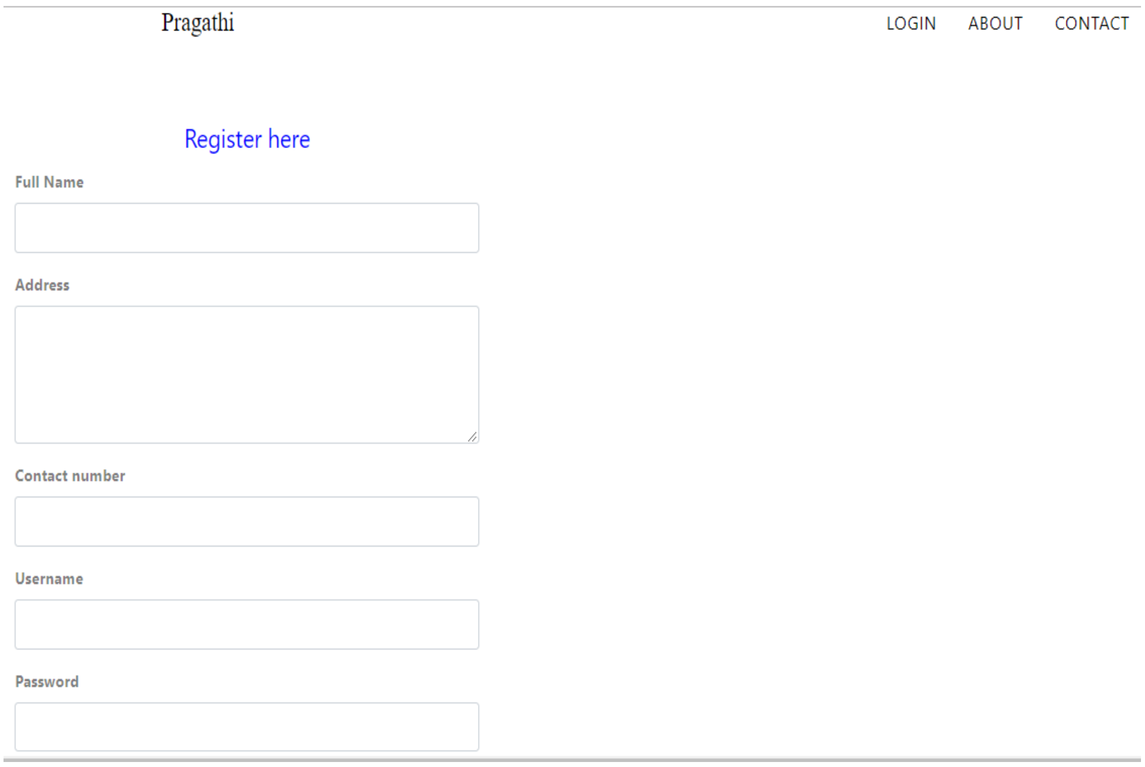


4.3.2 OUTPUT DESIGN

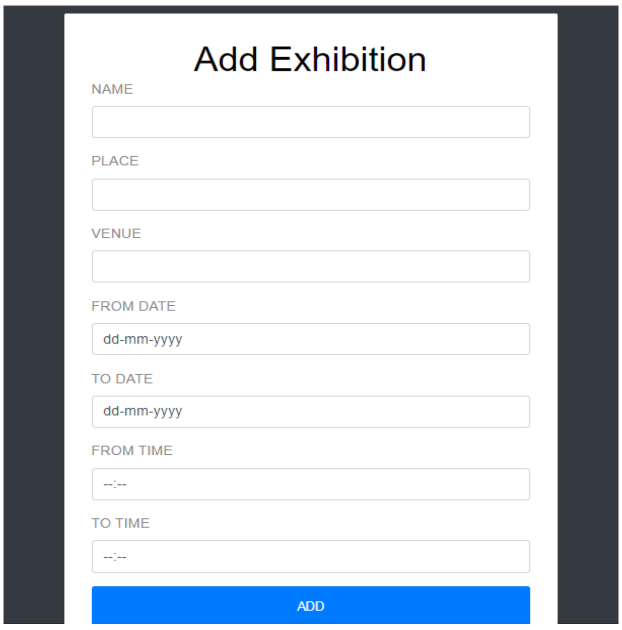
Login Page



User Registration page



Exhibition Registration



4.4 Database Design

A database is an organized mechanism that has the capability of storing information through which a user can retrieve stored information in an effective and efficient manner. The data is the purpose of any database and must be protected.

The database design is a two level process. In the first step, user requirements are gathered together and a database is designed which will meet these requirements as clearly as possible. This step is called Information Level Design and it is taken independent of any individual DBMS.

In the second step, this Information level design is transferred into a design for the specific DBMS that will be used to implement the system in question. This step is called Physical Level Design, concerned with the characteristics of the specific DBMS that will be used. A database design runs parallel with the system design. The organization of the data in the database is aimed to achieve the following two major objectives.

- Data Integrity
- Data independence

4.4.1 Relational Database Management System (RDBMS)

A relational model represents the database as a collection of relations. Each relation resembles a table of values or file of records. In formal relational model terminology, a row is called a tuple, a column header is called an attribute and the table is called a relation. A relational database consists of a collection of tables, each of which is assigned a unique name. A row in a tale represents a set of related values.

Relations, Domains & Attributes

A table is a relation. The rows in a table are called tuples. A tuple is an ordered set of n elements. Columns are referred to as attributes. Relationships have been set between every table in the database. This ensures both Referential and Entity Relationship Integrity. A domain D is a set of atomic values. A common method of specifying a domain is to specify a data type from which the data values forming the domain are drawn. It is also useful to specify a name for the domain to help in interpreting its values. Every value in a relation is atomic, that is not decomposable.

Relationships

- Table relationships are established using Key. The two main keys of prime importance are Primary Key & Foreign Key. Entity Integrity and Referential Integrity Relationships can be established with these keys.
- Entity Integrity enforces that no Primary Key can have null values.
- Referential Integrity enforces that no Primary Key can have null values.
- Referential Integrity for each distinct Foreign Key value, there must exist a matching Primary Key value in the same domain. Other key are Super Key and Candidate Keys.

4.4.2 Normalization

Data are grouped together in the simplest way so that later changes can be made with minimum impact on data structures. Normalization is formal process of data structures in manners that eliminates redundancy and promotes integrity. Normalization is a technique of separating redundant fields and breaking up a large table into a smaller one. It is also used to avoid insertion, deletion, and updating anomalies. Normal form in data modelling use two concepts, keys and relationships. A key uniquely identifies a row in a table. There are two types of keys, primary key and foreign key. A primary key is an element or a combination of elements in a table whose purpose is to identify records from the same table. A foreign key is a column in a table that uniquely identifies record from a different table. All the tables have been normalized up to the third normal form.

