**ABSTRACT**

In the present system the information of teachers is manually written in logs. This is an extremely time consuming and labor intensive process as there is a need for a person to sit and update the log every time there is an update or change in any teachers records. Also at present, if the head of department wants to view data about a group of teachers or the whole department then there is no method other than to manually go through the logs and find the data.

Faculty Management System automates this system by making the teachers to keep their profiles updated themselves with the help of an online portal. The portal will be designed in an easy to use manner so that the easily register and update their information. The teacher can also update the information regarding him or her using the portal as and when required. The head of departments and the college can also view data about the teachers under them in a consolidated manner and also view just specific fields of all the teachers. Furthermore we will be providing the functionality to generate reports which will be giving pdf files that can be downloaded and viewed.

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
|  | **Name of Figure** | **Page No.** |
| Fig 2.1  Fig 3.2  Fig. 3.3  Fig. 3.4.1  Fig. 3.4.2  Fig. 4.1  Fig. 6.1  Fig. 6.2  Fig. 6.3  Fig. 6.4  Fig. 6.5  Fig. 6.6  Fig. 6.7  Fig. 6.8  Fig. 6.9 | Functional Requirement  System Architecture  Entity Relationship Diagram  Dataflow Diagram Level 0  Dataflow Diagram Level 1  Schema Diagram  SnapshotOf Login Page  Snapshot of Registration Page  Snapshot of Overview Page  Snapshot of insert record page  Snapshot of Update Page  Snapshot of Delete Page  Snapshot of Report generation Page  Snapshot of Report generated page  Snapshot of verification email | 6  9  10  11  12  13  21  21  22  22  23  23  24  24  25 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**LIST OF TABLES**

|  | **Name of Table** | **Page No.** |
| --- | --- | --- |
| Table 4.3  Table 6.1 Table 6.2 Table 6.3 | Cardinality Ratios.  Unit test case for FMS  Integration Test case for FMS  System Test case for FMS | 17  22  23  23 |

|  |
| --- |
|  |

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| `Sl. No. | Chapter Name | Page No. |
| 1.    2.  3.  4.  5.  6.  7. | Introduction   * 1. Purpose   2. Scope   3. Motivation   4. Literature Survey   Software Requirement Specification  2.1 Overall Description  2.2 Specific Requirements  2.2.1 Functionality  2.2.1.1 Functional Requirement –User  2.3.Non-functional Requirement    Project Design  3.1. Design Considerations  3.1.1 [Assumptions and Dependencies](http://www.cmcrossroads.com/bradapp/docs/sdd.html" \l "TOC_SEC7%23TOC_SEC7)  3.1.1.1 Assumptions  3.1.1.2 [Functional](http://www.cmcrossroads.com/bradapp/docs/sdd.html#TOC_SEC8%23TOC_SEC8) Dependencies  3.1.2 General Constraints  3.2 System Architecture  3.3 Database Design  3.3.1 Entity Relationship Diagram  3.3.2 Dataflow Diagram  3.3.3 Schema Diagram  3.3.4 Definition of Tables in Database  3.3.5 Definition of Relations and Cardinality Ratio of participating entities  Implementation  4.1 Client – Server specification  4.2 Technologies  Testing  5.1 Unit Testing  5.2 Integration Testing  5.3 System Testing  Results  6.1 Snapshots  6.2 Advantages of the Project  6.3 Limitations of the Project  Conclusion  7.1 Future Enhancement  References  Appendix A List Of Acronyms  Appendix B Coding | 1  1  1  2  3  3  3  4  5  6  6  6  7  8  9  10  12  16  18  19  19  22  23  24  25  29  29  30  31  32  33 |
|  |  |  |

# Chapter 1

# Introduction

**1.1 Purpose:**

The purpose of developing this site is to make the teachers able to enter their details and manage them as it suits to them. The site aims to provide automated report generation for all sorts of details related to their academic qualification or professional areas such as awards won, books authored, journals published etc.

**1.2 Scope:**

The project focuses on teaching and administrative staff in the campus. This facilitates and eases the amount of paperwork done by the staff in the college. The web site will provide all the teachers in the college to enter, update, view and delete their professional and academic details. This is also scaled to various departments in the college. It can also provide the admin with the facility of generating reports of faculties belonging to different departments and specific department when chosen.

**1.3 Motivation:**

In the present system the information of teachers is manually written in logs. This is an extremely time consuming and labor intensive process as there is a need for a person to manage the logs every time, there is an update or change in any teachers’ records.  Also at present, if the head of department wants to view data about a group of teachers or the whole department then there is no method other than to manually go through the logs and find the data.

We wish to automate this system by making the teachers to keep their profiles updated themselves with the help of an online portal. The portal will be designed in an easy to use manner so that they can register on the portal and manage their information. The head of departments and the administrative staff can also view data about the teachers under them in a consolidated manner and also view specific fields of all the teachers. Furthermore we will be providing the functionality to generate reports which will be giving pdf files that can be downloaded and saved.

**1.4 Literature Survey:**

From the early 18th century till date, there have been many visionaries, many educationists who all have one goal, the goal to make their education institute big worldwide, so that everyone can take their resources easily. From many years there is been technological advancement in every field, so as in web, we can access the whole world from our desk by just having internet.

So the aim is to provide a portal for faculties to keep the data structured and make life easier for the faculties and the administrative staff. The portal simplifies the operations of inserting, updating and searching of records.

Now the topic of reliability comes into picture, the user should get some satisfaction on which they can rely on; the module is providing ratings which can be used for comparing between different sources.

Regular analysis by the concerned people has given dimensions to the project. The requirements given by the authority has been given the utmost priority and the project fulfills all the requirements. The user interface has been made user friendly and simple for the easy access of the user.

# Chapter 2

# SOFTWARE REQUIREMENT SPECIFICATIONS

**2.1 Overall Description**

The proposed system has been designed to help the faculties to store their academic and other information. It also helps the teachers to create and update their details. The data of the faculty is secured at every instance of time. The admin can also manage the accounts of every faculty.

**2.2 Specific Requirements**

The requirements of the faculty management system are to develop:

| RESPONSIBILITIES (INTERNAL REQUIREMENTS) |
| --- |
| * Communication Interfaces   Communication between database and front end pages is through PHP, which is auto configured application for virtual server in a computer which is automatically configured. Lampserver is a linux web development environment which allows the user to create web applications with apache2, php and mysql database. Alongside, phpmyadmin also allows managing databases with the help of GUI easily. It is connected with front end with a connection string used in HTML pages.  [ Proposed, Medium difficulty ] |
| Hardware Requirements   * Operating System: Windows XP, Windows Vista, Windows 7, Windows 8 * Processor: Pentium 3.0 or higher * RAM: 256 Mb or more * Hard Drive: 300MB or more   [ Proposed, Medium difficulty ] |
| Software Requirements   * Front End: HTML5, CSS, Javascript, AJAX, Jquery, JSON * Backend: PHP, Mysql * Browser: Any modern browser that supports AJAX * OS – Linux or Windows   [ Proposed, Medium difficulty ] |
| User Interfaces  The database designed should be very easy to use and user friendly.   * Admin can see and modify the details of the faculties. * One user can not edit the information of another user. * A user can retrieve information of oneself. * Admin can retrieve information of all the users.   [ Proposed, Medium difficulty ] |

**2.2.1 Functionality**

The two main user groups and access levels are:

* Faculty (restricted access level)
* Admin (privileged access level)

Therefore, the requirements could be efficiently analyzed depending on the user group and the functionalities they should be allowed to perform.

# 2.2.1.1 Functional Requirement –User



Fig. 2.1 Functional Requirements

### Features

#### Administrator's features

* Admin should be able to generate reports of all faculties.
* Admin should be able to generate reports for all faculties in all departments.
* Admin should be able to generate department-wise reports.
* Admin should be able to change his/her password.

#### User's features

* User should be able to login and register himself/herself to the portal.
* User should be able to change password.
* User should be able to add documents pertaining to given information.
* User should be able to overview the details of each category.
* User should be able to update the previous data.
* User should be able to filter on the basis of the given fields and then delete the records.
* User should also be able to filter the records in order to generate reports in pdfs.
* User should be able to generate customized resume, with categories he chooses, between the given dates.
* User should be able to upload images of awards won, conference attended and papers or journals published.
* User should also be able to upload merit cards for his/her qualification details.

.

## 2.3 Non functional Requirements

**2.3.1 Performance Requirement**

* The PCs used must be atleast penangalo machines so that they can give optimum performance on product.
* In addition to these requirements, the system should also embrace the following requirements
* **Reliability** - The system should have little or no downtime and be able to handle multiple concurrent users.
* **Ease of use** - The general and administrative views should be easy to use and intuitive. Online help and documentation should be provided.
* **System and browser compatibility testing -** The system should be accessible on the following browsers – Microsoft Internet Explorer 7.0+, Netscape Navigator 6.0+ and Mozilla 1.3+, Google Chrome

**2.3.2 Security Requirements**

It is of utmost importance to ensure that access to the database is restricted and the information is provided on “Need to Know” basis. Users are provided with a unique User ID, Username and corresponding Password to authenticate their access levels. Similarly, only by having a faculty\_id the admin can oversee the various data which has been created by all the faculties. Each client has a password with which he can access his account. The email id provided by the user at the time of registration is checked for validity by sending an activation mail to that account. Password recovery is also provided which is sent to the user's registered mail id.

