Project Report

On

STUDENT RESULT MANAGEMENT SYSTEM

By

Gagan Sudhir Chaudhari

Roll No.1812007

MCA (2018-2021)

MES’ INSTITUTE OF MANAGEMENT CAREER AND COURSES (IMCC) , PUNE

Acknowledgement

I am very glad to take this opportunity to acknowledge all those who helped me in designing, developing and successful execution of my Project **“Student Result Management System”.**

I would like to extend my thanks and gratitude to my project guide **Dr.Meenakshi More** (Assistant Professor, IMCC) – Internal Guide for their valuable guidance and timely assistance throughout the development of this project. I would also like to extend my thanks and gratitude to **Dr.Santosh Deshpande** (Director,IMCC), **Dr.Ravindra Vaidya**(HOD,IMCC), **Dr.Manasi Bhate** (Head-Training and Placement,IMCC) and **Dr.Swapnaja Patwardhan** (Assistant Professor,IMCC and Class Coordinator- TYMCA) for their constant help and support

- Gagan Chaudhari

Index

Chapter 1

Introduction

1.1 Existing System and Need for System:

The “Student Result Management System” has been developed to override the problems prevailing in the practicing manual system. This software is supported to eliminate and, in some cases, reduce the hardships faced by this existing system. Moreover, this system is designed for the particular need of the company to carry out operations in a smooth and effective manner.

The applications are reduced as much as possible to avoid errors while entering the data. It also provides error message while entering invalid data. No formal knowledge is needed for the user to use this system. Thus, by this all t proves it is user-friendly. Student Result Management System, as described above, can lead to error free, secure, reliable, and fast management system. It can assist the user to concentrate on their activities rather to concentrate on the record keeping. Thus, it will help organization in better utilization of resources.

Every organization, whether big or small, has challenges to overcome and managing the information of Result, Student, Class, Subject. Every Student Result Management System has different Student needs; therefore, we design exclusive employee management system that are adapted to your managerial requirements. This is designed to assists in strategic planning, and will help you ensure that your organization is equipped with the right level of information and details for your future goals. Also, for those busy executives who are always on the go, our systems come with remote access features, which will allow you to manage your workforce anything, at all times. These systems will ultimately allow you to better manage resources.

1.2 **Scope of Work:**

It may help collecting perfect management in details. In a very short time, the collection will be obvious, simple and sensible. It will help a person to know the management of passed year perfectly and vividly. It also helps in current all works relative to

Student Result Management System. It will be also reduced the cost of collecting the management and collection procedure will go on smoothly.

Our project aims at Business process automation, i.e., we have tried to computerize various processes of Student Result Management System.

- In computer system the person has to fill the various forms and number of copies of the forms can be easily generated at a time.
- In computer system, it is not necessary to create the manifest but we can directly print it, which saves our time.
- To assist the staff in capturing the effort spent in their respective working areas.
- To utilize resources in an efficient manner by increasing their productivity through automation.
- The system generates types of information that can be used for various purposes.

- It satisfies the user requirements
- Be easy to understand by the user and operator.
- Be easy to operate
- Have a good user interface
- Be expandable
- Delivered on schedule within the budget.

1.3 Operating Environment – Hardware and software:

Hardware Requirements

Server Side:

RAM	: Minimum 4 GB
Hard Disk	: 60GB and Above.

Client Side:

RAM Minimum	: 2 GB
Hard Disk Minimum	: 80GB
Processor	: Dual Core or above

Software Requirements

Server Side:

Operating system : Windows

Processor : Core 2 Duo

Software : Web Server, HTML, CSS.

Client Side:

Development Technologies : php,HTML,CSS,JavaScript

Development Tools : Visual Code, Notepad++

Database : My SQL

Web Browser : Mozilla Firefox, Google Chrome

Hardware : P-IV or +, 1 GB RAM, 80GB HDD

Supporting Technologies : ASP.net, c#, HTML, CSS, JavaScript, Bootstrap, MySql server

1.4 Detail Description of Technology Used:

HTML-

HTML or Hyper Text Markup Language is the main markup language for creating web pages and other information that can be displayed in a web browser.HTML is written in the form of HTML elements consisting of *tags* enclosed in angle brackets (like <html>), within the web page content. HTML tags most commonly come in pairs like <h1> and </h1>, although some tags represent *empty elements* and so are unpaired, for example <img>. The first tag in a pair is the *start tag*, and the second tag is the *end tag* (they are also called *opening tags* and *closing tags*). In between these tags web designers can add text, further tags, comments and other types of text-based content. The purpose of a web browser is to read HTML documents and compose them into visible or audible web pages. The browser does not display the HTML tags, but uses the tags to interpret the content of the page.HTML elements form the building blocks of all websites. HTML allows images and objects to be embedded and can be used to create interactive forms. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. It can embed scripts written in languages such as JavaScript which affect the behavior of HTML web pages.

CSS-

Cascading Style Sheets (CSS) is a style sheet language used for describing the look and formatting of a document written in a markup language. While most often used to style web pages and interfaces written in HTML and XHTML, the language can be applied to any kind of XML document, including plain XML, SVG and XUL. CSS is a cornerstone specification of the web and almost all web pages use CSS style sheets to describe their presentation.CSS is designed primarily to enable the separation of document content from document presentation, including elements such as the layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple pages to share formatting, and reduce complexity and repetition in the structural content (such as by allowing for table less web design).CSS can also allow the same markup page to be presented in different styles for different rendering methods, such as on-screen, in print, by voice (when read out by a speech-based browser or screen reader) and on Braille-based, tactile devices.

JAVA SCRIPT-

JavaScript (JS) is a dynamic computer programming language. It is most commonly used as part of web browsers, whose implementations allow client-side scripts to interact with the user, control the browser, communicate asynchronously, and alter the document content that is displayed. It is also being used in server-side programming, game development and the creation of desktop and mobile applications. JavaScript is a prototype-based scripting language with dynamic typing and has first class functions. Its syntax was influenced by C. JavaScript copies many names and naming conventions from Java, but the two languages are otherwise unrelated and have very different semantics. The key design principles within JavaScript are taken from the self and Scheme programming languages. It is a multiparadigm language, supporting object-oriented, imperative, and functional programming styles. The application of JavaScript to use outside of web pages—for example, in PDF documents, site-specific browsers, and desktop widgets—is also significant. Newer and faster JavaScript VMs and platforms built upon them (notably Node.js) have also increased the popularity of JavaScript for server-side web applications. On the client side, JavaScript was traditionally implemented

PHP

PHP is a server-side scripting language designed specifically for the web. Within an HTML page, you can embed PHP code that will be executed each time the page is visited. Your PHP code is interpreted at the web server and generates HTML or other output that the visitor will see.

PHP was introduced in 1994. As of November 2007, it was installed on more than 21 million domains worldwide, and this number is growing rapidly. You can see the current number at http://www.php.net/usage.php

PHP is an Open Source project. PHP originally stood for Personal Home Page and now stands for PHP Hypertext Preprocessor.

Unique Features

If you are familiar with other server side language like ASP.NET or JSP you might be wondering what makes PHP so special, or so different from these competing alternatives well, here are some reasons:

Performance

Portability(Platform Independent)

Ease Of Use

Open Source

Third-Party Application Support

Community Support

Performance

Scripts written in PHP executives faster than those written in other scripting language, with numerous independent benchmarks, putting the language ahead of competing alternatives like JSP,ASP.NET and PERL.The PHP 5.0 engine was completely redesigned with an optimized memory manager to improve performance, and is noticeable faster than previous versions.In addition, third party accelerators are available to further improve performance and response time.

Portability

PHP is available for UNIX, MICROSOFT WINDOWS, MAC OS, and OS/2.PHP Programs are portable between platforms. As a result, a PHP application developed on, say, Windows will typically run on UNIX without any significant issues.This ability to easily undertake cross-platform development is a valuable one, especially when operating in a multi platform corporate environment or when trying to address multiple market segments.

Ease Of Use

“Simplicity is the ultimate sophistication”, Said Leonardo da Vinci, and by that measure, PHP is an extremely sophisticated programming language. Its syntax is clear and consistent, and it comes with exhaustive documentation for the 5000+ functions included with the core distributions .This significantly reduces the learning curve for both novice and experienced programmers, and it’s one of the reasons that PHP is favored as a rapid prototyping tool for Web-based applications.

Open Source

PHP is an open source project – the language is developed by a worldwide team of volunteers who make its source code freely available on the Web, and it may be used without payment of licensing fees or investments in expensive hardware or software .This reduces software development costs without affecting either flexibility or reliabilityThe open-source nature of the code further means that any developer, anywhere , can inspect the code tree, spit errors, and suggest possible fixes, this produces a stable, robust product wherein bugs, once discovered, are rapidly resolved – sometimes within a few hours of discovery !.

Third-Party Application Support

One of PHP’s Strengths has historically been its support for a wide range of different databases, including MySQL, PostgreSQL, Oracle, and Microsoft SQL Server. PHP 5.3 Supports more than fifteen different database engines, and it includes a common API for database access. XML support makes it easy to read and write XML documents though they were native PHP data structures, access XML node collections using Xpath, and transform XML into other formats with XSLT style sheets.

Community Support

One of the nice things about a community-supported language like PHP is the access it offers to the creativity and imagination of hundreds of developers across the world.Within the PHP community, the fruits of this creativity may be found in PEAR, the PHP Extension and Application Repository and PECL, the PHP Extension Community Library, which contains hundreds of ready-,made widgets and extensions that developers can use to painlessly and new functionality to PHP.Using these widgets is often a more time-and cost-efficient alternative to rolling your own code.

PHP Server

The PHP Community Provides Some types of Software Server solution under The GNU (General Public License).

These are the following:

WAMP Server

LAMP Server

MAMP Server

XAMPP Server

All these types of software automatic configure inside operating system after installation it having PHP, MySQL, Apache and operating system base configuration file, it doesn’t need to configure manually.

WAMP----- Microsoft window o/s,ApacheMysql PHP

LAMP---- Linux Operating System Apache Mysql PHP

XAMPP---- x-os(cross operating system) Apache Mysql PHP Perl

Chapter 2

Proposed System

2.1 Proposed System:

The purposes of Student Result Management System are to automate the existing manual system by the computerized equipment and full-fledged computer software, fulfilling their requirements, so that their valuable data/information can be stored for a longer period with easy accessing and manipulation of the same. The required software and hardware are easily available and easy to work with.