As the name implies, it denotes putting things in the normal form. The application developer via normalization tries to achieve a sensible organization of data into proper tables and columns and where names can be easily correlated to the data by the user. Normalization eliminates repeating groups at data and thereby avoids data redundancy which proves to be a great burden on the computer resources. These include:

4.4.2.1 Normalize the data.

4.4.2.2 Choose proper names for the tables and columns.

4.4.2.3 Choose the proper name for the data.

First Normal Form

The First Normal Form states that the domain of an attribute must include only atomic values and that the value of any attribute in a tuple must be a single value from the domain of that attribute. In other words 1NF disallows “relations within relations” or “relations as attribute values within tuples”. The only attribute values permitted by 1NF are single atomic or indivisible values. The first step is to put the data into First Normal Form. This can be donor by moving data into separate tables where the data is of similar type in each table. Each table is given a Primary Key or Foreign Key as per requirement of the project. In this we form new relations for each non-atomic attribute or nested relation. This eliminated repeating groups of data. A relation is said to be in first normal form if only if it satisfies the constraints that contain the primary key only.

Second Normal Form

According to Second Normal Form, for relations where primary key contains multiple attributes, no non-key attribute should be functionally dependent on a part of the primary key. In this we decompose and setup a new relation for each partial key with its dependent attributes. Make sure to keep a relation with the original primary key and any attributes that are fully functionally dependent on it. This step helps in taking out data that is only dependent on a part of the key. A relation is said to be in second normal form if and only if it satisfies all the first normal form conditions for the primary key and every non-primary key attributes of the relation is fully dependent on its primary key alone.

Third Normal Form

According to Third Normal Form, Relation should not have a non-key attribute functionally determined by another non-key attribute or by a set of non-key attributes. That is, there should be no transitive dependency on the primary key. In this we decompose and set up relation that includes the non-key attributes that functionally determines other non-key attributes. This step is taken to get rid of anything that does not depend entirely on the Primary Key. A relation is said to be in third normal form if only if it is in second normal form and more over the non key attributes of the relation should not be depend on other non-key attribute.

4.5 TABLE DESIGN

1. **Table Name** :reg

   Description: Table showing registration details.

Data Field	Data Type	Constraint	Description
user_id	Varchar(5)	Primary Key	Id of user
Full_name	Varchar(20)	Notnull	Name of user
address	Varchar(50)	Notnull	Address of user
group_name	Varvhar(15)	Notnull	Group name
ward_no	Varchar(6)	Notnull	Ward number
adhar_no	Number(12)	Notnull	Adhar number
dob	date	Notnull	Date of birth
contact_no	Number(10)	Notnull	Contact number
user_name	Varchar(10)	Notnull	Username of user
password	Varchar(10)	Notnull	Password of user

2. **Table Name:** login

   Description: Table showing login details.

Data Field	Data Type	Constraint	Description
logid	Varchar(10)	Primary Key	Login id of user
userid	Varchar(10)	Foreign Key	Id of user
username	Varchar(10)	Not null	Username
password	Varchar(10)	Not null	Password

3. **Table Name:** exhibition

   Description: Table showing exhibition details.

Data Field	Data Type	Constraint	Description
exh_id	Varchar(8)	Primary Key	Id of exhibition
name	Varchar(8)	Not null	Name of exhibition
f_date	date	Not null	From date
t_date	date	Not null	To date
place	Varchar(14)	Not null	Place of exhibition
venue	Varchar(20)	Not null	Venue
f_time	date	Not null	From time
t_time	date	Not null	To time

4. **Table Name:** feedback

   Description: Table showing feedback details.

Data field	Data type	Constraint	Description
fe_id	varchar(8)	Primary key	Id of feedback
feedback	varchar(8)	Not null	Feedback

5. **Table Name:** job

Description: Table showing job details.

Data Field	**Data Type**	**Constraint**	**Description**
job_id	varchar(8)	Primary key	Id of job
f-date	date	Not null	From date
t_date	date	Not null	To date
place	varchar(30)	Not null	Place of job
no_workers	number(500)	Not null	Number of workers n
f_time	time	Not null	From time
t_time	time	Not null	To time

6. **Table Name:** payment

Description: Table showing payment details.

Data field	**Data type**	**Constraint**	**Description**
payment_id	Varchar(8)	Primary Key	Id of payment
c_name	Varchar(20)	Notnull	Name in the card
c_no	Number(10)	Notnull	Number in card
c_type	Varchar(30)	Notnull	Card type
E_date	date	Notnull	Expiary date
cvv	number	Notnull	CVV number

8. **Table Name:** leave

Description: Table showing leave details.

Data field	Data type	Constraint	Description
le_id	varchar(8)	Primary key	Id of leave
date	date	Not null	Date of leave
an	time	Not null	Afternoon
fn	time	Not null	Fornoon
fullday	time	Not null	Full day

CHAPTER 5

TESTING

5.1 INTRODUCTION

Software Testing is the process of executing software in a controlled manner, in order to answer the question - Does the software behave as specified? Software testing is often used in association with the terms verification and validation. Validation is the checking or testing of items, includes software, for conformance and consistency with an associated specification. Software testing is just one kind of verification, which also uses techniques such as reviews, analysis, inspections, and walkthroughs. Validation is the process of checking that what has been specified is what the user actually wanted.

Other activities which are often associated with software testing are static analysis and dynamic analysis. Static analysis investigates the source code of software, looking for problems and gathering metrics without actually executing the code. Dynamic analysis looks at the behavior of software while it is executing, to provide information such as execution traces, timing profiles, and test coverage information.

Testing is a set of activity that can be planned in advanced and conducted systematically. Testing begins at the module level and work towards the integration of entire computers based system. Nothing is complete without testing, as it vital success of the system testing objectives, there are several rules that can serve as testing objectives. They are:

Testing is a process of executing a program with the intent of finding an error.

- A good test case is one that has high possibility of finding an undiscovered error.
- A successful test is one that uncovers an undiscovered error.

If a testing is conducted successfully according to the objectives as stated above, it would uncover errors in the software. Also testing demonstrate that the software function appear to be working according to the specification, that performance requirement appear to have been met.

There are three ways to test program.

- For correctness
- For implementation efficiency
- For computational complexity

Test for correctness are supposed to verify that a program does exactly what it was designed to do. This is much more difficult than it may at first appear, especially for large programs.

5.2 TEST PLAN

A test plan implies a series of desired course of action to be followed in accomplishing various testing methods. The Test Plan acts as a blue print for the action that is to be followed. The software engineers create a computer program, its documentation and related data structures. The software developers is always responsible for testing the individual units of the programs, ensuring that each performs the function for which it was designed. There is an independent test group (ITG) which is to remove the inherent problems associated with letting the builder to test the thing that has been built. The specific objectives of testing should be stated in measurable terms. So that the mean time to failure, the cost to find and fix the defects, remaining defect density or frequency of occurrence and test work-hours per regression test all should be stated within the test plan.

The levels of testing include:

- Unit testing
- Integration Testing
- Data validation Testing
- Output Testing

5.2.1 Unit Testing

Unit testing focuses verification effort on the smallest unit of software design – the software component or module. Using the component level design description as a guide, important control paths are tested to uncover errors within the boundary of the module. The relative complexity of tests and uncovered scope established for unit testing. The unit testing is white-box oriented, and step can be conducted in parallel for multiple components. The modular interface is tested to ensure that information properly flows into and out of the program unit under test. The local data structure is examined to ensure that data stored temporarily maintains its integrity during all steps in an algorithm's execution. Boundary conditions are tested to ensure that all statements in a module have been executed at least once. Finally, all error handling paths are tested.