# Chapter 3

**PROJECT DESIGN**

**3.1 Design Considerations**

**3.1.1 Assumptions and Dependencies**

**3.1.1.1 Assumptions**

* Every faculty would register into the portal.
* The ids for the documents are generated on the order of insertion.
* The faculties registered are mailed with the activation link.
* The faculties should follow the activation link to activate their account.
* All the forms have a drop down button allowing the user to select only the valid id numbers.
* The data respective to the faculty can be modifies by the faculty. Admin can modify any data.

**3.1.1.2 Functional Dependencies**

**3.1.2** [**General Constraints**](http://www.cmcrossroads.com/bradapp/docs/sdd.html#TOC_SEC8%23TOC_SEC8)

The design involves the production of technical and visual prototypes. This stage has some non-technical aspects such as gathering of web content; content can be one of the biggest problems in web projects. For the server side programming and other technical aspects of the design emphasis will be laid on such design concepts. The goal is to make the system easier to adapt, enhance, test and use.

Some of the general constraints are:

* **Clarity of the information**

Each information inserted in the project shall be clear, without ambiguity. Each faculty has unique information and faculty\_id that should be stored and retrieved as and when required.

* **Structuring**

For guaranteeing reusability of data and its information for different views and layouts the structuring of da ta and separation of content, layout, and structure should be supported in

future.

* **Verifiability of the information**

Each item of information inserted in the project shall be verifiable.

* 1. **System Architecture**

Image2

**Fig: 3.2 System Architecture**

**3.4 Database Design**

**3.4.1 Entity Relationship Diagram**

# 3.4.2 Data Flow Diagram

**3.4.2.1 Level 0 Diagram**

**C:\Documents and Settings\Administrator\Desktop\Image2.EMF**

**Fig:3.4.1 dataflow diagram level 0**

* The User sends a request which is to be authenticated which if allowed for that type of user and if that kind of request exists is allowed and converted into a query.
* If that requests is not authenticated, a response to the GUI or user end is generated for eg: validations on fields.
* Query hence is executed on the server side and an output is generated and displayed to the user.

**3.4.2.2 Level 1 Diagram**

**C:\Documents and Settings\Administrator\Desktop\Image2.EMF**

**Fig:3.4.2 dataflow diagram level 1**

* The user enters the credentials.
* The credentials are checked and verified against the user registration.
* If the user is not registered, he is not authenticated.
* The appropriate view is displayed based on the access level of the user.

**3.4.2.3 Level 2 Diagram**

**C:\Documents and Settings\Administrator\Desktop\Image2.EMF**

**Fig3.4.3: Data Flow Diagram level 2**

* The user, once he is authenticated, is presented with various views that he allows him to update and change data.
* The user also has views where he can overview the data and generate the report.

**3.4.2.4 Level 3 Diagram**

**C:\Documents and Settings\Administrator\Desktop\Image2.EMF**

**Fig3.4.4: Data Flow Diagram level 3**

* The user is authenticated**.**
* Once the user is authenticated, he is validated and presented the insertion screen.
* There are various tables as mentioned in the DFD, in which user can insert data.
* Once the data is collected from the user, it is then inserted into mysql database.

**C:\Documents and Settings\Administrator\Desktop\Image2.EMF**

**Fig:3.4.5: Data Flow Diagram level 3**

* The user is authenticated**.**
* Once the user is authenticated, he is validated and presented the updation screen.
* There are various tables as mentioned in the DFD, in which user can update data.
* Once the data is collected from the user, it is then updated in mysql database.

**C:\Documents and Settings\Administrator\Desktop\Image2.EMF Fig 3.4.6: Data Flow Diagram level 3**

* The user is authenticated**.**
* Once the user is authenticated, he is validated and presented the deletion screen.
* There are various tables as mentioned in the DFD, in which user can delete data.
* Once the data is collected from the user, it is then deleted from mysql database.

**C:\Documents and Settings\Administrator\Desktop\Image2.EMF**

**Fig3.4.7: Data Flow Diagram level 3**

* The user is authenticated**.**
* Once the user is authenticated, he is validated and presented with the screen where he can specify filters to generate report.
* There are various tables as mentioned in the DFD, in which user can query information.
* Once the data is collected from the database, the user is presented with a report of the queried information.

**3.4.3 Schema Diagram**

**3.4.4 Definition of tables in the database**

CREATE TABLE IF NOT EXISTS `award` (

`id` varchar(10) NOT NULL,

`award\_id` int(11) NOT NULL,

`award\_name` varchar(80) NOT NULL,

`award\_agency` varchar(80) NOT NULL, `url` varchar(40) DEFAULT NULL,

`remark` varchar(100) DEFAULT NULL,

`file` varchar(250) DEFAULT NULL,

`award\_date` date DEFAULT NULL,

PRIMARY KEY (`id`,`award\_id`),

KEY `aid` (`award\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE IF NOT EXISTS `book` (

`book\_id` int(11) NOT NULL,

`role` varchar(20) NOT NULL,

`book\_or\_chapter` varchar(20) NOT NULL,

`book\_title` varchar(20) NOT NULL,

`book\_edition` varchar(20) NOT NULL,

`publisher\_name` varchar(20) NOT NULL,

`isbn` varchar(25) NOT NULL,

`id` varchar(10) NOT NULL,

`book\_date` date DEFAULT NULL, PRIMARY KEY (`book\_id`,`id`),

UNIQUE KEY `isbn` (`isbn`),

KEY `id` (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE IF NOT EXISTS `book\_authors` (

`id` varchar(10) NOT NULL,

`book\_id` int(11) NOT NULL,

`author\_number` int(11) NOT NULL,

`author\_name` varchar(40) NOT NULL,

PRIMARY KEY (`id`,`book\_id`,`author\_number`),

KEY `book\_authors\_ibfk\_2` (`book\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE IF NOT EXISTS `book\_chapters` (

`id` varchar(10) NOT NULL,

`book\_id` int(11) NOT NULL,

`chapter\_name` varchar(50) NOT NULL,

PRIMARY KEY (`book\_id`,`id`,`chapter\_name`),

KEY `book\_chapters\_ibfk\_2` (`book\_id`),

KEY `id` (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE IF NOT EXISTS `communityuser` (

`community\_user\_id` int(11) NOT NULL,

`name\_of\_event` varchar(40) NOT NULL,

`role` varchar(40) DEFAULT NULL,

`location` varchar(40) DEFAULT NULL,

`date\_of\_event` date DEFAULT NULL,

`url` varchar(40) DEFAULT NULL,

`additional\_information` varchar(40)

DEFAULT NULL,

`id` varchar(10) NOT NULL,

PRIMARY KEY (`community\_user\_id`,`id`),

KEY `id` (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE IF NOT EXISTS `conference` (

`id` varchar(10) NOT NULL,

`conference\_id` int(11) NOT NULL, `conference\_type` varchar(100) NOT NULL,

`research\_area` varchar(100) NOT NULL,

`paper\_associated\_project` varchar(5) NOT NULL,

`project\_name` varchar(100) DEFAULT NULL,

`conference\_name` varchar(200) NOT NULL,

`organizer` varchar(100) DEFAULT NULL,

`from\_date` date NOT NULL,

`to\_date` date NOT NULL,

`venue` varchar(100) NOT NULL,

`abstract` varchar(500) NOT NULL,

`keywords` varchar(200) NOT NULL,

`url` varchar(100) DEFAULT NULL,

`file` varchar(500) NOT NULL,

`from\_page` int(11) DEFAULT NULL,

`to\_page` int(11) DEFAULT NULL,

`paper\_title` varchar(100) DEFAULT NULL,

PRIMARY KEY (`id`,`conference\_id`),

KEY `ctype` (`conference\_type`),

KEY `cid` (`conference\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE IF NOT EXISTS `conference\_authors` (

`id` varchar(10) NOT NULL,

`conference\_id` int(11) NOT NULL,

`author\_name` varchar(40) NOT NULL,

PRIMARY KEY (`id`,`conference\_id`,`author\_name`),

KEY `con\_id` (`conference\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE IF NOT EXISTS `consultancy` (

`consultancy\_id` int(11) NOT NULL,

`client` varchar(80) NOT NULL,

`work\_title` varchar(80) NOT NULL,

`start\_date` date NOT NULL,

`end\_date` date NOT NULL,

`revenue\_generated` varchar(15) NOT NULL,

`summary` varchar(100) DEFAULT NULL,

`labs\_allocated` varchar(100) DEFAULT NULL,

`url` varchar(40) DEFAULT NULL,

`id` varchar(10) NOT NULL DEFAULT '',‘’ PRIMARY KEY (`consultancy\_id`,`id`), KEY `id` (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE IF NOT EXISTS `consultancy\_collaboration` (

`id` varchar(10) NOT NULL,

`consultancy\_id` int(11) NOT NULL,

`collaborator\_name` varchar(40) NOT NULL,

PRIMARY KEY (`id`,`consultancy\_id`),

KEY `consultancy\_id` (`consultancy\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE IF NOT EXISTS `consultancy\_faculty\_involved` (

`id` varchar(10) NOT NULL,

`consultancy\_id` int(11) NOT NULL,

`fname` varchar(40) NOT NULL,

PRIMARY KEY (`id`,`consultancy\_id`,`fname`),

KEY `consultancy\_fac\_inv\_ibfk\_3`

(`consultancy\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE IF NOT EXISTS `consultancy\_student\_involved` (