Student Result Management System, as described above, can lead to error free, secure, reliable and fast management system. It can assist the user to concentrate on their other activities rather to concentrate on the record keeping. Thus, it will help organization in better utilization of resources. The organization can maintain computerized records without redundant entries. That means that one need not be distracted by information that is not relevant, while being able to reach the information.

The aim is to automate its existing manual system by the help of computerized equipment and full – fledged computer software, fulfilling their requirements, so that their valuable data/information can be stored for a longer period with easy accessing and manipulation of the same. Basically, the project describes how to manage for good performance and better services for the clients.

2.1.1 <u>Module Specification</u>

1. Student Management Module: Used for managing the Student details.
2. Subject Module: Used for managing the details of Subject
3. Result Management Module: Used for managing the information and details of the Result.
4. Class Module: Used for managing Class information
5. Login Module: Used for managing the login details.
6. Users Module: Used for managing the users of the system.

2.2 <u>Objective of system:</u>

The purposes of Student Result Management System are to automate the existing manual system by the computerized equipment and full-fledged computer software, fulfilling their requirements, so that their valuable data/information can be stored for a longer period with easy accessing and manipulation of the same. The required software and hardware are easily available and easy to work with.

Student Result Management System, as described above, can lead to error free, secure, reliable and fast management system. It can assist the user to concentrate on their other activities rather to concentrate on the record keeping. Thus, it will help organization in better utilization of resources. The organization can maintain computerized records without redundant entries. That means that one need not be distracted by information that is not relevant, while being able to reach the information.

The aim is to automate its existing manual system by the help of computerized equipment and full – fledged computer software, fulfilling their requirements, so that their valuable data/information can be stored for a longer period with easy accessing and manipulation of the same. Basically, the project describes how to manage for good performance and better services for the clients.

The main objective of the Project on Student Result Management System is to manage the details of Student, Result, Subject, Class, Semester. It manages all the information about Student, Result, Subject, Class, Semester. The project is totally built at administrative end and thus only the administrator is guaranteed the access. The purpose of the project is to build an application program to reduce the manual work for managing the Student, Result, Subject, Class, Semester. It tracks all the details about the Student, Result, Subject, Class, Semester.

2.3 User Requirements

Feature Set

- Admin can add/update/ Class explain them
- Admin can add/update/ Subjects

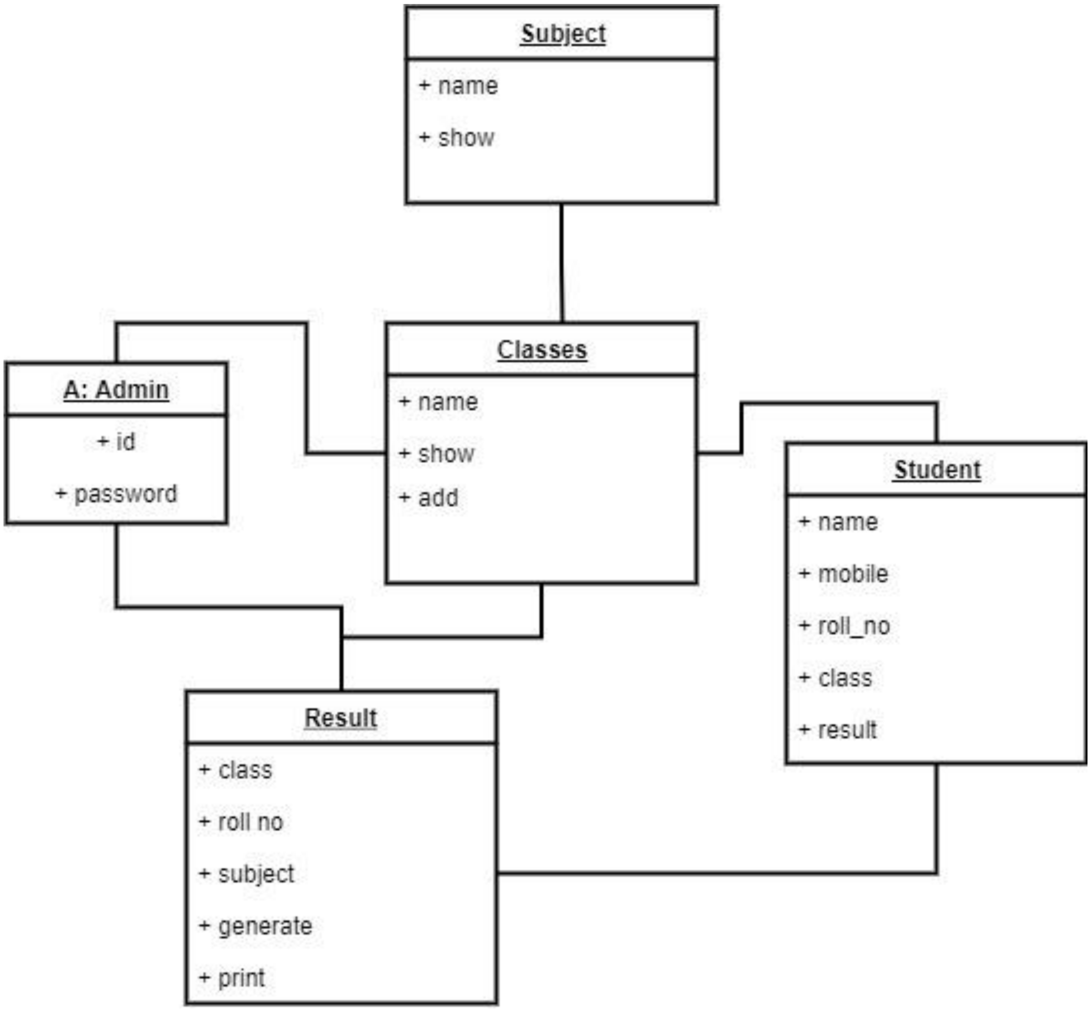
- Admin can add/update/ Active/Inactive Subject combination with class
- Admin can register new student and also edit info of the student
- If the Result is Already Declared then it won’t allow to declare with the same.
- Product and Component based.
- Creating and changing issues at ease.
- Query issue list to any depth
- Reporting and charting in more comprehensive way
- User accounts to control the access and maintain security
- Simple status and resolutions
- Multi-level priorities and Severities.
- Targets and milestones for guiding the programmers.
- Attachments and additional comments for more information.
- Robust database back – end.
- Various level of reports available with a lot of filter criteria’s
- It contains better storage capacity.

- Accuracy in work.
- Easy and fast retrieval of information
- Well-designed reports
- Decrease the load of the person involve in existing manual system.
- Access of any information individually.
- Work becomes very speedy
- Easy to update information.

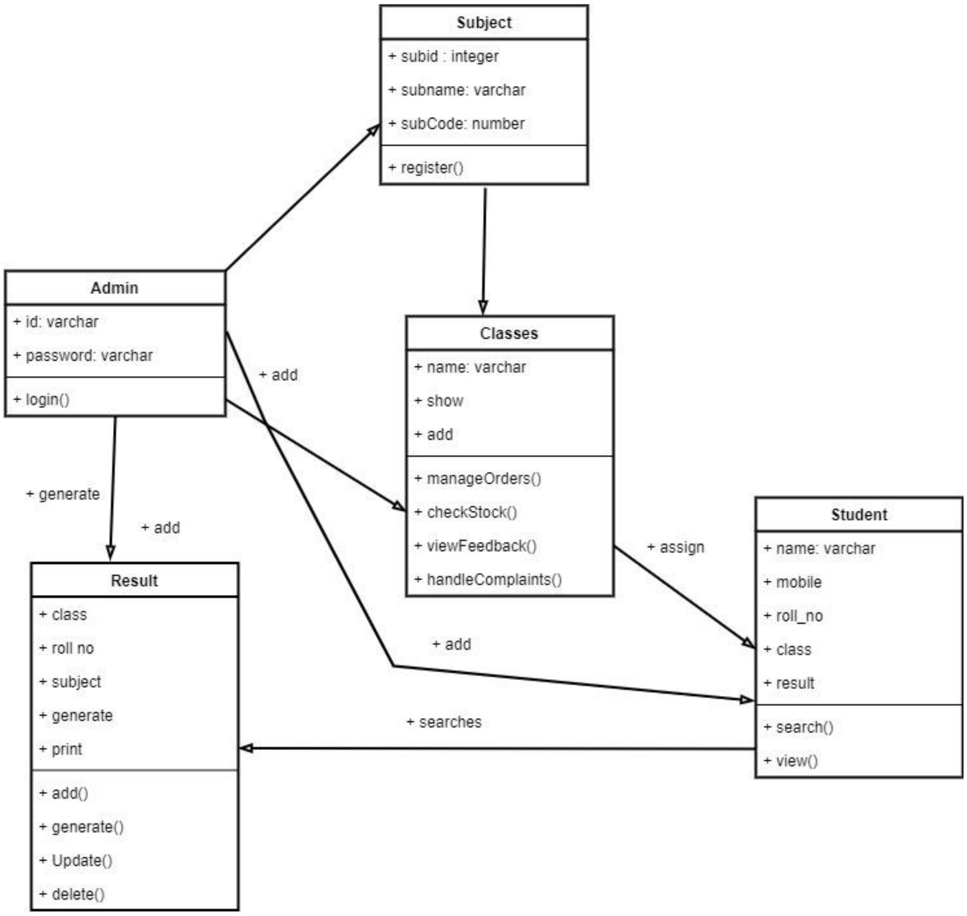
.