Tests of data flow across a module interface are required before any other test is initiated. If data do not enter and exit properly, all other tests are moot. Selective testing of execution paths is an essential task during the unit test. Good design dictates that error conditions be anticipated and error handling paths set up to reroute or cleanly terminate processing when an error does occur. Boundary testing is the last task of unit testing step. Software often fails at its boundaries.

Unit testing was done in Sell-Soft System by treating each module as separate entity and testing each one of them with a wide spectrum of test inputs. Some flaws in the internal logic of the modules were found and were rectified. After coding each module is tested and run individually. All unnecessary code where removed and ensured that all modules are working, and gives the expected result.

5.2.2 Integration Testing

Integration testing is systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing. The objective is to take unit tested components and build a program structure that has been dictated by design. The entire program is tested as whole. Correction is difficult because isolation of causes is complicated by vast expanse of entire program. Once these errors are corrected, new ones appear and the process continues in a seemingly endless loop. After performing unit testing in the System all the modules were integrated to test for any inconsistencies in the interfaces. Moreover differences in program structures were removed and a unique program structure was evolved.

5.2.3 Validation Testing or System Testing

This is the final step in testing. In this the entire system was tested as a whole with all forms, code, modules and class modules. This form of testing is popularly known as Black Box testing or System tests.

Black Box testing method focuses on the functional requirements of the software. That is, Black Box testing enables the software engineer to derive sets of input conditions thatwill fully exercise all functional requirements for a program.

Black Box testing attempts to find errors in the following categories; incorrect or missing functions, interface errors, errors in data structures or external data access, performance errors and initialization errors and termination errors.

5.2.4 Output Testing or User Acceptance Testing

The system considered is tested for user acceptance; here it should satisfy the firm's need. The software should keep in touch with perspective system; user at the time of developing and making changes whenever required. This done with respect to the following points:

- Input Screen Designs,
- Output Screen Designs,

The above testing is done taking various kinds of test data. Preparation of test data plays a vital role in the system testing. After preparing the test data, the system under study is tested using that test data. While testing the system by which test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

CHAPTER 6

IMPLEMENTATION

6.1 INTRODUCTION

Implementation is the stage of the project where the theoretical design is turned into a working system. It can be considered to be the most crucial stage in achieving a successful new system gaining the users confidence that the new system will work and will be effective and accurate. It is primarily concerned with user training and documentation. Conversion usually takes place about the same time the user is being trained or later. Implementation simply means convening a new system design into operation, which is the process of converting a new revised system design into an operational one.

At this stage the main work load, the greatest upheaval and the major impact on the existing system shifts to the user department. If the implementation is not carefully planned or controlled, it can create chaos and confusion.

Implementation includes all those activities that take place to convert from the existing system to the new system. The new system may be a totally new, replacing an existing manual or automated system or it may be a modification to an existing system. Proper implementation is essential to provide a reliable system to meet organization requirements. The process of putting the developed system in actual use is called system implementation. This includes all those activities that take place to convert from the old system to the new system. The system can be implemented only after through testing is done and if it is found to be working according to the specifications. The system personnel check the feasibility of the system. The more complex the system being implemented, the more involved will be the system analysis and design effort required to implement the three main aspects: education and training, system testing and changeover.

The implementation state involves the following tasks:

- Careful planning.
- Investigation of system and constraints.
- Design of methods to achieve the changeover.

6.2 IMPLEMENTATION PROCEDURES

Implementation of software refers to the final installation of the package in its real environment, to the satisfaction of the intended uses and the operation of the system. In many organizations someone who will not be operating it, will commission the software development project. In the initial stage people doubt about the software but we have to ensure that the resistance does not build up, as one has to make sure that:

- The active user must be aware of the benefits of using the new system.
- Their confidence in the software is built up.
- Proper guidance is imparted to the user so that he is comfortable in using the application.

Before going ahead and viewing the system, the user must know that for viewing the result, the server program should be running in the server. If the server object is not up running on the server, the actual process won’t take place.

6.2.1 User Training

User training is designed to prepare the user for testing and converting the system. To achieve the objective and benefits expected from computer based system, it is essential for the people who will be involved to be confident of theirrole in the new system. As system becomes more complex, the need for training is more important. By user training the user comes to know how to enter data, respond to error messages, interrogate the database and call up routine that will produce reports and perform other necessary functions.

6.2.2 Training on the Application Software

After providing the necessary basic training on computer awareness the user will have to be trained on the new application software. This will give the underlying philosophy of the use of the new system such as the screen flow, screen design type of help on the screen, type of errors while entering the data, the corresponding validation check at each entry and the ways to correct the date entered. It should then cover information needed by the specific user/ group to use the system or part of the system while imparting the training of the program on the application. This training may be different across different user groups and across different levels of hierarchy.

6.2.3 System Maintenance

Maintenance is the enigma of system development. The maintenance phase of the software cycle is the time in which a software product performs useful work. After a system is successfully implemented, it should be maintained in a proper manner. System maintenance is an important aspect in the software development life cycle. The need for system maintenance is for it to make adaptable to the changes in the system environment. Software maintenance is of course, far more than "Finding Mistakes".

CHAPTER 7

CONCLUSION AND FUTURE SCOPE

7.1 CONCLUSION

The aim of our project is to computerize the manual activities of a KudumbaShree. Our system ”PRAGATHI” consumes less time, work efficiently with the available resources, the information will be more accurate and update the database can do without much human error. We hope that our system effectively overcomes the defects of the existing system.

KudumbaShree has been recognized as an effective strategy for empowerment of women in rural as well as urban areas. The overall empowerment of women’s closely linked to economic empowerment, which can leverage recent technological advancements. The application PRAGATHI is capable of assessing many parameters, so that each user can be used this system effectively. It can incorporate more parameters in future.

7.2 FUTURE SCOPE

It can be used for different KudumbaShree units in different panchayats in Kerala. It will be a victorious web application for the members includes in the KudumbaShree units and they can use this application with a minimum computer knowledge. It can used for avoid paper works and manual financial transactions conducted in KudumbaShree..

CHAPTER 8

BIBLIOGRAPHY

REFERENCES:

- Gary B. Shelly, Harry J. Rosenblatt, “*System Analysis and Design*”, 2009.
- Roger S Pressman, “*Software Engineering*”, 1994.
- PankajJalote, “So*ftware engineering*: a precise approach”, 2006.
- James lee and Brent ware Addison, “Open source web development with LAMP”, 2003
- IEEE Std 1016 Recommended Practice for Software Design Descriptions.