`id` varchar(10) NOT NULL,

`consultancy\_id` int(11) NOT NULL,

`student\_usn` varchar(10) NOT NULL,

PRIMARY KEY (`id`,`consultancy\_id`,`student\_usn`),

KEY `consultancy\_id` (`consultancy\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE IF NOT EXISTS `courses\_list` (

`course\_id` varchar(8) NOT NULL,

`course\_name` varchar(40) NOT NULL,

`semester` int(11) NOT NULL,

`ug/pg` varchar(2) NOT NULL,

`syllabus\_year` year(4) NOT NULL,

PRIMARY KEY (`course\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE IF NOT EXISTS `courses\_taught` (

`course\_id` varchar(8) NOT NULL,

`academic\_year` year(4) DEFAULT NULL,

`number\_of\_students` int(4) DEFAULT NULL,

`pass\_percent` decimal(5,2) NOT NULL,

`id` varchar(10) NOT NULL DEFAULT '',‘’ PRIMARY KEY (`id`,`course\_id`),

KEY `course\_id` (`course\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE IF NOT EXISTS `faculty\_exchange\_program` (

`id` varchar(10) NOT NULL,

`institution` varchar(40) NOT NULL,

`subject` varchar(20) NOT NULL,

`start\_date` date NOT NULL,

`end\_date` date NOT NULL,

`ug\_pg` varchar(20) NOT NULL,

`collaboration\_type` varchar(20) NOT NULL,

`details\_of\_collaboration` varchar(20)

DEFAULT NULL,

`feedback` text NOT NULL,

PRIMARY KEY (`id`) ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE IF NOT EXISTS `faculty\_member` (

`fid` varchar(10) NOT NULL,

`username` varchar(50) NOT NULL, `password` varchar(60) NOT NULL,

`name` varchar(30) NOT NULL,

`address` varchar(100) NOT NULL,

`phone\_number` varchar(30) NOT NULL,

`picture` varchar(100) NOT NULL, `gender` varchar(100) NOT NULL,

`email` varchar(30) NOT NULL,

`date\_of\_join` date NOT NULL,

`date\_of\_birth` date DEFAULT NULL,

`department` varchar(3) DEFAULT NULL,

PRIMARY KEY (`fid`,`username`),

KEY `fid` (`fid`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1

;

**3.4.5 Definition of relations and the cardinality ratios of the participating entities**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl. No.** | **Participating**  **Entity 1** | **Relation** | **Participating**  **Entity 2** | **Cardinality Ratio** |
| 1. | Faculty | Has | Profile | 1:1 |
| 2. | Faculty | Has won | Awards | M:N |
| 3. | Faculty | Attended/Conducted | Conference | M:N |
| 4. | Faculty | Part of/Conducted | Projects | M:N |
| 5. | Faculty | Was Part of | Faculty\_Exchange | M:N |
| 6. | Faculty | Authored | Book/Journals | M:N |
| 7. | Faculty | Teaches | Courses | M:N |

**Table 3.4** Cardinality Ratios

# Chapter 4

**IMPLEMENTATION**

**4.1 Client-Server Specification**

The software runs on Client Server Architecture where the server side consists of php scripts running on the server. The server can be run on windows as well as linux based systems. It works best with Apache server 2.4, php 5.5.9 and Mysql server 5.5.xx installed on the system.

The clients can use any AJAX compatible browser. Further specifications for the browsers are listed below:

Mozilla Firefox – Version 31

Google Chrome – Version 44

Having the above mentioned versions of the browsers at the client side enables the use of AJAX and JQuery which is used for the development of the software.

The software developed has fat server architecture, which implies that the clients only need to have minimal configuration for their systems.

**4.2 Technologies**

The various technologies used for development purposes are:

**4.2.1 HTML5**

* HTML5 is a markup language used for structuring and presenting content on the World Wide Web. It was finalized, and published, on 28 October 2014 by the World Wide Web Consortium.
* Its core aims are to improve the language with support for the latest multimedia while keeping it easily readable by humans and consistently understood by computers and devices.
* Various new APIs have been included :- Geolocation, Local Storage, etc.

**4.2.2 Javascript**

* JavaScript is a lightweight, interpreted programming language. It is designed for creating network-centric applications.
* JavaScript is very easy to implement because it is integrated with HTML. It is open and cross-platform.
* Client-side JavaScript is the most common form of the language. The script should be included in or referenced by an HTML document for the code to be interpreted by the browser.
* JavaScript can be used to trap user-initiated events such as button clicks, link navigation, and other actions that the user initiates explicitly or implicitly.
* We can create interfaces that react when the user hovers over them with a mouse or activates them via the keyboard.

**4.2.3 JQuery**

* JQuery is a fast and concise JavaScript library created by John Resig in 2006. jQuery simplifies HTML document traversing, event handling, animating, and Ajax interactions for Rapid Web Development.
* The JQuery offers an elegant way to capture a wide variety of events, such as a user clicking on a link, without the need to clutter the HTML code itself with event handlers.
* The JQuery helps you a lot to develop a responsive and feature-rich site using AJAX technology.
* JQuery is very lightweight library - a–out 19KB in size.

**4.2.4 AJAX**

* Ajax is a set of web development techniques utilizing many web technologies used on the client-side to create asynchronous Web applications.
* With Ajax, web applications can send data to and retrieve from a server asynchronously (in the background) without interfering with the display and behavior of the existing page.
* Ajax is not a technology, but a group of technologies. HTML and CSS can be used in combination to mark up and style information. The DOM is accessed with JavaScript to dynamically display – and allow the user to interact with – the information presented.

**4.2.5 XML**

* XML stands for E**X**tensible **M**arkup **L**anguage.
* XML is designed to store and transport data.
* XML is designed to be self-descriptive.
* XML is designed to be both human- and machine-readable.
* XML is a W3C Recommendation.

**4.2.6 CSS**

* Cascading Style Sheets, fondly referred to as CSS, is a simple design language intended to simplify the process of making web pages presentable.
* CSS saves time.
* Pages load faster.
* Easy maintenance - T– make a global change, simply change the style, and all elements in all the web pages will be updated automatically.
* Offline Browsing - C–S can store web applications locally with the help of an offline catche.Using of this, we can view offline websites.The cache also ensures faster loading and better overall performance of the website.
* Platform Independence - T–e Script offer consistent platform independence and can support latest browsers as well.

**4.2.7 JSON**

* JSON or JavaScript Object Notation is a lightweight text-based open standard designed for human-readable data interchange.
* It is used while writing JavaScript based applications that includes browser extensions and websites.
* It is primarily used to transmit data between a server and web applications.
* Web services and APIs use JSON format to provide public data.
* It can be used with modern programming languages.

**4.2.8 PHP**

The programming language used for the development work is PHP. The reason for selection of this language includes among many others the following few.

* Open Source, PHP is completely free.
* PHP can be easily embedded directly into HTML.
* Platform independent can run on Windows Linux or Mac servers.
* Run faster on the internet and easily integrate AJAX, Callback etc.
* Interfaces very easily with Apache/MySQL
* Lots of good books and on-line help.
* It's available with documentation in many languages.
* Easy to learn compared to many other scripting languages. It has a syntax that is easy to parse and is actually rather human-friendly.
* Lots of hosting services have it ready to use, no special configuration.Pretty easy to access other web-based tools through PHP i.e. google maps, etc

**4.2.9 MySQL**

**MySQLi** is a relational data base management system (RDBMS) that runs as a server providing multi-user access to a number of databases.

* MySQL is an open source tool.
* MySQL is a popular choice of database for use in web applications, and is a central component of the widely-used LAMP web application software stack — LAMP is an acronym for "Linux, Apache, MySQL, PHP.
* MySQL is primarily an RDBMS and therefore ships with no GUI tools to administer MySQL databases or manage data contained within.
* Microsoft SQL server Express Management studio provides an ease in creating tables by a graphical as well as query based interface.

# 

# Chapter 5

**TESTING**

Software Testing is the process used to help identify the correctness, completeness, security and quality of the developed computer software. Testing is the process of technical investigation and includes the process of executing a program or application with the intent of finding errors.

**Test Strategies**

Test strategy tells the test plan of the project. It also tells how to test and what to test. The testing done in this project are Unit testing and Integration testing.

* Features to be tested: Form navigation and generation of reports.
* Items to be tested: Functioning of forms and buttons.
* Purpose of testing: To check the effective working of FMS.
* Pass / Fail Criteria: Changes made on the back end like recreation of tables should affect the front end as well. If so, the test is successful.
* Assumptions and Constraints: Tables should be created and values have to be entered at the back end before testing and entity integrity and referential integrity constraints should be taken care

**5.1 Unit Testing**

Unit testing is a software verification and validation method in which a programmer tests if individual units of source code are fit for use. Some of the tests performed in the project are inserting, delete, retrieve and modify.