Chapter 3
Analysis and Design

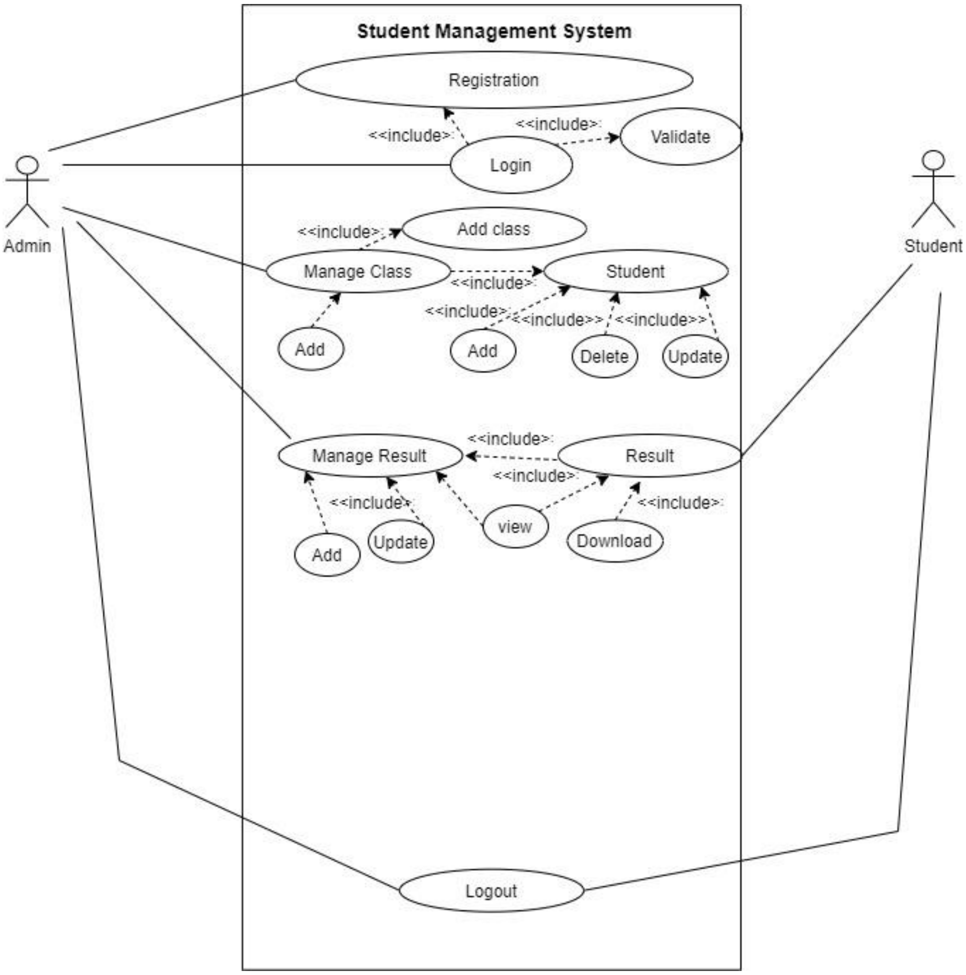
3.1 Object Diagram:



3.2 Class Diagram:

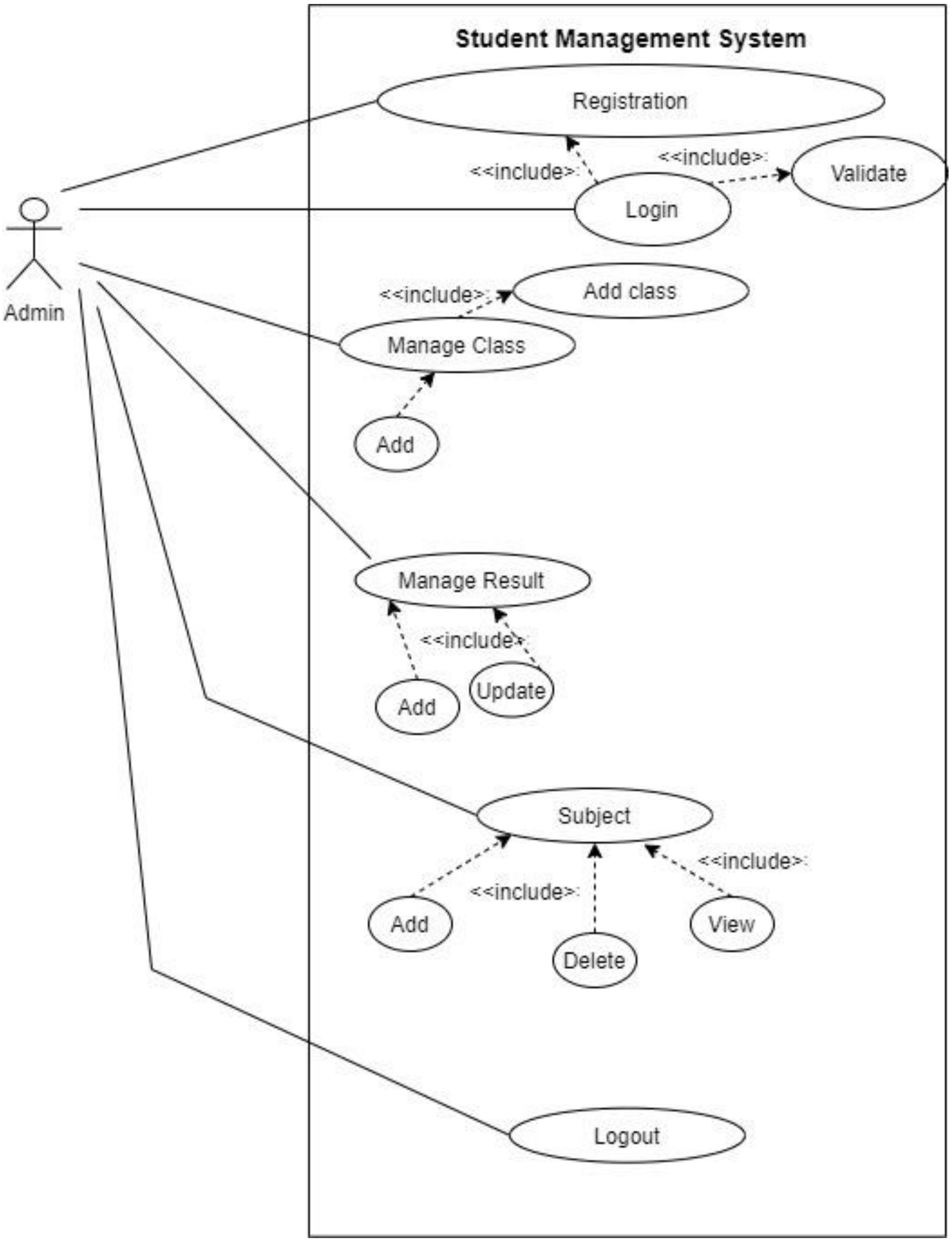


3.1 Use Case Diagrams:

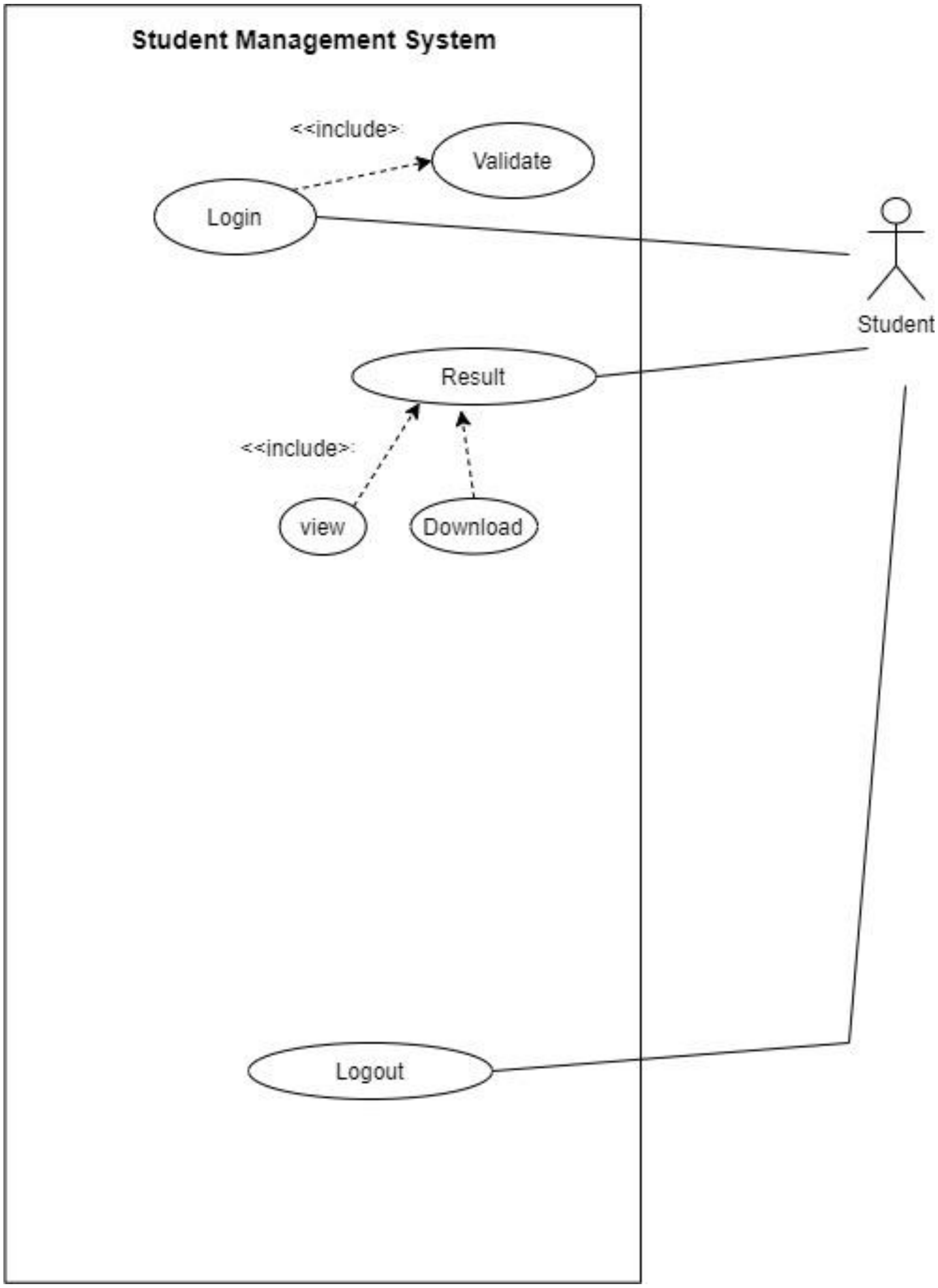


3. Use Case Diagram:

Admin

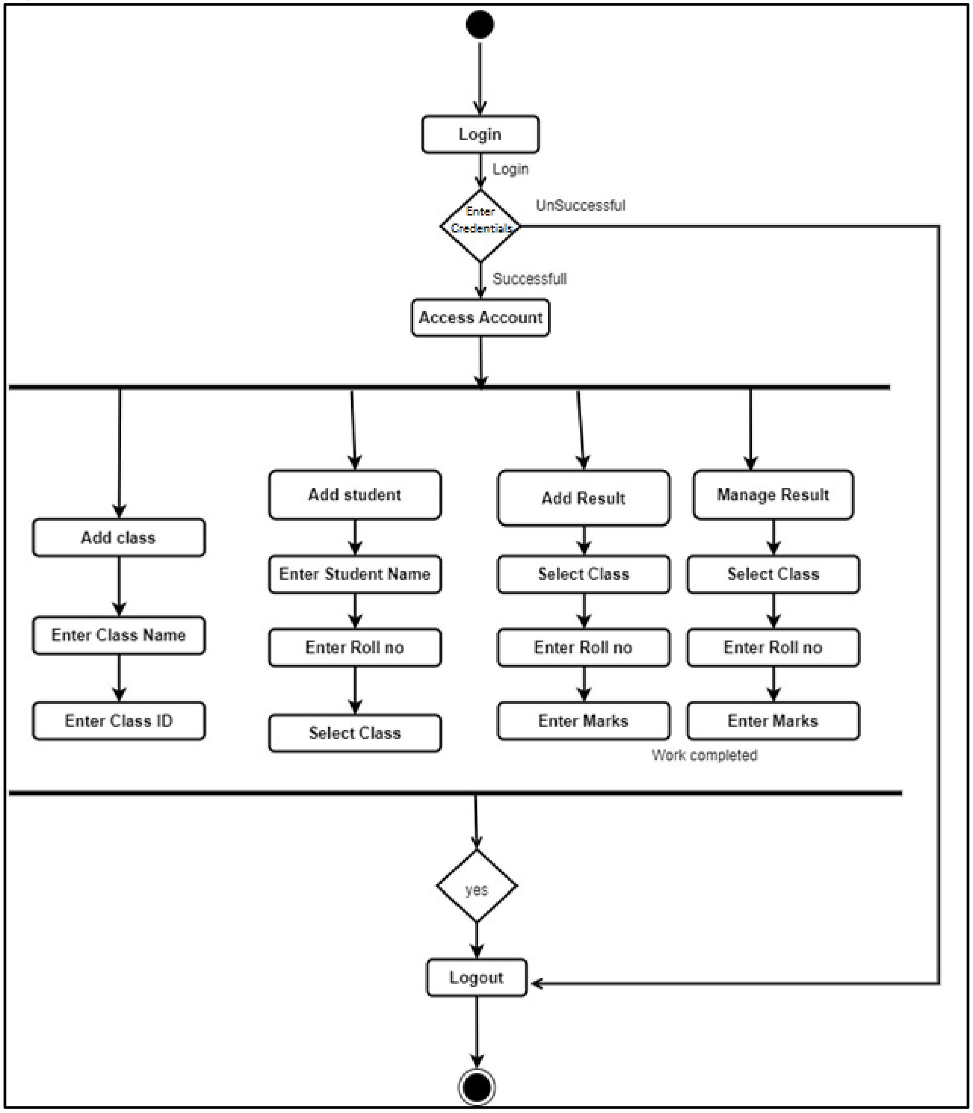


Student:

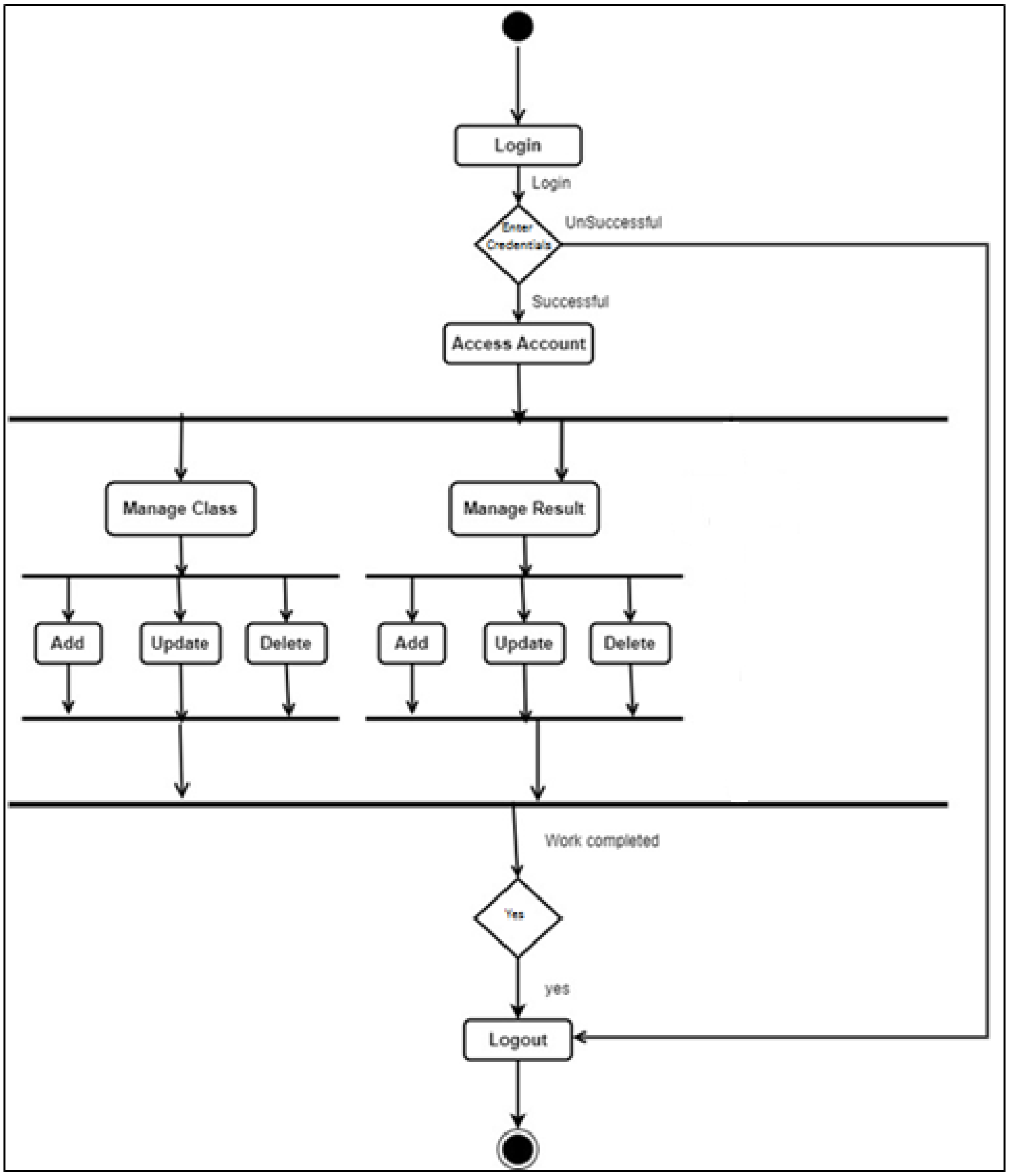

Student Management System
<<include>>
Validate
Login
Student
Result
<<include>>
view
Download
Logout

3.4 Activity Diagram:

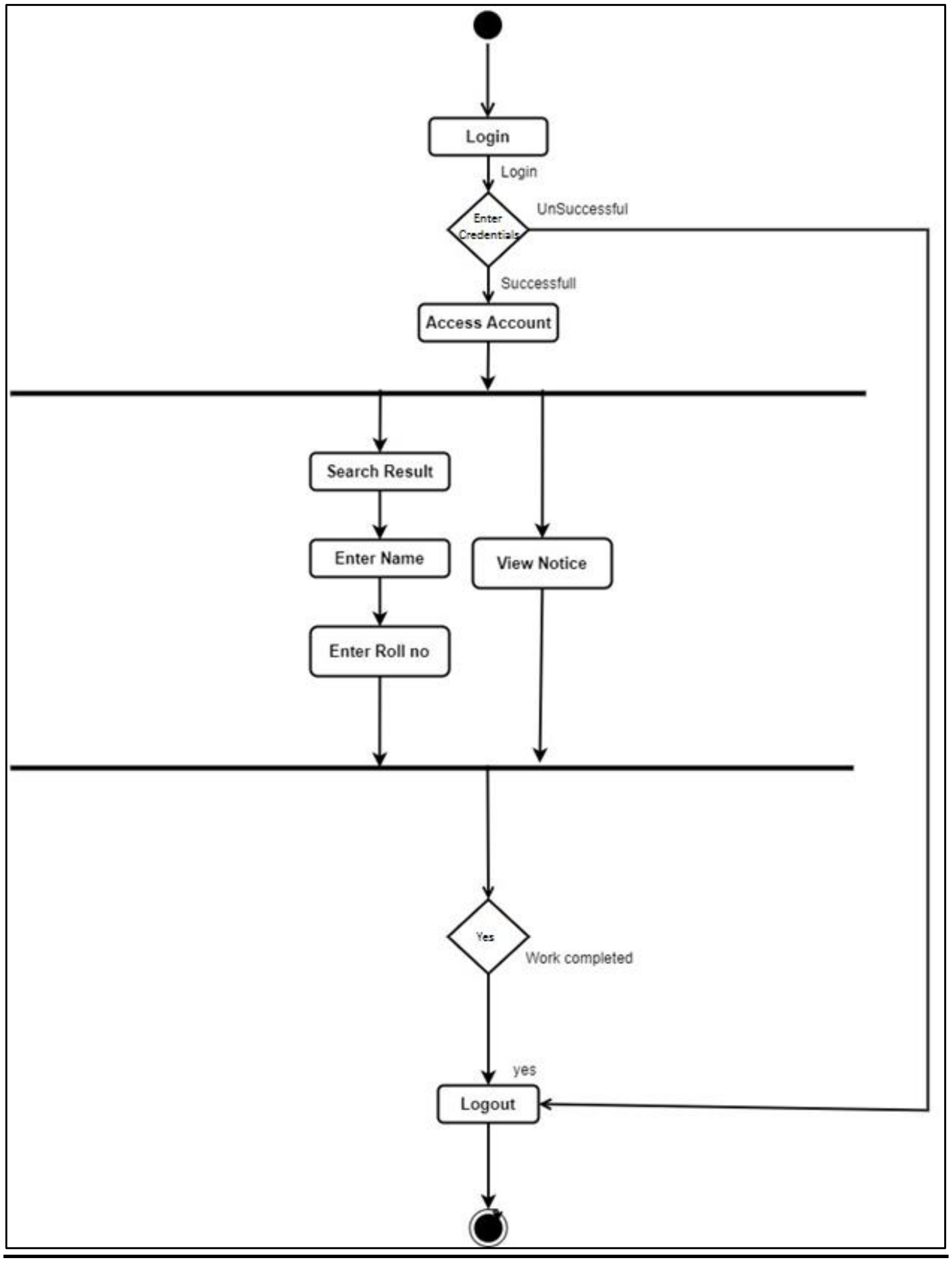
System:



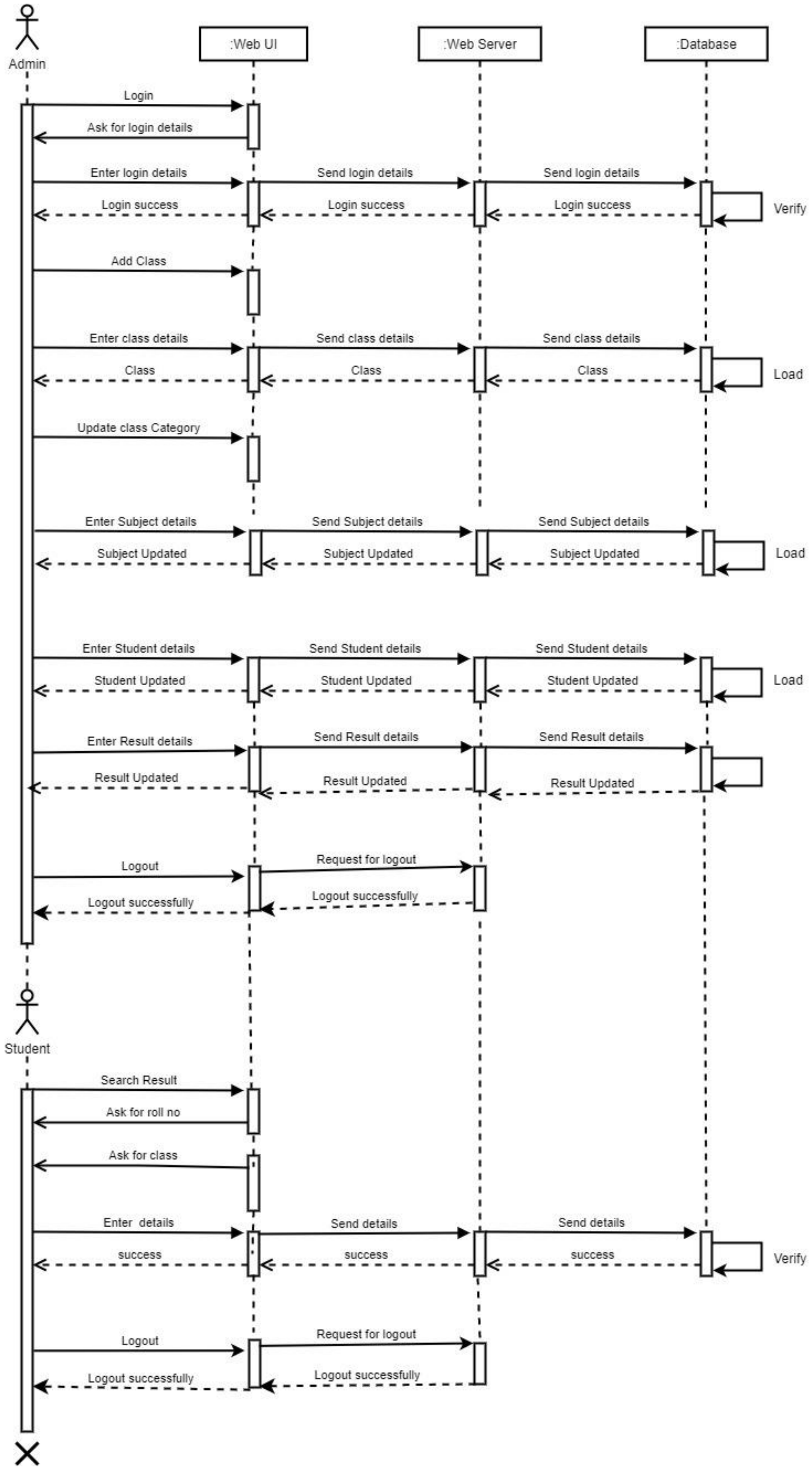
Admin:


Login
Login
Enter Credentials
UnSuccessful
Successful
Access Account
Manage Class
Manage Result
Add
Update
Delete
Add
Update
Delete
Work completed
Yes
yes
Logout

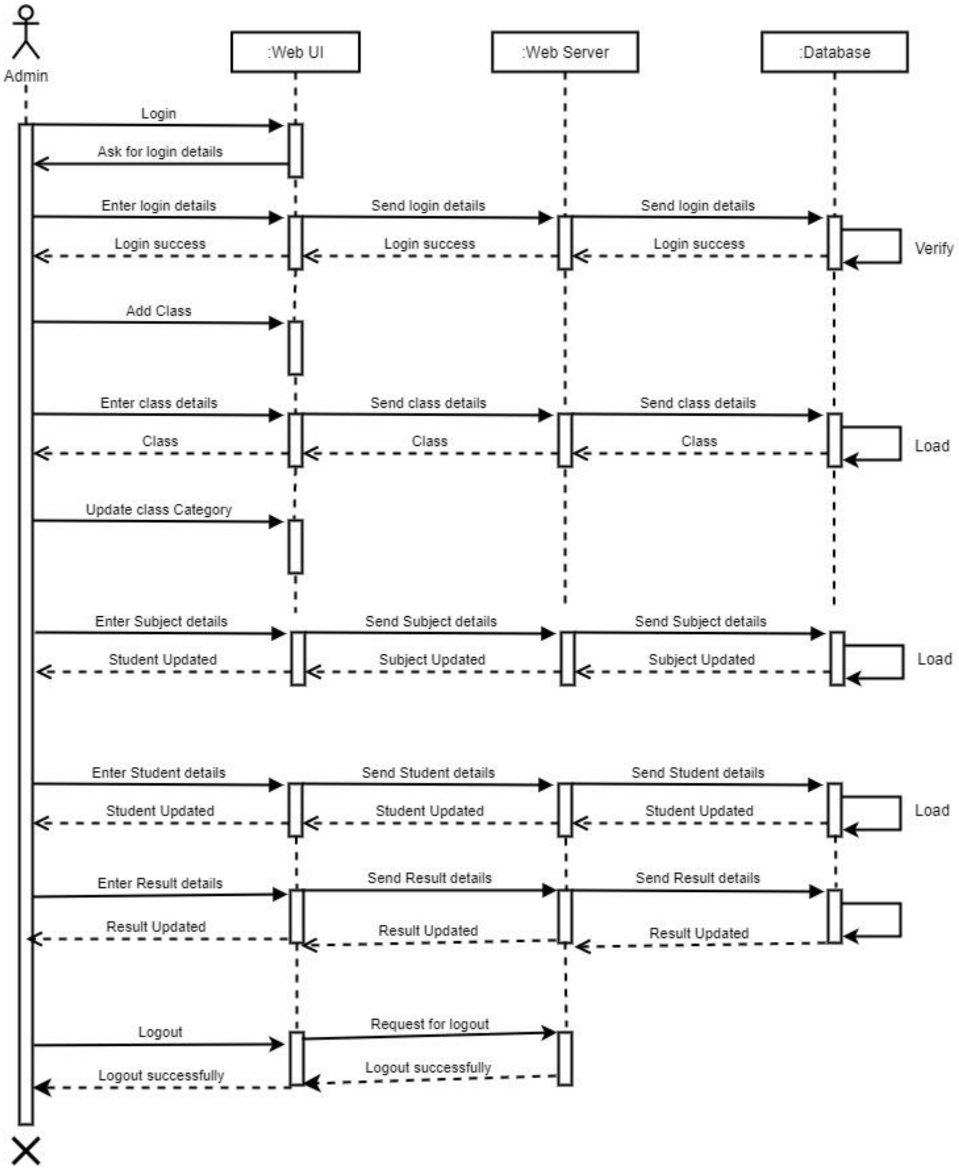
Student:


Login
Login
Enter Credentials
UnSuccessful
Successfull
Access Account
Search Result
Enter Name
Enter Roll no
View Notice
Yes
Work completed
yes
Logout

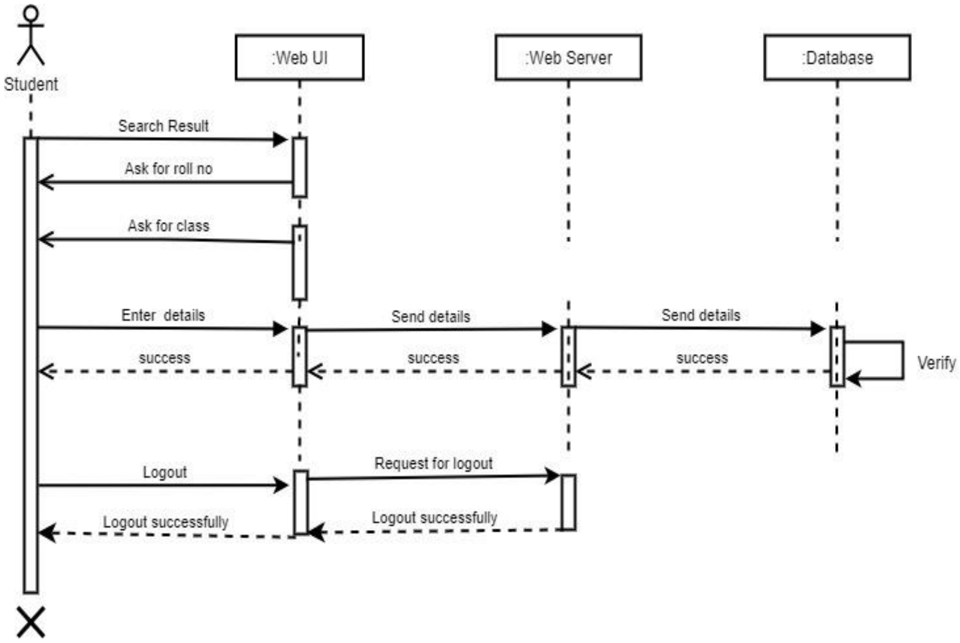
3.5 Sequence Diagram System:


Admin
:Web UI
:Web Server
:Database
Login
Ask for login details
Enter login details
Send login details
Send login details
Login success
Login success
Login success
Verify
Add Class
Enter class details
Send class details
Send class details
Class
Class
Class
Load
Update class Category
Enter Subject details
Send Subject details
Send Subject details
Subject Updated
Subject Updated
Subject Updated
Load
Enter Student details
Send Student details
Send Student details
Student Updated
Student Updated
Student Updated
Load
Enter Result details
Send Result details
Send Result details
Result Updated
Result Updated
Result Updated
Logout
Request for logout
Logout successfully
Logout successfully
Student
Search Result
Ask for roll no
Ask for class
Enter details
Send details
Send details
success
success
success
Verify
Logout
Request for logout
Logout successfully
Logout successfully

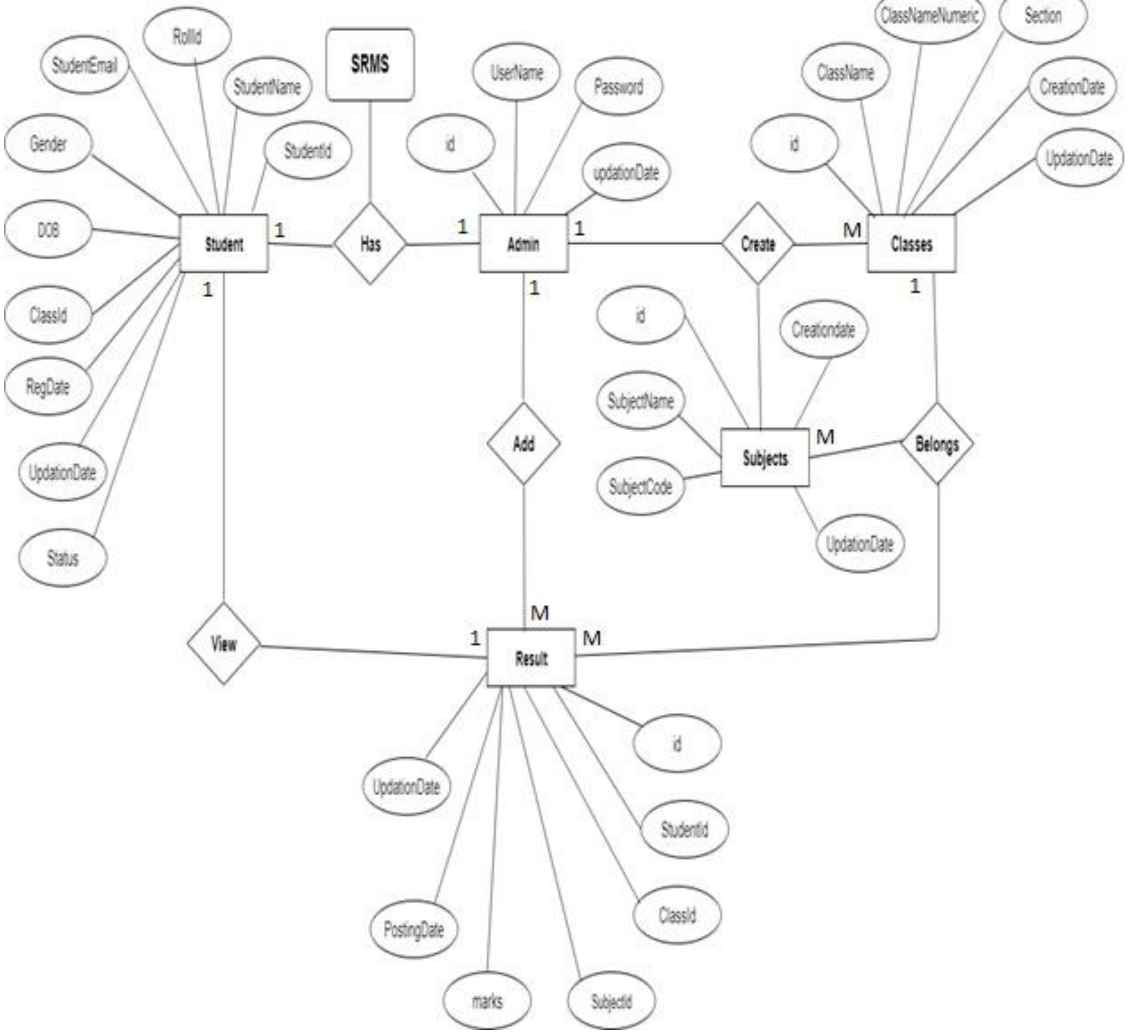
Admin


Admin
:Web UI
:Web Server
:Database
Login
Ask for login details
Enter login details
Send login details
Send login details
Login success
Login success
Login success
Verify
Add Class
Enter class details
Send class details
Send class details
Class
Class
Class
Load
Update class Category
Enter Subject details
Send Subject details
Send Subject details
Student Updated
Subject Updated
Subject Updated
Load
Enter Student details
Send Student details
Send Student details
Student Updated
Student Updated
Student Updated
Load
Enter Result details
Send Result details
Send Result details
Result Updated
Result Updated
Result Updated
Logout
Request for logout
Logout successfully
Logout successfully

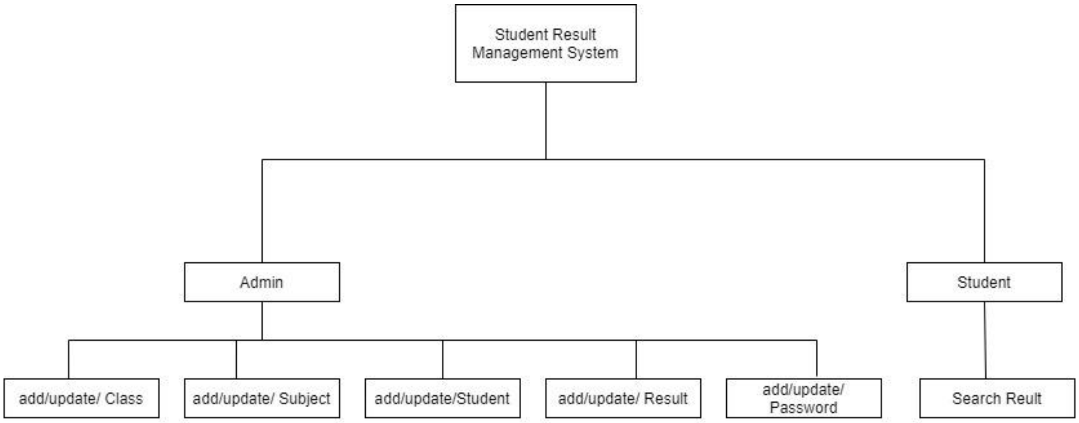
Student


Student
:Web UI
:Web Server
:Database
Search Result
Ask for roll no
Ask for class
Enter details
Send details
Send details
success
success
success
Verify
Logout
Request for logout
Logout successfully
Logout successfully

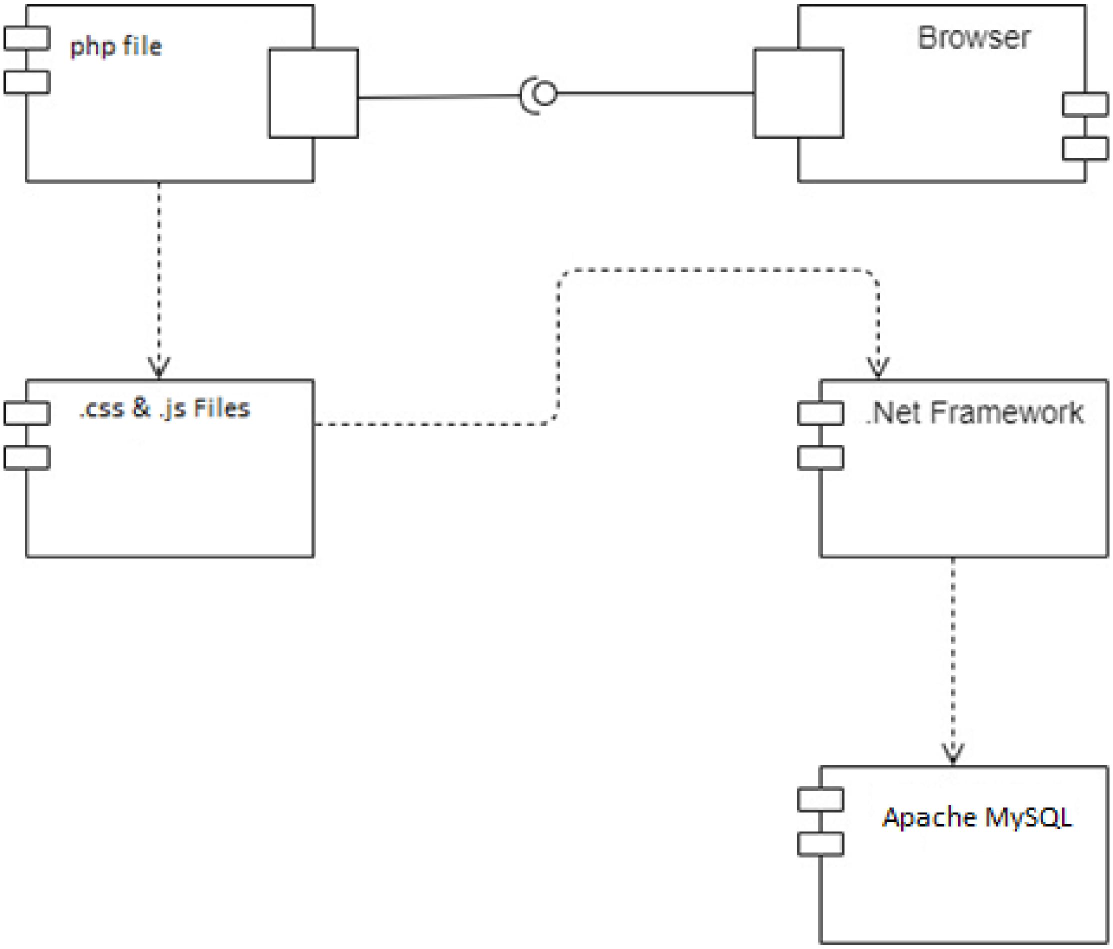
3.6 Entity Relationship Diagram:



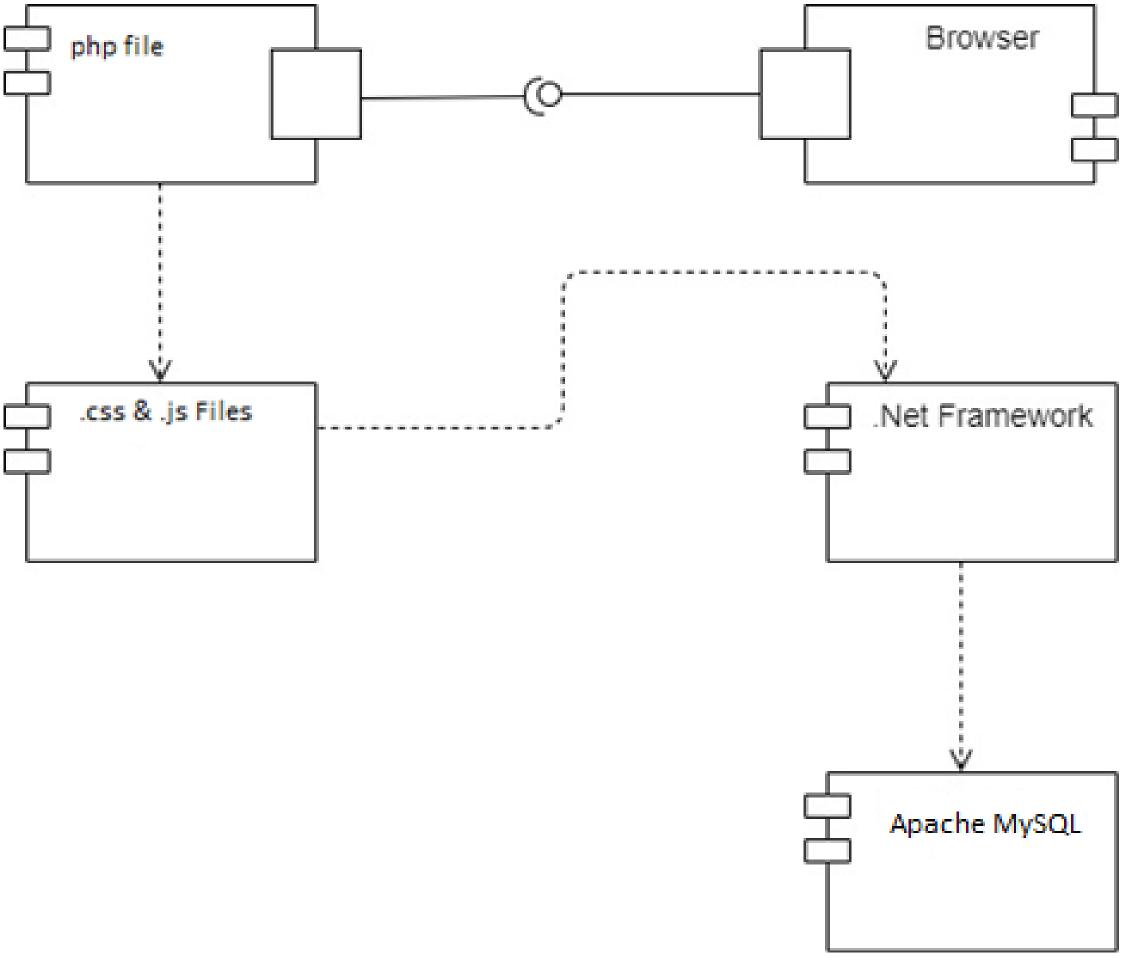
3.7 Module Hierarchy Diagram



3.8 Component Diagram



3.9 Deployment Diagram



3.10 Module Specification

As part of module hierarchy diagram there are four main modules in the project

- Class
- Subjects
- Active/Inactive Subject combination with class
- register new student and also edit info of the student
- Admin can change own password
- Result

This module will give all the available classes of the Students.

Admin can create the new class and update the class by selecting the appropriate fields along with that the admin can search the classes

Subject

This module will allow the admin to Add the subjects along with their respective code. The admin can also be able to update the code or the name of the subject.

With the respective classes Admin can combine the subjects which give the list of the subjects at the time of declaration of the student.

Student

This module will allow the Admin to Add into the Application and along with that it allows to create the result.

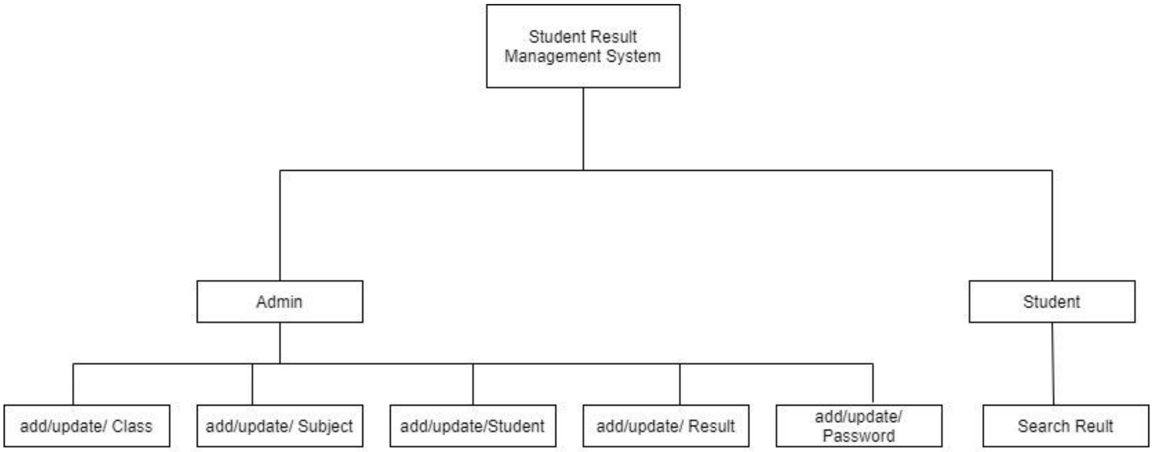
Student can search the result with their respective Roll Id and class selected

Result

In this module Admin and the Student both can play the roles.

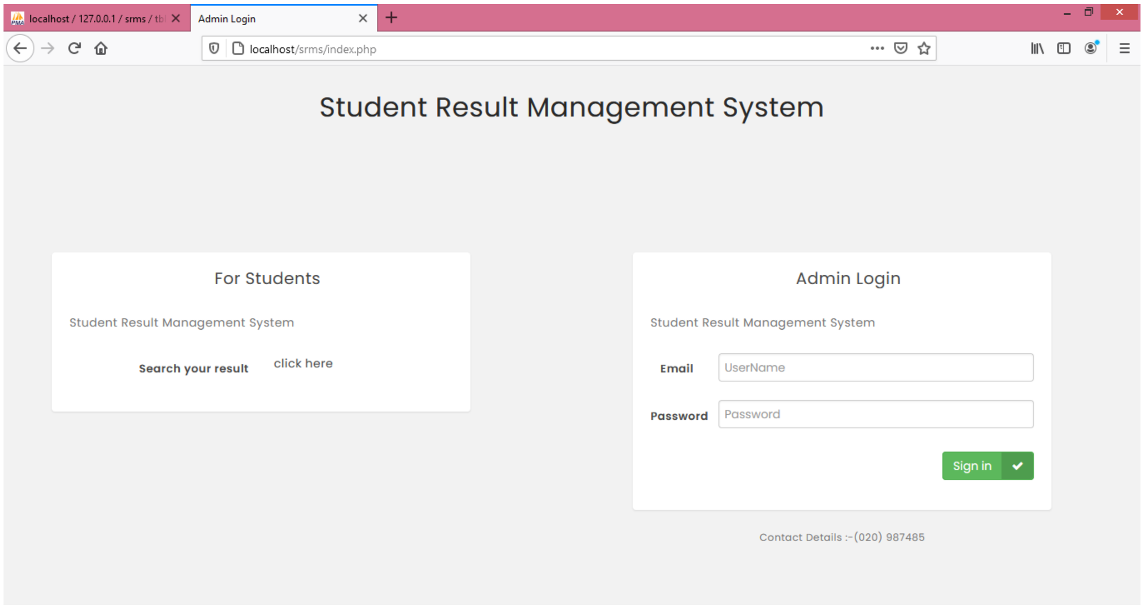
The admin declare the result and Student can search the result with the Roll ID and class respectively