WEBSITES:

- www.w3schools.com
- www.jquery.com
- http://homepages.dcc.ufmg.br/~rodolfo/es-1-03/IEEE-Std-830-1998.pdf
- www.agilemodeling.com/artifacts/useCaseDiagram.html

CHAPTER 9

APPENDIX

9.1 Sample Code

adminaddcds.php

```
<?php
session_start();
$login=$_SESSION['login'];
$type=$_SESSION['type'];
if($login)
{
  ?>
<!doctype html>
                    <div>
                       <label>Full Name</label>
                       <input      type="text"      id="name"      name="name"
class="form-control"  placeholder=""  onChange="return  validateform1()"
autocomplete="off" required>
                                                         <script>
                              function validateform1()
                              {
var x=document.forms["registration"]["name"].value;
if(x=="")
{
alert("Please Fill name");
document.getElementById("name").focus();
return false;
}
var x=new RegExp("^[a-zA-z ]*$");
```

```
if(!x.test(document.registration.name.value))
{
   alert("Enter name with characters only");
   registration.name.focus();
   document.getElementById("name").value="";
   return false;
}
var x=document.forms["registration"]["name"].value;
if(x.length<2 || x.length>30)
{
alert("Please enter name with more than 2 letters and less than 30 letters");
document.getElementById("name").focus();
document.getElementById("name").value="";
return false;
}
return true;
                  }

                           </script>
                  </div>
                                         <div class="form-group">
                     <label>Address</label>
                     <textarea name="address" id="class="address" class="form-control" placeholder="" onChange="return validateform2()" autocomplete="off" required></textarea>
                                                 <script>
                           function validateform2()
                           {
                                   var
```

```
    x=document.forms["registration"]["address"].value;
if(x=="")
{
alert("Please Fill address");
document.getElementById("address").focus();
return false;
}
return true;
                                    }
                                    </script>
                    </div>
                                                    <div class="form-group">
                        <label>Adhar number</label>
                        <input type="text" name="anumber" id="ano" class="form-
control"    placeholder=""    onChange="return    validateform4()"
autocomplete="off" required>
                                                            <script>
                                    function validateform4()
                                    {

var x=document.forms["registration"]["ano"].value;
if(x=="")
{
alert("Please Fill Adhar number");
 document.getElementById("ano").focus();
return false;
}
else if(document.registration.ano.value.length!="12")
{
        alert("Provide Adher number as 12 numbers");
        document.getElementById("ano").focus();
        document.getElementById("ano").value="";
        return false;
}
var x=document.forms["registration"]["ano"].value;
if(isNaN(x))
{
        alert("Only numbers are allowed");
        document.getElementById("ano").focus();
        document.getElementById("ano").value="";
        return false;
}
return true;
                                        }
                                        </script>
                    </div>
                                                        <div class="form-group">
                        <label>Date of birth</label>
                        <input      type="date"     name="dob"      id="dob"
class="form-control"          placeholder=""           onChange="return
validateform3()" autocomplete="off" required>
                         <script>
                                        function validateform3()
                                        {
var x=document.forms["registration"]["dob"].value;
if(x=="")
{
alert("Please Fill Date of birth");
```

```
document.getElementById("dob").focus();
return false;
}

var x =document.forms["registration"]["dob"].value;
var d = "1960-01-01";
if(x < d){
   alert('Please enter year above 1960');
     document.getElementById("dob").focus();
     document.getElementById("dob").value="";
     return false;
}
var y =document.forms["registration"]["dob"].value;
var s = "2000-01-01";
if(y > s){
   alert('Please enter year below 2000');
     document.getElementById("dob").focus();
     document.getElementById("dob").value="";
     return false;
}
return true;
                              }
                              </script>
                                              </div>
                    <div class="form-group">
                       <label>contact number</label>
                       <input    type="text"    id="cno"    name="cnumber"
autocomplete="off"         class="form-control"         placeholder=""
onChange="return validateform6()" required>
```

```
                                        <script>
                        function validateform6()
                        {
                                var digits = "0123456789";
    var phoneNumberDelimiters = "()- ";
    var validWorldPhoneChars = phoneNumberDelimiters + "+";
    var minDigitsInIPhoneNumber = 10;

    function isInteger(s)
    {   var i;
        for (i = 0; i < s.length; i++)
  {
     var c = s.charAt(i);
     if (((c < "0") || (c > "9"))) return false;
  }

  return true;
}
function trim(s)
{   var i;
  var returnString = "";

  for (i = 0; i < s.length; i++)
  {

     var c = s.charAt(i);
     if (c != " ") returnString += c;
  }
  return returnString;
}
function stripCharsInBag(s, bag)
{   var i;
   var returnString = "";

   for (i = 0; i < s.length; i++)
   {

      var c = s.charAt(i);
      if (bag.indexOf(c) == -1) returnString += c;
   }
   return returnString;
}

var x=new RegExp("^([6-9]{1})([0-9]{9})$");
if(!x.test(document.registration.cno.value))
{
     alert("Phone number is invalid.Phone number must begin with
9,8,7 or 6");
     registration.cno.focus();
     return false;
}
return true;
                              }
                              </script>
                   </div>
                                              <!--<div class="form-group">
                      <label>Username</label>
                      <input   type="text"   id="uname"   name="uname"
```

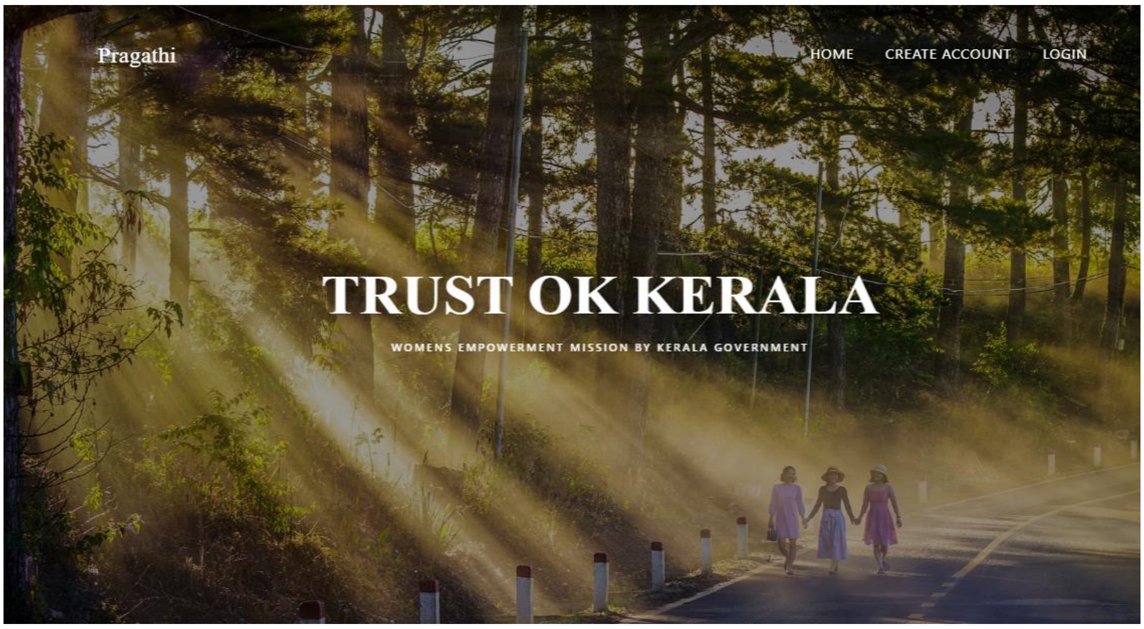
```
        autocomplete="off"        class="form-control"        placeholder=""
        onChange="return validateform7()">
                                                        <script>
                                        function validateform7()
                                        {
                                        var
        x=document.forms["registration"]["uname"].value;
        if(x=="")
        {
        alert("Please Fill username");
         document.getElementById("uname").focus();
        return false;
        }

else if(document.registration.uname.value.length<6)
{
      alert("Enter username with more than six characters");
      document.getElementById("uname").focus();
      document.getElementById("uname").value="";
      return false;
}
return true;
                                  }

                                                  <script>
                                  function validateform8()
                                  {
                                        var x=document.forms["registration"]["pass"].value;
if(x=="")
{
alert("Please Fill  password");
 document.getElementById("pass").focus();
return false;
}
else if(document.registration.pass.value.length<8)
{
        alert("Password should be minimum 8 characters long");
        document.getElementById("pass").focus();
        return false;
}
var passw=  /^[A-Za-z]\w{7,14}$/;
if(document.registration.pass.value.match(passw))
{


}
else
{
alert('Password must contain numbers,characters,underscore and first
letter as character');
document.getElementById("pass").focus();
return false;
}
return true;
                                    }
                                    </script>
                      </div>
                                                        <div class="form-group">
                         <label>confirm Password</label>
```

```
                    <input type="password" id="cpass" name="cpass"
autocomplete="off"      class="form-control"      placeholder=""
onChange="return validateform9()" required>
                                                  <script>
                              function validateform9()
                              {
                                    var
x=document.forms["registration"]["cpass"].value;
if(x=="")
{
alert("Please Fill confirm password");
  document.getElementById("cpass").focus();
 return false;
 }

 var pwd=document.getElementById("pass").value;
 var cpwd=document.getElementById("cpass").value;
 if(pwd!=cpwd){
      alert("Password not matching");
           document.getElementById("cpass").focus();
    document.getElementById("cpass").value="";
           return false;
 }
 return true;
                              }
                              </script>

 </html>
 <?php
```