**Table 5.1 Unit test cases**

| **Name** | **Object** | **Test Type** | **Current Status** | **Description** | **Input** | **Acceptance Criteria** | **Last Run** | **Result Details** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 6 | WorkingSet1 | Standard | Fail | Unit case for search operation (heck the search button) | Keyword=USN  Upon mouse click on button. | Displaying all details with USN matching the Keyword. | Invalid DateTime | No record |
| 5 | WorkingSet1 | Standard | Pass | Unit test case for display operation(check the display button) | USN=”1RV11IS100” . Upon mouse click on button. | Data successfully retrieved from the database and displayed | Invalid DateTime | Data successfully retrieved from the database and displayed |
| 4 | WorkingSet1 | Standard | Pass | Unit test case for delete operation (Check the delete button) | Username=’admin’,Password=’admin’ Upon mouse click on button. | Visibility of done button | Invalid DateTime | Done Button Appears and profile is deleted |
| 3 | WorkingSet1 | Standard | Pass | Unit test case for update operation (testing of save button) | Password=’admin’,FN=’Bhagya’,LN=’Batra’,address=’#503 Rajajinagar banangaloremail\_id=’bhagya@vsnl.net’  Upon mouse click on button. | Appearance of Next Button | Invalid DateTime | Next Button Appears |
| 2 | WorkingSet1 | Standard | Pass | Unit test case for login information check operation | USERNAME=’admin’,PASSWORD=’isw’ Upon mouse click on button. | Login Button Appears if login information is correct if not “Invalid Username or Password” appears. | Invalid DateTime | Message: Invalid Username or Password |
| 1 | WorkingSet1 | Standard | Pass | This checks if validation happens while logging in and button appears or not | USERNAME=’admin’,PASSWORD=’admin’ Upon mouse click on button. | Login Button Appears if login information is correct if not “Invalid Username or Password” appears. | Invalid DateTime | Login Button appears. |

**5.2 INTEGRATION TESTS**

| **Name** | **Object** | **Test Type** | **Current Status** | **Description** | **Input** | **Acceptance Criteria** | **Last Run** | **Result Details** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | WorkingSet1 | Standard | Pass | Password recovery operation | Username=’Admin’ answer(code)=’rvce’ | Message: Original Password display | Invalid DateTime | Message: “admin”-actual password |
| 2 | WorkingSet1 | Standard | Pass | Generation of grid view | Click Search button after entering pattern in textbox | Display of grid view with Name,Contact | Invalid DateTime | Display of grid view with Name,Contact |
| 1 | WorkingSet1 | Standard | Pass | Integrated test for login navigation | Click login button on the login page after entering username and password | Move to Profile page. | Invalid DateTime | Move to Profile page. |

**Table 5.2 Integration test cases**

**5.3 System Testing**

| **Name** | **Object** | **Test Type** | **Current Status** | **Description** | **Input** | **Acceptance Criteria** | **Last Run** | **Result Details** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | WorkingSet1 | Standard | Pass | Overall system check | Data entered through forms | Updated data displayed in generated reports | Invalid DateTime | Updated data displayed in generated reports |

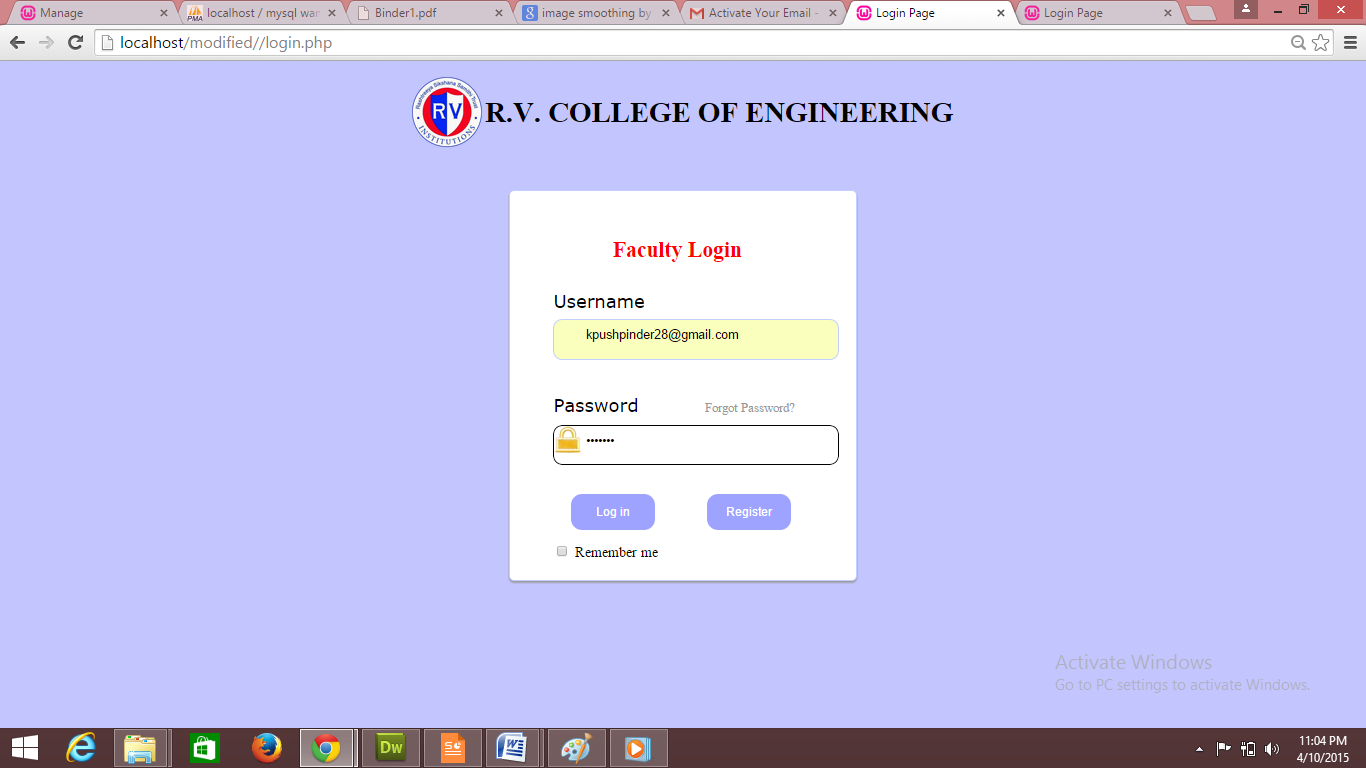
# Table 5.3 System test case

**Chapter 6**

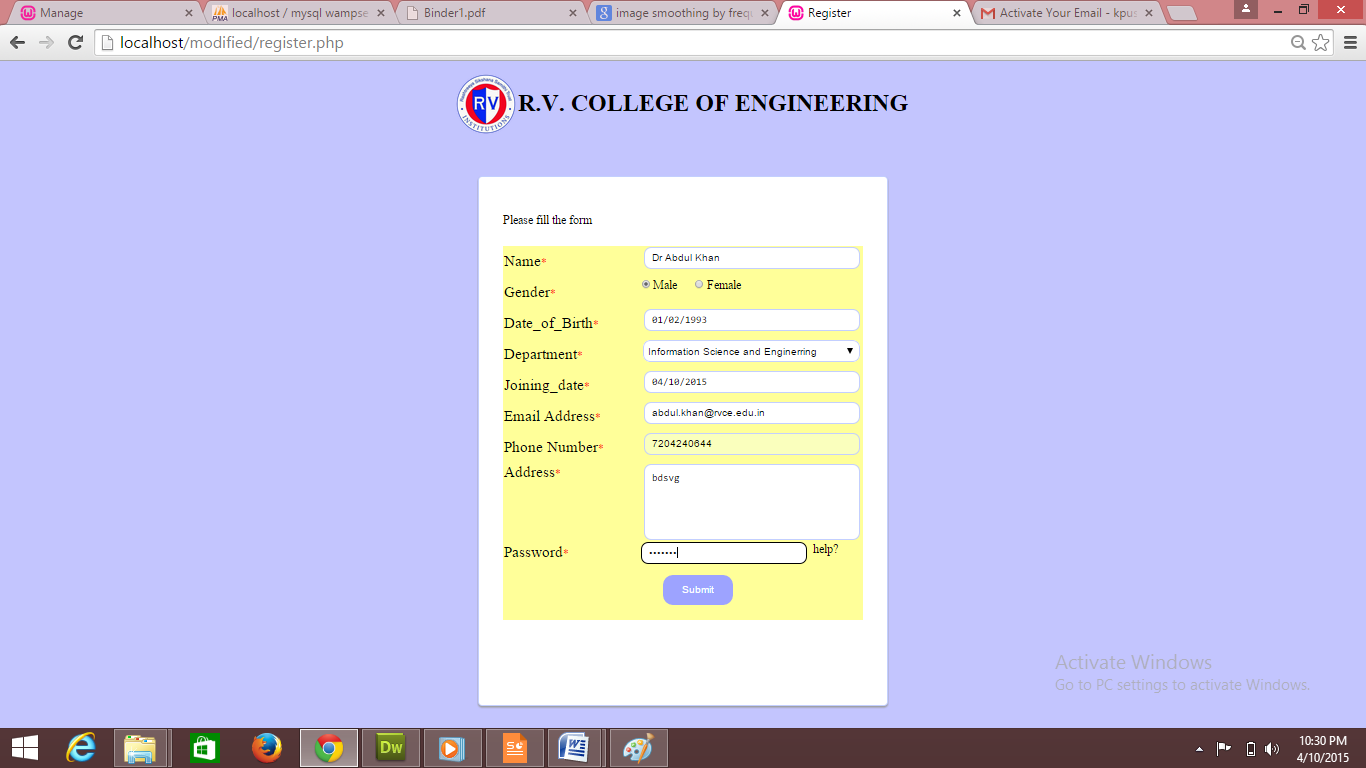
**RESULTS**

**6.1 Snapshots**

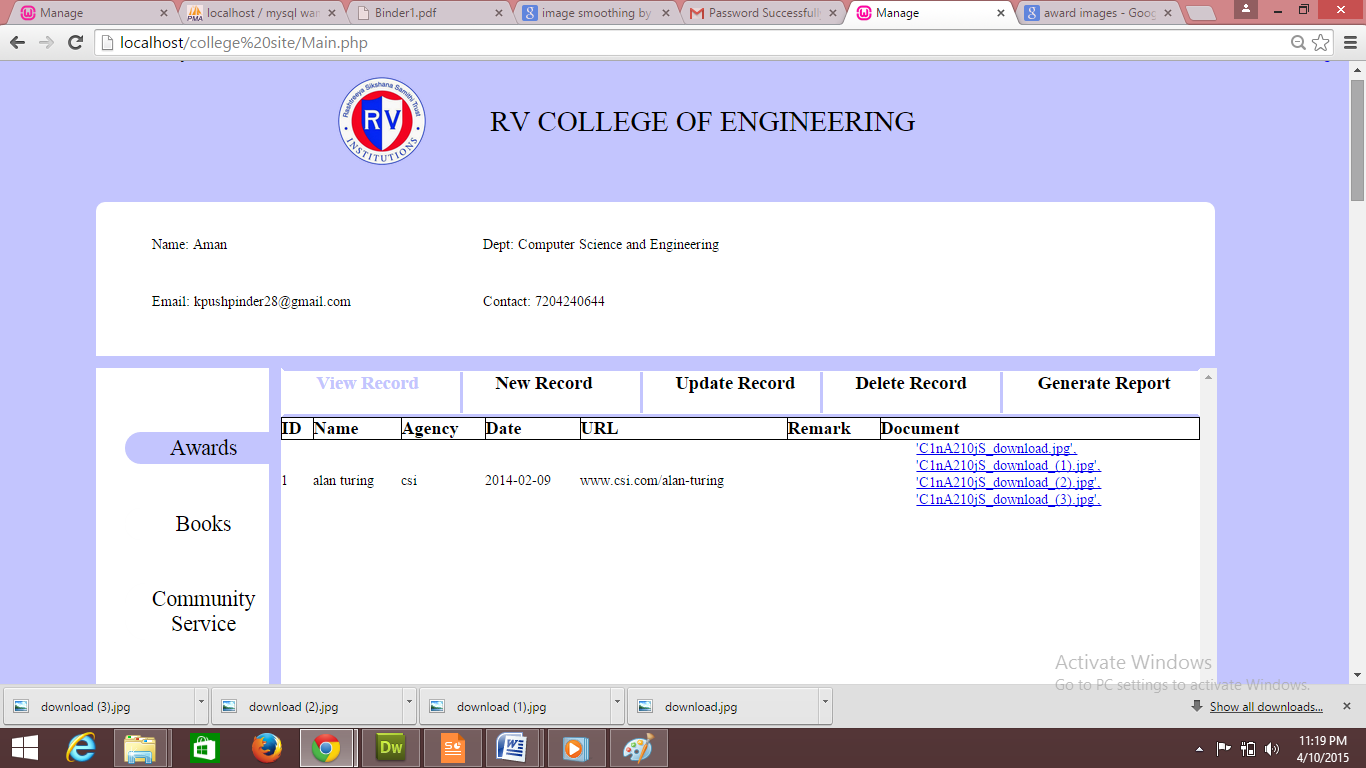
**Figure 6.1 Snapshot of Login page**



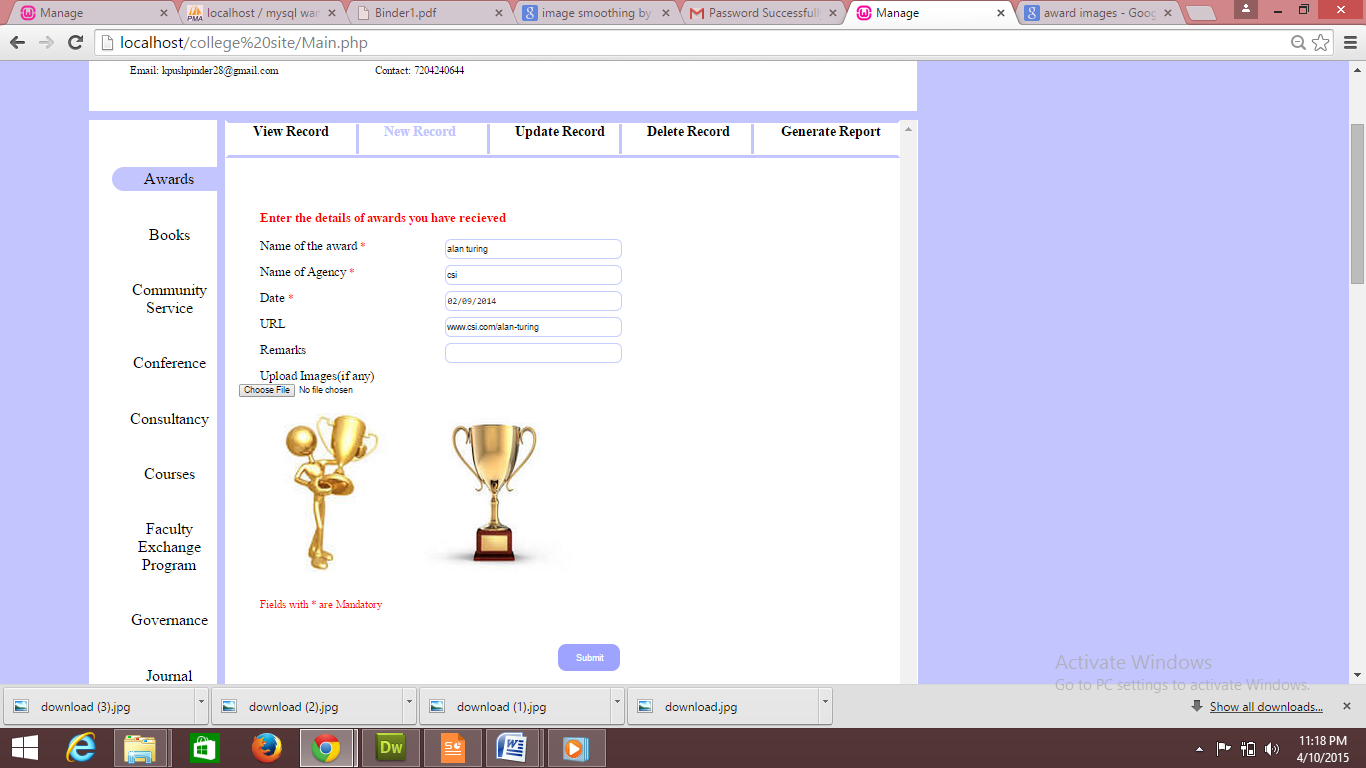
**Figure 6.2 Snapshot of Registration page**

****

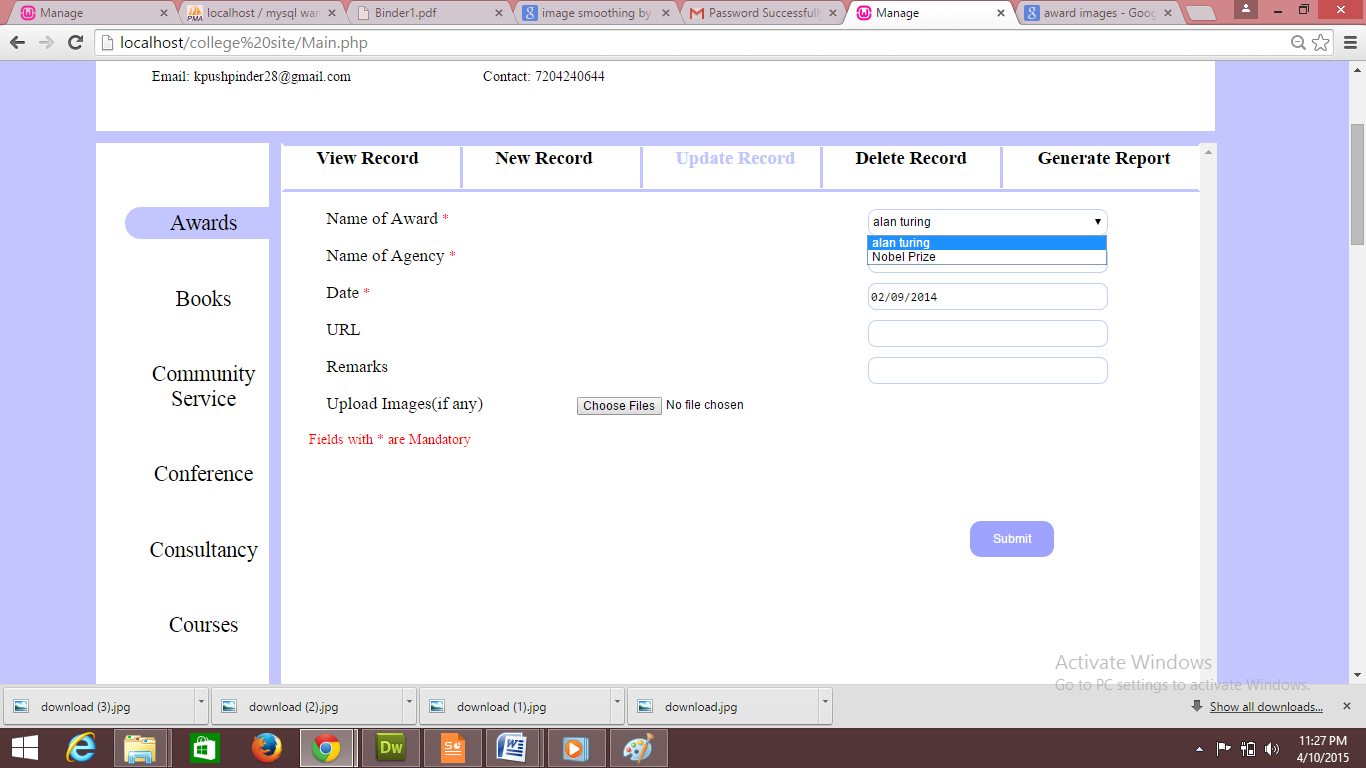
**Figure 6.3 Snapshot of overview page**