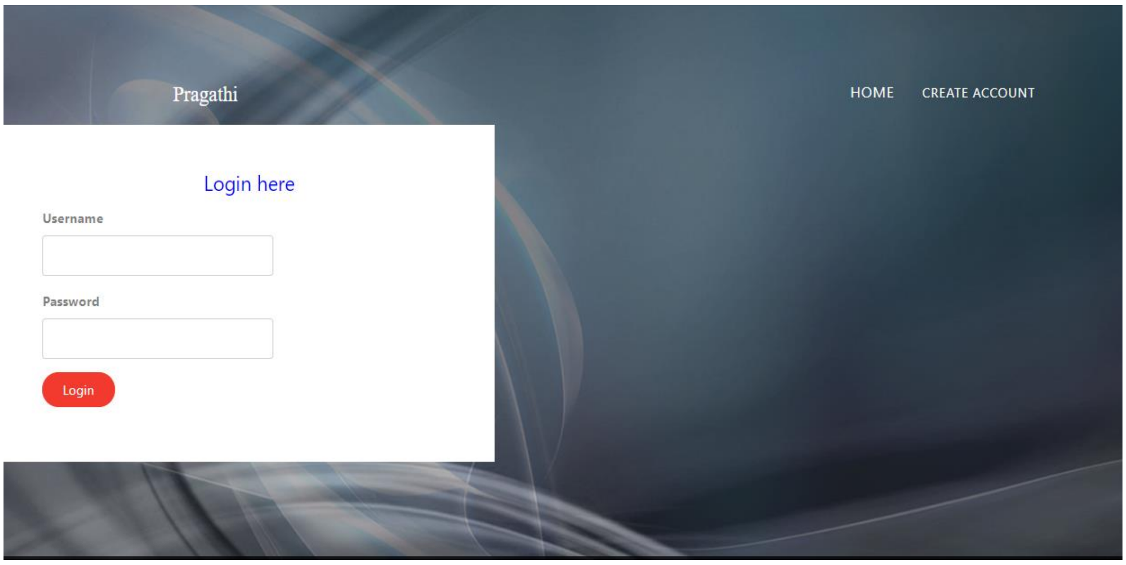
```
include 'co.php';
if(isset($_POST['submit']))
{
  $a=$_POST['name'];
  $b=$_POST['address'];
  $c=$_POST['dob'];
  $d=$_POST['anumber'];
  $e=$_POST['cnumber'];
  //$g=$_POST['uname'];
  $h=$_POST['pass'];
   //echo "<script>alert('$g');</script>";
$sq="insert                                                                    into
login(username,password,usertype,status)values('$e','$h',4,1)";
if(mysqli_query($con,$sq))
{
      $s=mysqli_query($con,"select   logid   from   login   where
username='$e'");
      $r=mysqli_fetch_array($s,MYSQLI_ASSOC);
      $lid=$r['logid'];
      //echo "<script>alert('$lid');</script>";
$sql="insertinto
reg(logid,name,address,dob,ano,cno)values('$lid','$a','$b','$c','$d','$e')
";
 $ch=mysqli_query($con,$sql);
if($ch)
{?>
       <script>
 alert("Regesteration Successfull");
 window.location='admincds.php'
</script>
     <?php
}
else
{
  echo"error:".$sql."<br>".mysqli_error($con);
}
}

else
{?>
     <script>
 alert("User Already Exists");
</script>";
<?php
}
mysqli_close($con);
?>
<?php
}
}

else
header("location:/MINIPROJECT/login.html");
?>
```