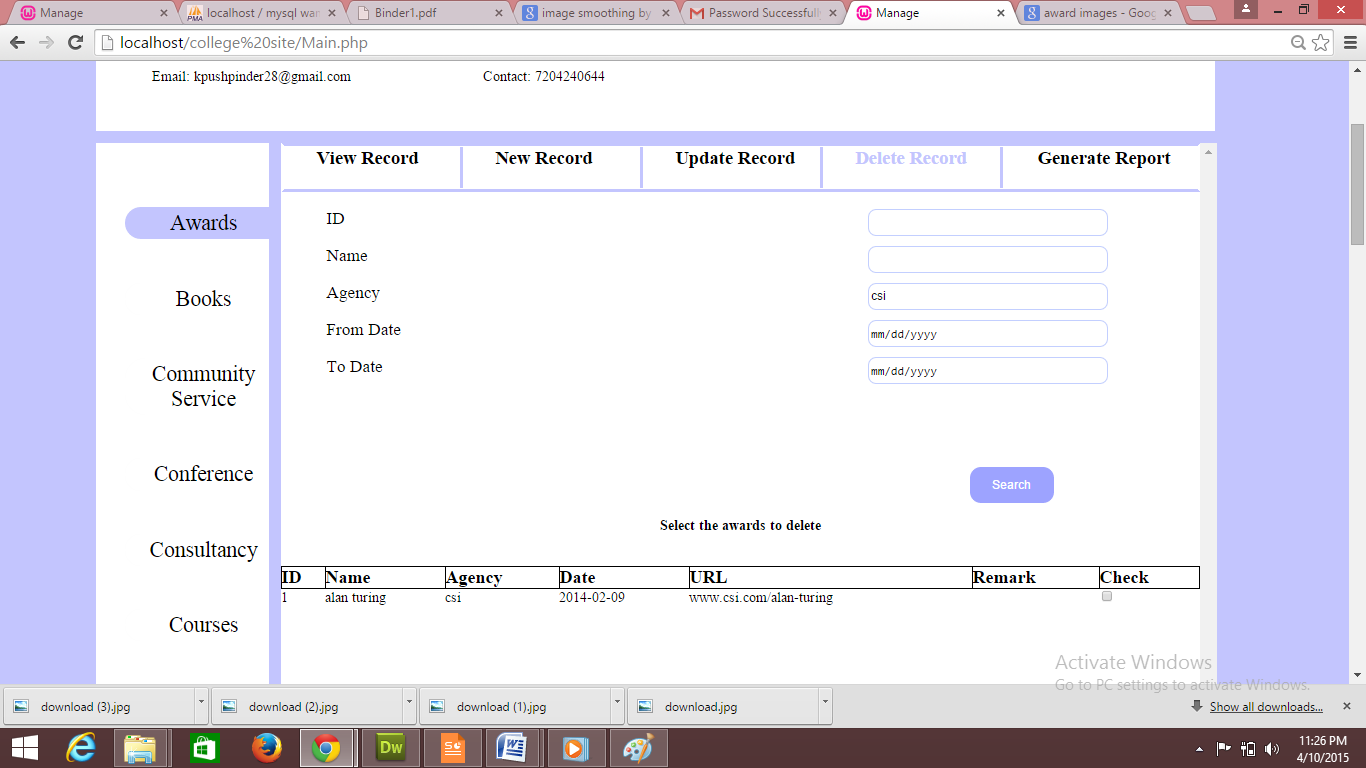
**Figure 6.4 Snapshot of insertion page**



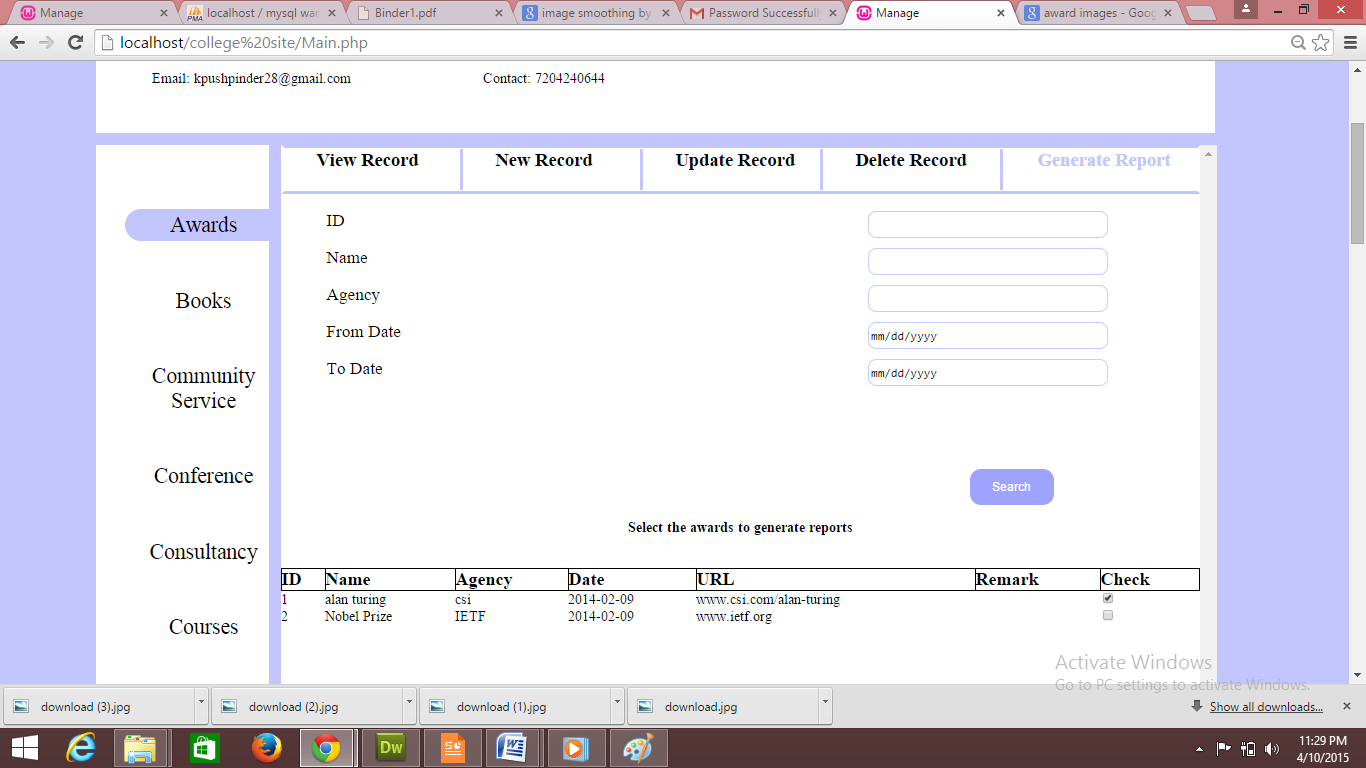
**Figure 6.5 Snapshot of update page**



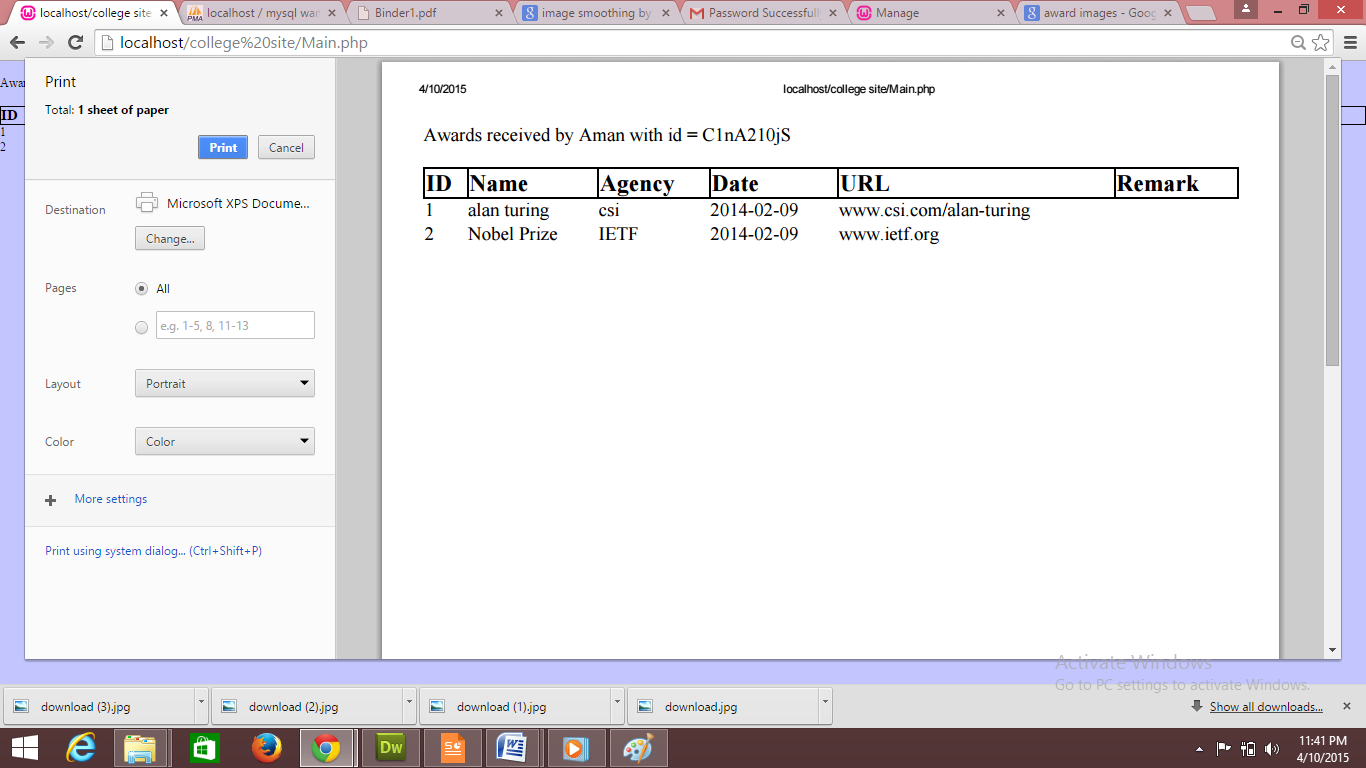
**Figure 6.6 Snapshot of deletion page**

****

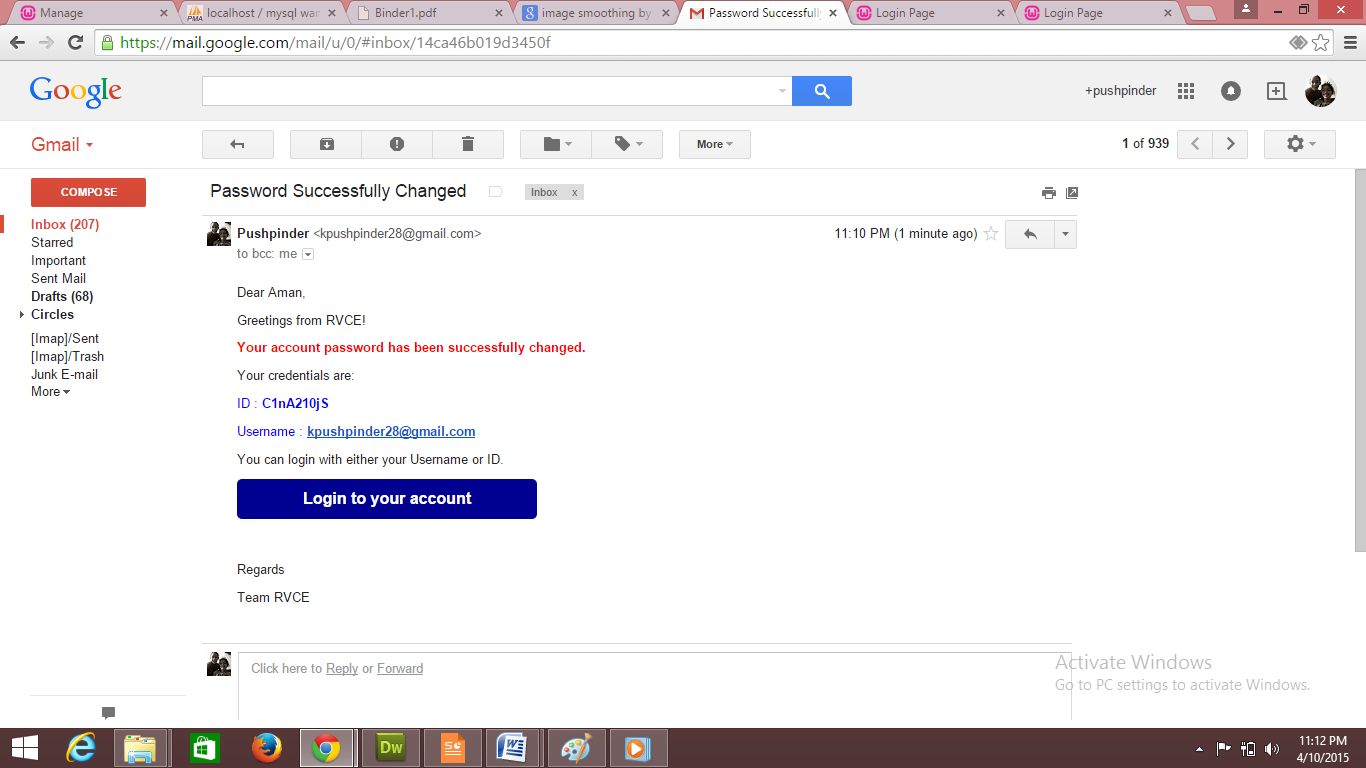
**Figure 6.7 Snapshot of report generation page**

****

**Figure 6.8 Snapshot of report generated**

****

**Figure 6.9 Snapshot of verification email**

****

**6.2 Advantages of the project**

This project is built to provide the following advantages.

* The FMP is created to keep track of all faculty data.
* FMP module provides the facility of username and password for each faculty so that they can easily access or modify their info without taking help of administrator.
* The FMP is one place where the faculty manage all their data
* The FMP module provides the Faculty to view their various acheivments.
* Verification mail is sent to verify the user
* FMP module also provides report generations.
* FMP allows the person to update the data once it is entered

**6.3 Limitations of the project**

The project has its own limitations

* FMP does not provide the functionality to generate customized reports.
* FMP doesn’t allow the user to manage their college transactions and salary.

**Chapter 7**

**CONCLUSION**

The Faculty Management for R V College of Engineering focuses on reducing the human efforts for details of the faculties of the college. Thus FMS module gives customized results all the time, as by giving the pdf of the selected data. The FMS module is tested and re-tested to ensure its effectiveness and provide error free functionality to end user. The AR module hence assures saving of time in finding resources in a appropriate way.

**7.1 Future Enhancement**

The Faculty Management module can overcome the previously stated limitations by adding few extra features to it such as

* FMS will advance in a spam control for non authorized users.

# 

# References

[1] Elmasri and Navathe, *Fundamentals of Database Systems,* 5th edition, Addison-Wesley, 2007.

[2] Raghu Ramakrishnan and Johannes Gehrke, *Database Management Systems*, 3rd edition,

McGraw-Hill, 2003.

[3] Silberschatz, Korth and Sudharshan, *Database System Concepts*, 5th Edition, McGraw-Hill 2006.

# [4] Eric T Freeman, Elizabeth Freeman, Elizabeth Robson, *Head First HTML with CSS & XHTML,* OReilly’

[5] C.J.Date, A.Kannan, S.Swamynatham, *A Introduction to Database Systems*, 8th edition,

Pearson Education, 2006.

[6] [www.w3schools.com](http://www.w3schools.com) for online tutorials on PHP, HTML, CSS, MySQL.

**Appendix-A**

**LIST OF ACRONYMS**

1. FMS: Faculty Management System
2. SRS: Software Requirement Specifications
3. ER: Entity Relationship
4. OS: Operating System
5. RAM: Random Access Memory
6. SQL: Structured Query Language
7. PHP: Hypertext Preprocessor(Personal Home Page)
8. RDBMS: Relational Database management system

**Appendix-B**

**CODING**

**Coding For Login Page**

<!DOCTYPE html>

<html>

<head>

<title>Admin Login</title>

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta charset="utf-8">

<link href="event.css" rel="stylesheet" type="text/css">

</head>

<body>

<br >

<div align=center><font color="FF0000">

<?php

session\_start();

$host="localhost"; // Host name

$username="root"; // Mysql username

$password=""; // Mysql password

$db\_name="studentadmission"; // Database name

$tbl\_name="student"; // Table name

$f\_email\_id="";

$f\_mobile\_number="";

$con=mysqli\_connect("$host", "$username", "$password","$db\_name")or die("cannot connect");

if (mysqli\_connect\_errno())

{

echo "Failed to connect to MySQL: " . mysqli\_connect\_error();

}

$username=$\_POST['admusername'];

$password=$\_POST['admpassword'];

if($username=="admin" && $password=="admin") { echo("Login successful <br> Click <a href=\"/AC/student\_form.html\">here</a> for the admission forms.<br> Click <a href=\"/AC/student\_update.html\">here</a> for the updating forms.<br> ");

}

else {

echo "Username and Password not found! You will be redirected to home";

echo("Click <a href=\"/AC/home1.html\">here</a> to go back.<br>");

}

?>

Here Username and Password are being verified by the query if they are not correct then a label is turned on stating “Invalid username and Password”.

If they are correct a session is generated by passing ID of that user to next profile page.