9.2 Screen Shots

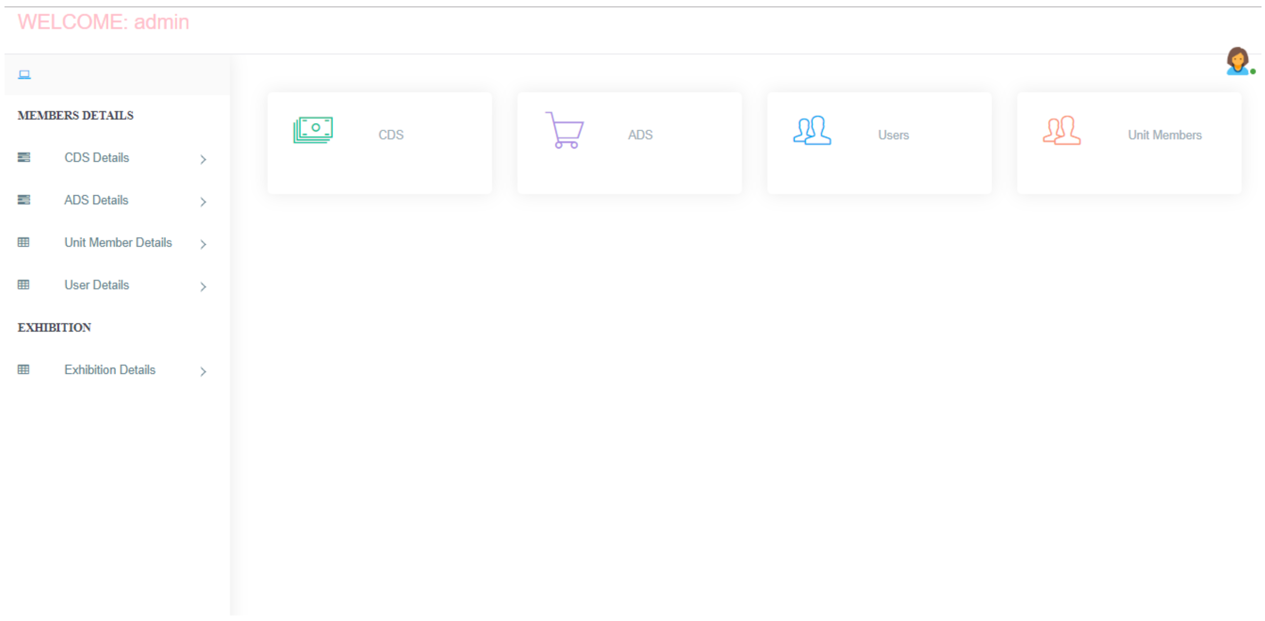
➢ **Home page**



➢ **Login page**



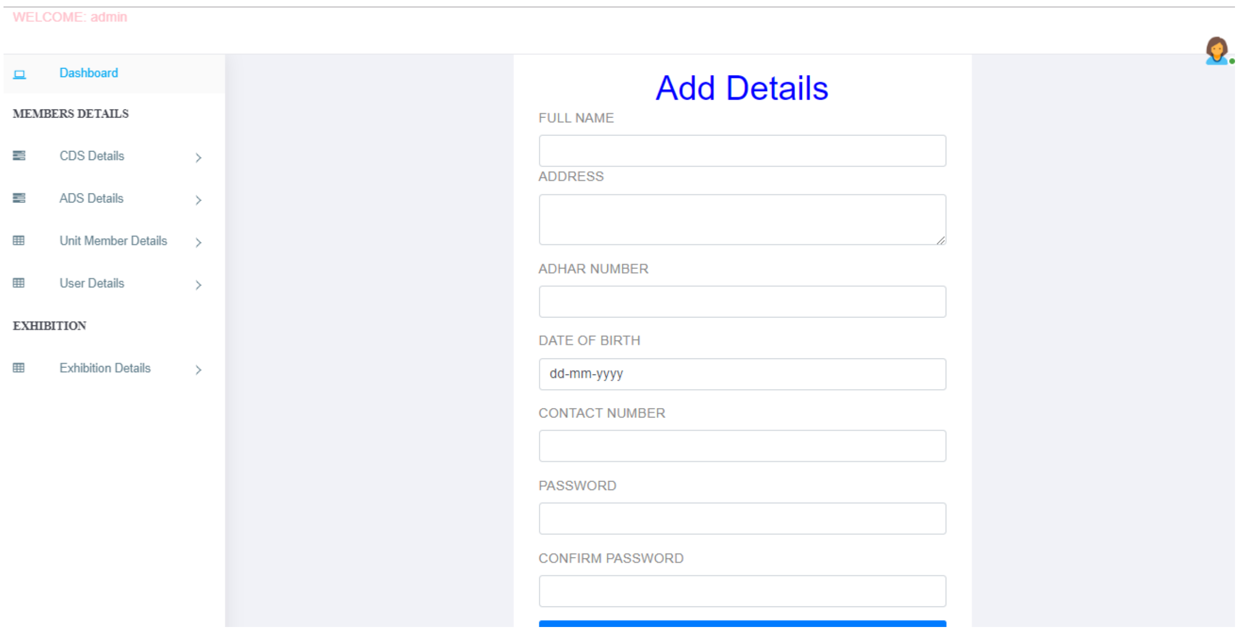
➢ .Admin page


WELCOME: admin

MEMBERS DETAILS
- CDS Details
- ADS Details
- Unit Member Details
- User Details

EXHIBITION
- Exhibition Details

➢ Adding Cds


WELCOME: admin

Dashboard

MEMBERS DETAILS
- CDS Details
- ADS Details
- Unit Member Details
- User Details

EXHIBITION
- Exhibition Details

Add Details

FULL NAME

ADDRESS

ADHAR NUMBER

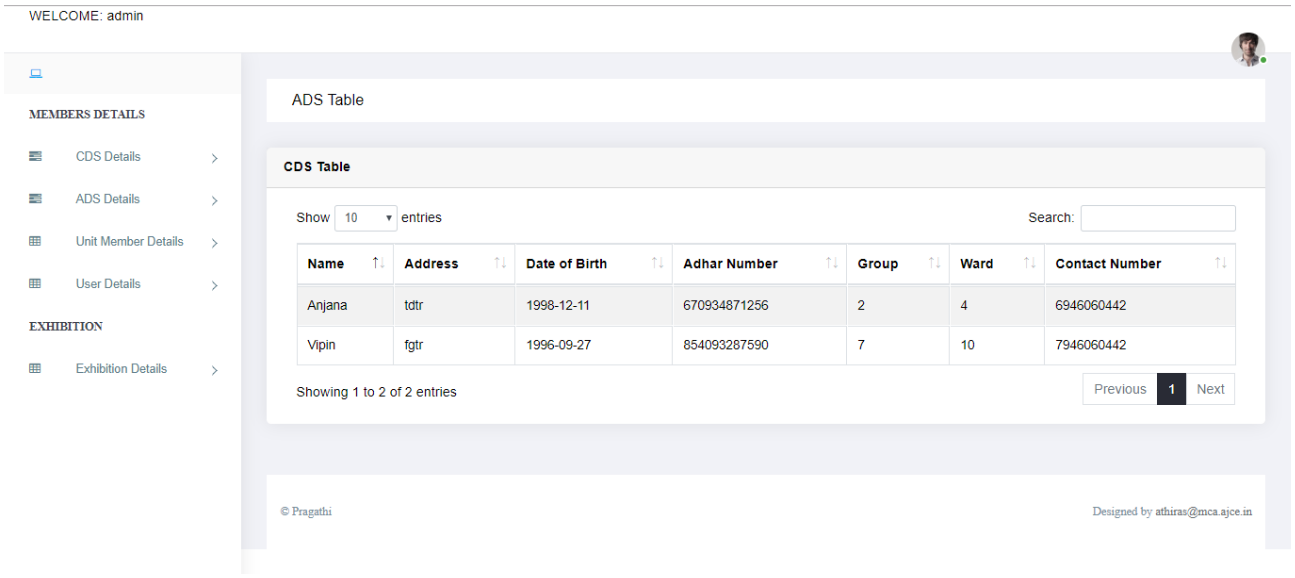
DATE OF BIRTH
dd-mm-yyyy

CONTACT NUMBER

PASSWORD

CONFIRM PASSWORD

➢ Viewing Ads



WELCOME: admin

ADS Table

CDS Table

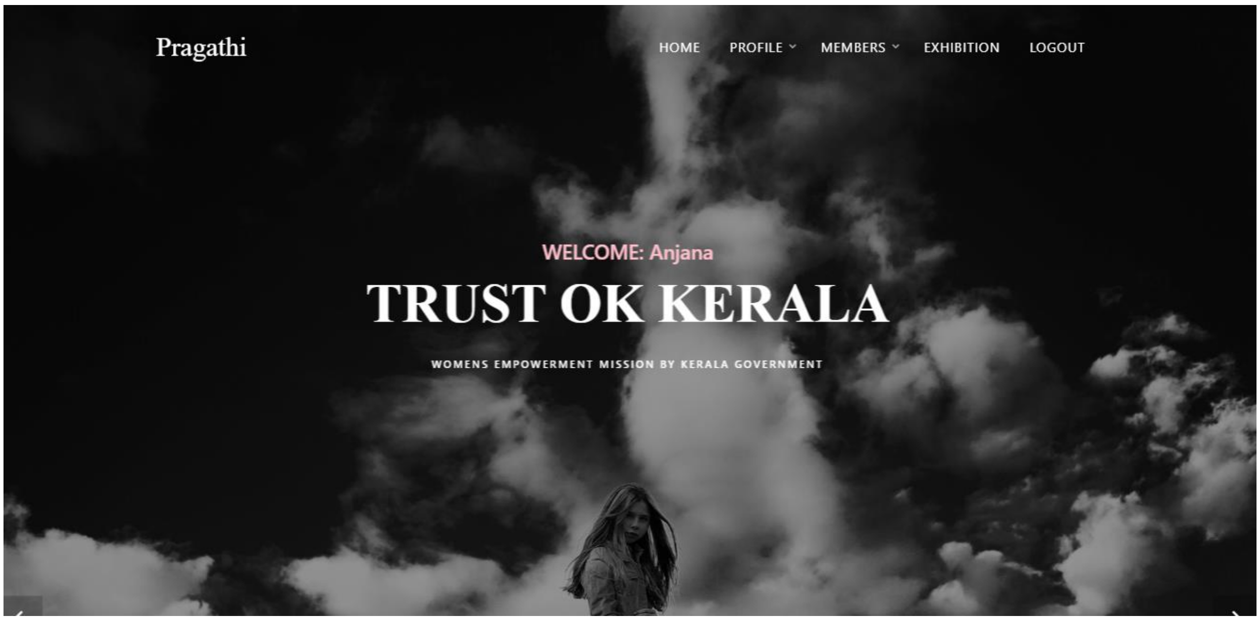
Show 10 entries Search:

Name	Address	Date of Birth	Adhar Number	Group	Ward	Contact Number
Anjana	tdtr	1998-12-11	670934871256	2	4	6946060442
Vipin	fgtr	1996-09-27	854093287590	7	10	7946060442

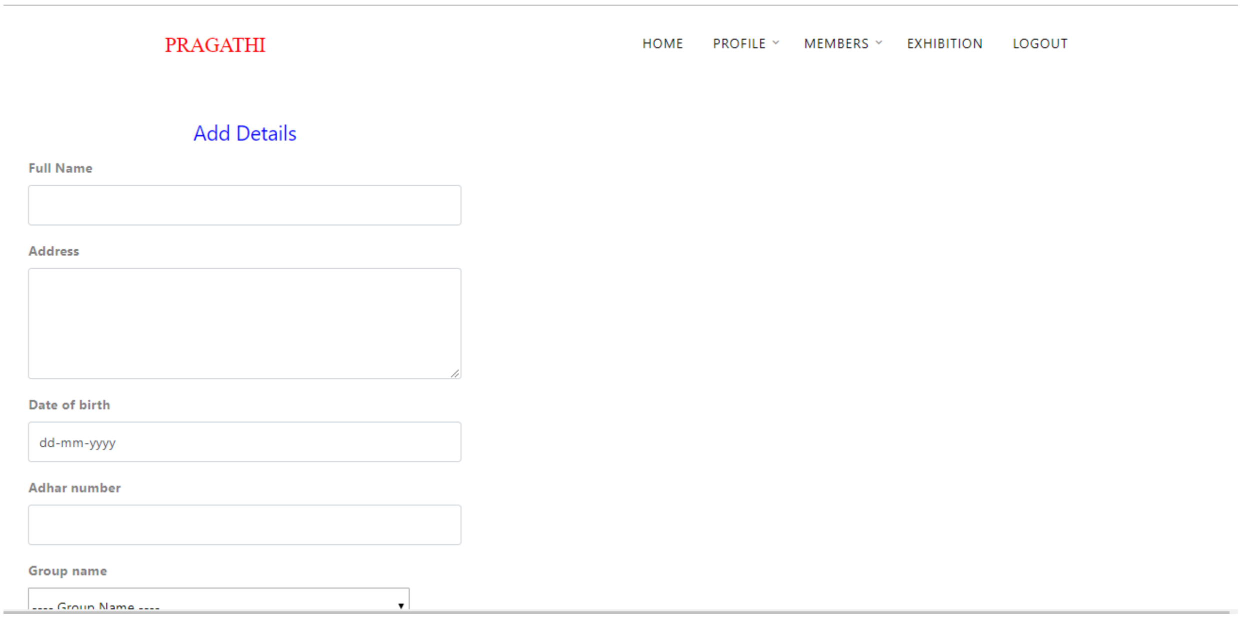
Showing 1 to 2 of 2 entries

© Pragathi Designed by athiras@mca.ajce.in

➢ Ads homepage



➢ Adding Unit members



Add Details

Full Name

Address

Date of birth

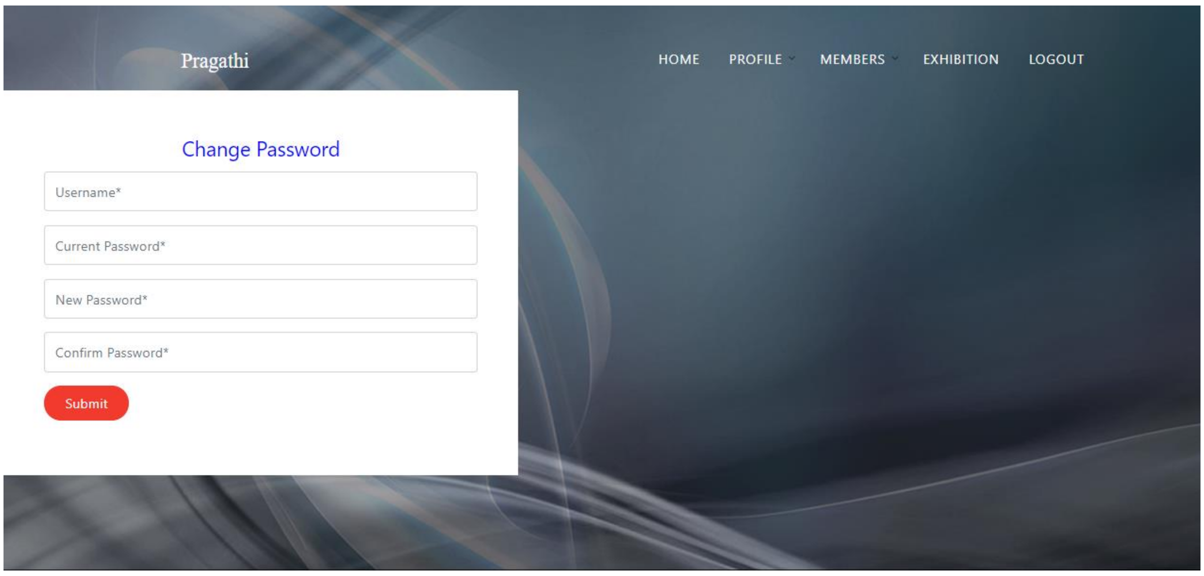
dd-mm-yyyy

Adhar number

Group name

---- Group Name ----

➢ Change password



Change Password

Username*

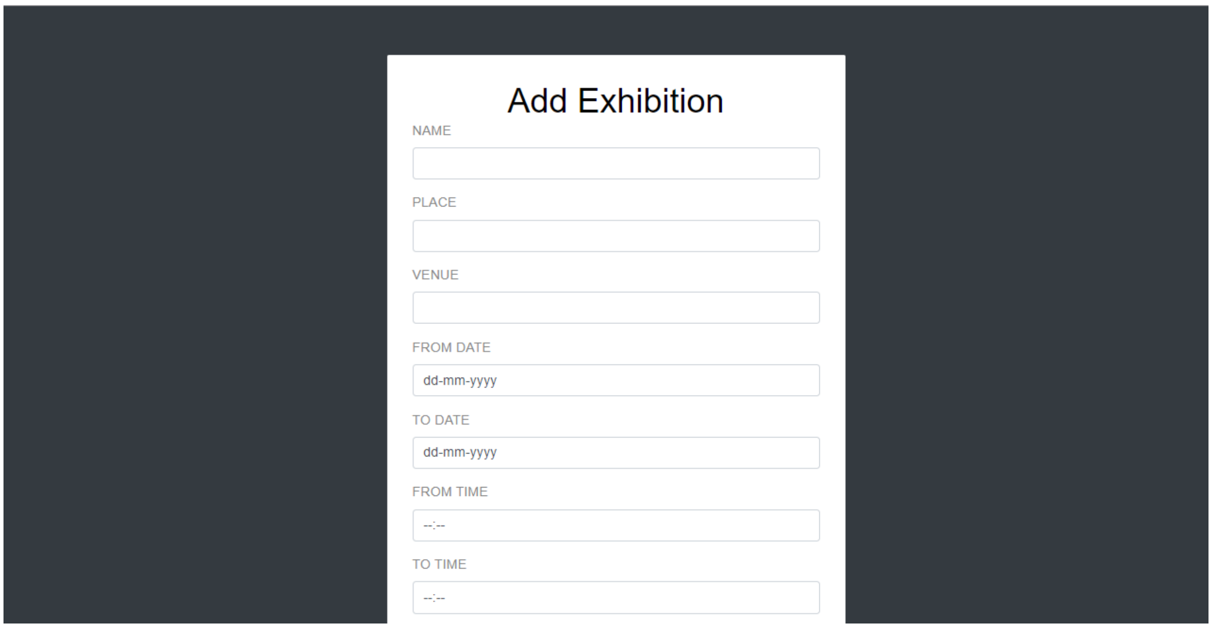
Current Password*

New Password*

Confirm Password*

Submit

➢ Adding Exhibition



Add Exhibition

NAME

PLACE

VENUE