**Coding For Insertion in Registration Page**

<!DOCTYPE html>

<html>

<head>

<title>Applicant Registration</title>

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta charset="utf-8">

<link href="event.css" rel="stylesheet" type="text/css">

<style>

.error {color: #FF0000;}

</style>

</head>

<body>

<br>

<div align=center><font color="FF0000">

<?php

$host="localhost"; // Host name

$username="root"; // Mysql username

$password="abc"; // Mysql password

$db\_name="studentadmission"; // Database name

$tbl\_name="student"; // Table name

$f\_email\_id="";

$f\_mobile\_number="";

$con=mysqli\_connect("$host", "$username", "$password","$db\_name")or die("cannot connect");

if (mysqli\_connect\_errno())

{

echo "Failed to connect to MySQL: " . mysqli\_connect\_error();

}

$a1e=0;

if ($\_SERVER["REQUEST\_METHOD"] == "POST")

{

if (empty($\_POST["Admission\_no"]))

{

echo("Admission number is required.<br>");

$a1e=1;

}

else

{$Admission\_no = test\_input($\_POST["Admission\_no"]);}

if (empty($\_POST["First\_name"]))

{ echo("First name is required.<br>");

$a1e=1;

}

else

{$First\_name= test\_input($\_POST["First\_name"]);}

if (!preg\_match("/^[a-zA-Z ]\*$/",$First\_name))

{

echo ("Only letters and white space allowed.<br>");

$a1e=1;

}

$Middle\_name= test\_input($\_POST["Middle\_name"]);

if (empty($\_POST["Last\_name"]))

{echo "Last name is required.<br>"; $a1e=1;

}

else

{$Last\_name= test\_input($\_POST["Last\_name"]);}

if (!preg\_match("/^[a-zA-Z ]\*$/",$Last\_name))

{

echo "Only letters and white space allowed.<br>";

$a1e=1;

}

if (empty($\_POST["Age"]))

{echo "Age is required.<br>";

$a1e=1;

}

else if (!(is\_numeric($\_POST["Age"])))

{ echo " Enter a valid age.<br>";

echo $Age;

$a1e=1;}

else

{$Age= test\_input($\_POST["Age"]);}

if (empty($\_POST["DOB"]))

{echo "DOB is required.<br>";

$a1e=1;}

else

{$DOB= test\_input($\_POST["DOB"]);}

$Mother\_tongue=$\_POST["Mother\_tongue"];

if (empty($\_POST["Gender"]))

{echo "Gender is required.<br>";

$a1e=1;}

else

{$Gender = test\_input($\_POST["Gender"]);}

$Blood\_group=$\_POST["Blood\_group"];

$Religion=$\_POST["Religion"];

/\* $Nationality=$\_POST["Nationality"]; \*/

$Category=$\_POST["Category"];

if (empty($\_POST["Caste"]))

{echo "Caste is required<br>";

$a1e=1;}

else

{$Caste= test\_input($\_POST["Caste"]);}

if (!preg\_match("/^[a-zA-Z ]\*$/",$Caste))

{

echo "Only letters and white space allowed";

$a1e=1;

}

if (empty($\_POST["10\_markscard"]))

{echo "10th marks card is required<br>";

$a1e=1;}

else

{$markscard10 = test\_input($\_POST["10\_markscard"]);}

if (empty($\_POST["12\_markscard"]))

{echo "12th marks card is required<br>";

$a1e=1;

}

else

{$markscard12 = test\_input($\_POST["12\_markscard"]);}

if (empty($\_POST["Diploma\_markscard"]))

{echo "12th marks card is required<br>";

$a1e=1;

}

else

{$Diploma\_markscard = test\_input($\_POST["Diploma\_markscard"]);}

if (empty($\_POST["TC"]))

{echo "TC is required<br>";

$a1e=1;

}

else

{$TC = test\_input($\_POST["TC"]);}

if (empty($\_POST["eligibility\_VTU"]))

{echo "VTU eligibility is required<br>";

$a1e=1;

}

else

{$eligibility\_VTU = test\_input($\_POST["eligibility\_VTU"]);}

if (empty($\_POST["caste\_certi"]))

{echo "Required field caste certificate<br>";

$a1e=1;}

else

{$caste\_certi = test\_input($\_POST["caste\_certi"]);}

if (empty($\_POST["income\_certi"]))

{echo "Required field income certificate <br>";}

else

{$income\_certi = test\_input($\_POST["income\_certi"]);}

if (empty($\_POST["photograph"]))

{echo "phtograph is required<br>";

$a1e=1;

}

else

{$photograph = test\_input($\_POST["photograph"]);}

$extra\_cirricular=$\_POST["extra\_cirricular"];

$family\_member=$\_POST["family\_member"];

$family\_member\_name=$\_POST["family\_member\_name"];

if (!preg\_match("/^[a-zA-Z ]\*$/",$family\_member\_name))

{

echo "Only letters and white space allowed<br>";

$a1e=1;

}

$program=$\_POST["program"];

$f\_batch=$\_POST["f\_batch"];

$f\_mobile\_number=test\_input($\_POST["f\_mobile\_number"]);

if( !(($f\_mobile\_number=="") or !preg\_match("/^([1]-)?[0-9]{3}-[0-9]{3}-[0-9]{4}$/i", $f\_mobile\_number)) )

{

echo "Please enter a valid phone number<br>";

$a1e=1;

}

$f\_email\_id=$\_POST["f\_email\_id"];

$f\_email\_id = test\_input($\_POST["f\_email\_id"]);

if (!($f\_email\_id=="" or preg\_match("/([\w\-]+\@[\w\-]+\.[\w\-]+)/",$f\_email\_id)))

{

echo "Invalid email format<br>";

$a1e=1;

}

$f\_designation=$\_POST["f\_designation"];

$f\_org=$\_POST["f\_org"];

$f\_add\_org=$\_POST["f\_add\_org"];

}

if($a1e==1)

{ echo("Error in registration. Click <a href=\"/phase2/student\_form.html\">here</a> to go back.<br>");

die(mysqli\_error($con));

}

$sql="INSERT INTO student(Admission\_no,First\_name,Middle\_name,Last\_name,Age,DOB,Mother\_tongue,Gender,Blood\_group,Religion,Nationality,Category,Caste,10\_markscard,12\_markscard,Diploma\_markscard,TC, eligibility\_VTU, caste\_certi, income\_certi, photograph, extra\_cirricular, family\_member,family\_member\_name, program, f\_batch, f\_mobile\_number,f\_email\_id, f\_designation, f\_org,f\_add\_org)

VALUES('$Admission\_no','$First\_name','$Middle\_name','$Last\_name','$Age','$DOB','$Mother\_tongue','$Gender','$Blood\_group','$Religion','$\_POST[Nationality]','$Category','$Caste','$markscard10','$markscard12','$Diploma\_markscard','$TC','$eligibility\_VTU','$caste\_certi','$income\_certi','$photograph','$extra\_cirricular','$family\_member','$family\_member\_name','$program','$f\_batch','$f\_mobile\_number','$f\_email\_id','$f\_designation','$f\_org','$f\_add\_org')";

$result=mysqli\_query($con,$sql);

if (!($result)) {

echo("Error in registration. Click <a href=\"/AC/student\_form.html\">here</a> to go back.<br>");

}

else{

echo("Thank you for registering. Click <a href=\"/AC/contact1\_form.php\">here</a> to continue.");

}

function test\_input($data)

{

$data = trim($data);

$data = stripslashes($data);

$data = htmlspecialchars($data);

return $data;

}

mysqli\_close($con);

?>

</font></div>

</body>

</html>

**Code for report generation**

<!DOCTYPE html>

<html>

<head>

<title>Applicant Registration</title>

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta charset="utf-8">

<link href="event.css" rel="stylesheet" type="text/css">

<style>

.error {color: #FF0000;}

</style>

</head>

<body>

<br><br><br><br><br><br><br><br>

<div align=center><font color="000000">

<?php

$host="localhost"; // Host name

$username="root"; // Mysql username

$password="abc"; // Mysql password

$db\_name="studentadmission"; // Database name

$tbl\_name="student"; // Table name

$f\_email\_id="";

$f\_mobile\_number="";

// Connect to server and select database.

$con=mysqli\_connect("$host", "$username", "$password","$db\_name")or die("cannot connect");

$x = $\_POST['Admission\_no'];

if (mysqli\_connect\_errno())

{

echo "Failed to connect to MySQL: " . mysqli\_connect\_error();

}

$result = mysqli\_query($con,"SELECT Admission\_no,First\_name,Last\_name,Age,DOB,Gender,Nationality,Category,eligibility\_VTU FROM student");

echo"<h1>Student details are:</h1>" ;

echo "<table border='1'>

<tr>

<th>Admission\_no </th>

<th>Firstname</th>

<th>Lastname</th>

<th> Age</th>

<th>DOB </th>

<th> Gender</th>

<th> Nationality</th>

<th> Category</th>

<th> eligibility\_VTU</th>

</tr>";

while($row = mysqli\_fetch\_array($result))

{

echo "<tr>";

echo "<td>" . $row['Admission\_no'] . "</td>";

echo "<td>" . $row['First\_name'] . "</td>";

echo "<td>" . $row['Last\_name'] . "</td>";

echo "<td>" . $row['Age'] . "</td>";

echo "<td>" . $row['DOB'] . "</td>";

echo "<td>" . $row['Gender'] . "</td>";

echo "<td>" . $row['Nationality'] . "</td>";

echo "<td>" . $row['Category'] . "</td>";

echo "<td>" . $row['eligibility\_VTU'] . "</td>";

echo "</tr>";

}

echo "</table>";

mysqli\_close($con);

?>