FROM DATE

dd-mm-yyyy

TO DATE

dd-mm-yyyy

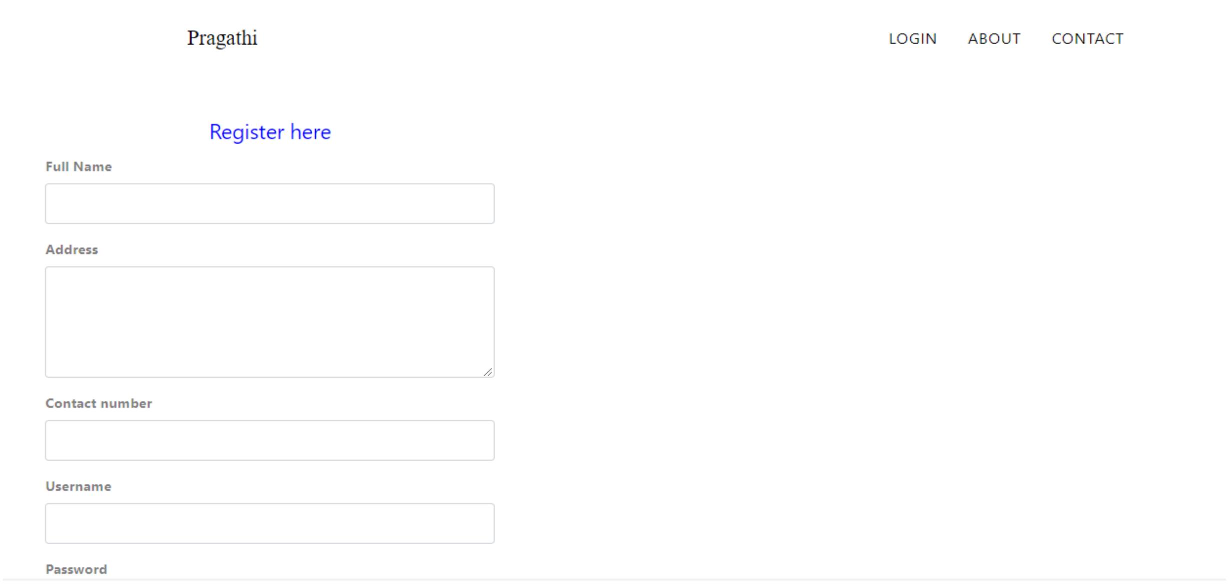
FROM TIME

--:--

TO TIME

--:--

➢ User Registration page



Pragathi

LOGIN ABOUT CONTACT

Register here

Full Name

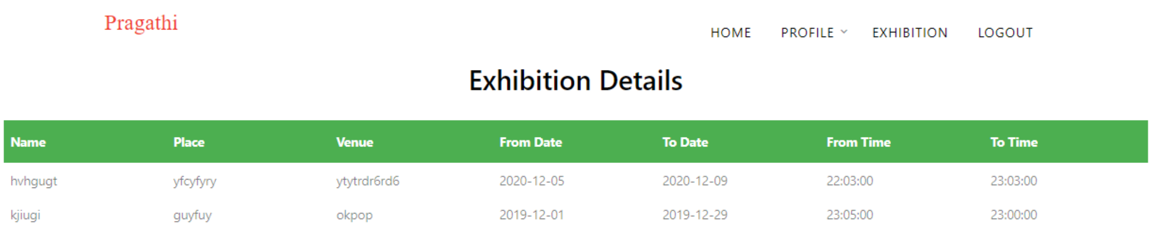
Address

Contact number

Username

Password

➢ User viewing exhibition



Exhibition Details

Name	Place	Venue	From Date	To Date	From Time	To Time
hvhgugt	yfcyfyry	ytytrdr6rd6	2020-12-05	2020-12-09	22:03:00	23:03:00
kjiugi	guyfuy	okpop	2019-12-01	2019-12-29	23:05:00	23:00:00