3.11 Website Map



3.12 User Interface Design

Home Page



Student Result Management System

For Students

Student Result Management System

Search your result click here

Admin Login

Student Result Management System

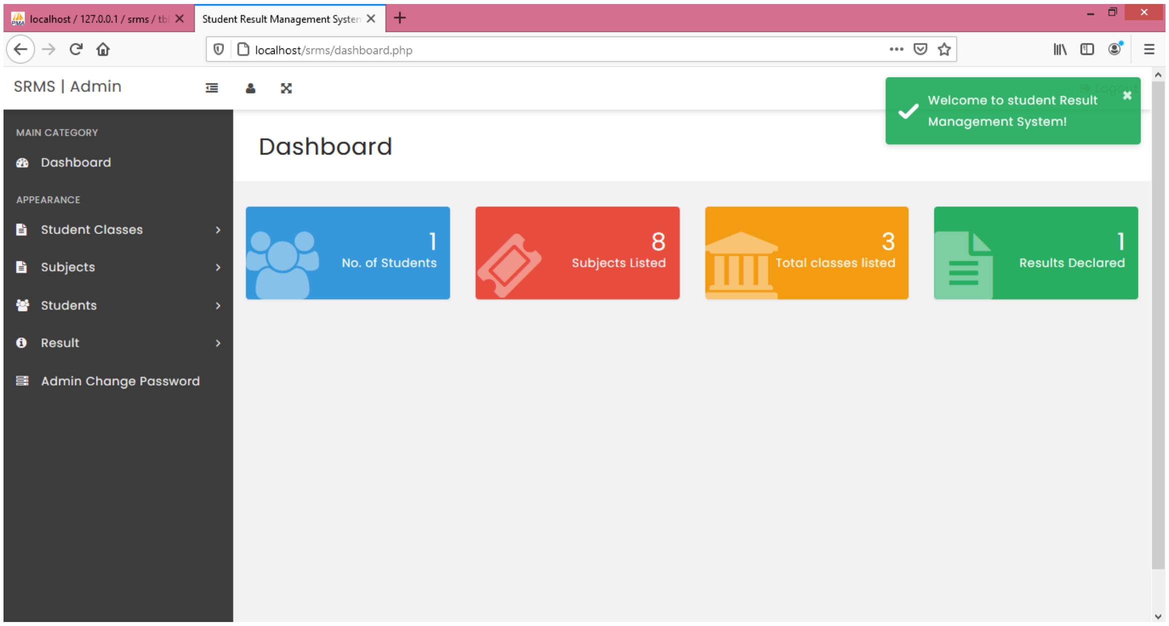
Email: UserName

Password: Password

Sign in ✓

Contact Details :-(020) 987485

Dashboard



Welcome to student Result Management System!

Dashboard

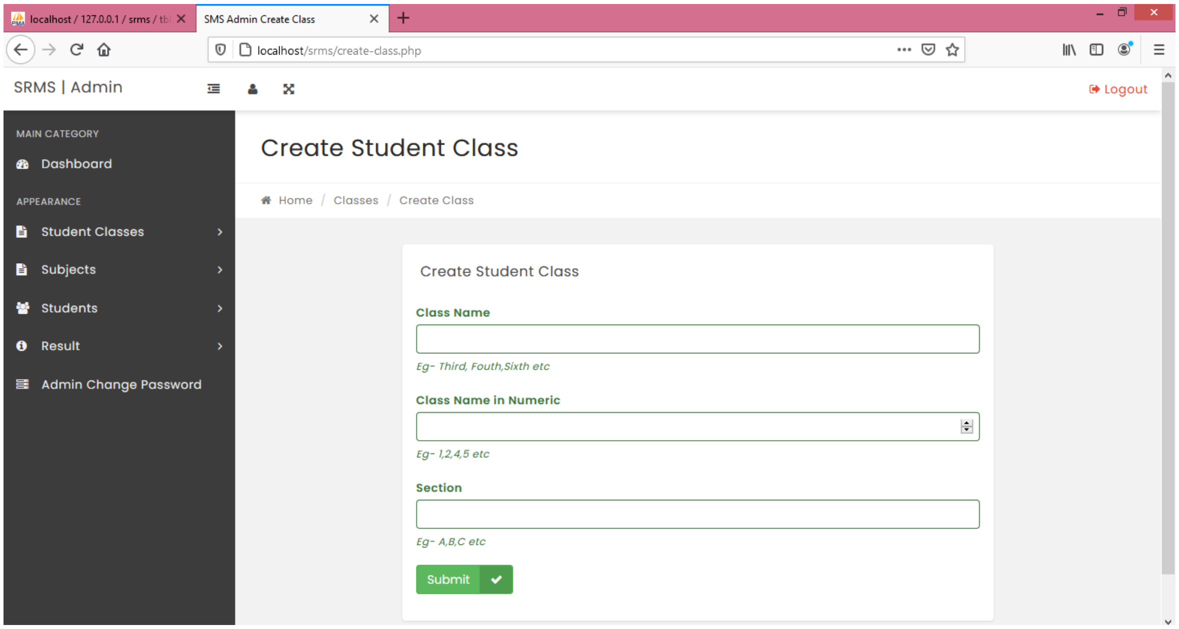
1 No. of Students

8 Subjects Listed

3 Total classes listed

1 Results Declared

Create Class



Create Student Class

Create Student Class

Class Name

Eg- Third, Fouth,Sixth etc

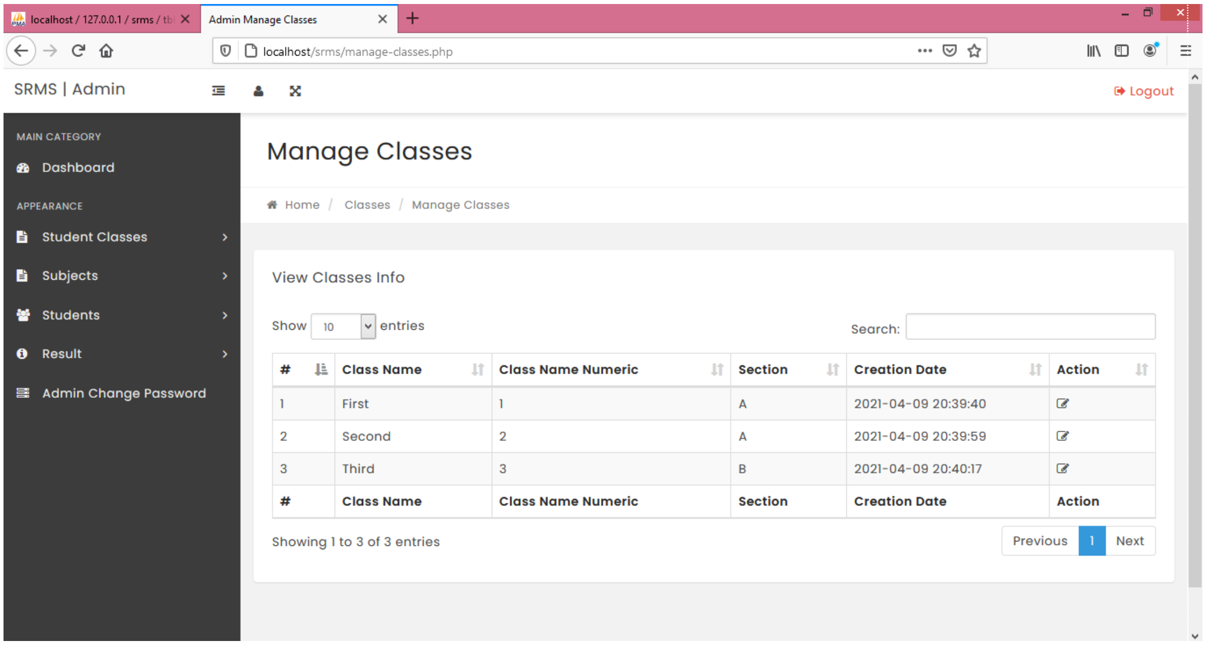
Class Name in Numeric

Eg- 1,2,4,5 etc

Section

Eg- A,B,C etc

Manage Class



Manage Classes

Home / Classes / Manage Classes

View Classes Info

Show 10 entries

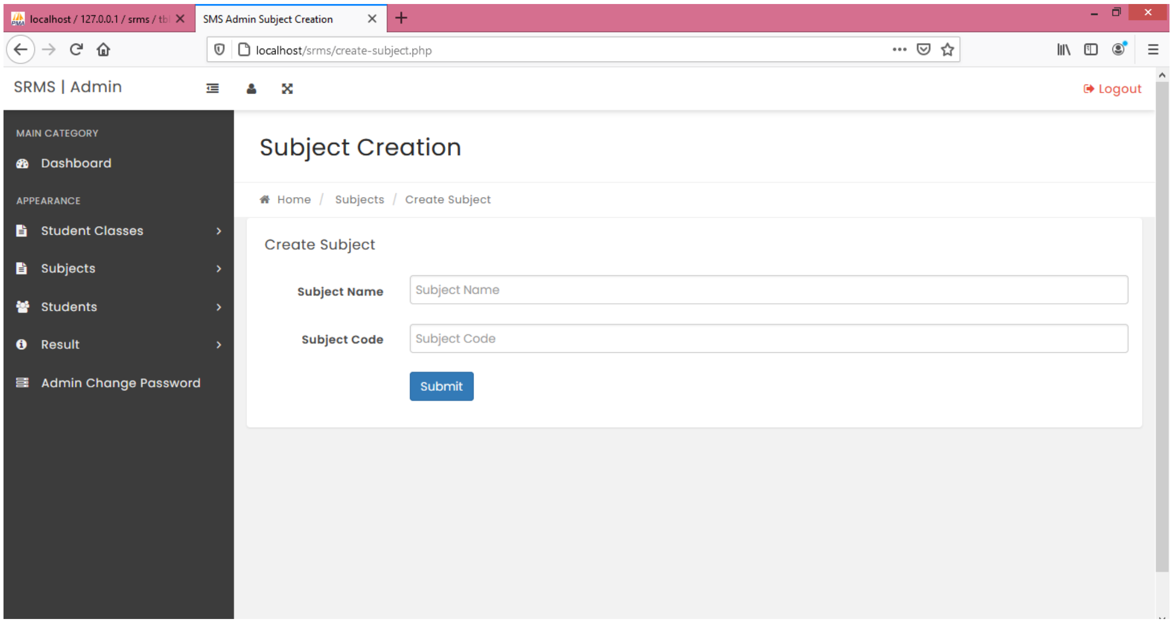
Search:

#	Class Name	Class Name Numeric	Section	Creation Date	Action
1	First	1	A	2021-04-09 20:39:40	
2	Second	2	A	2021-04-09 20:39:59	
3	Third	3	B	2021-04-09 20:40:17	
#	Class Name	Class Name Numeric	Section	Creation Date	Action

Showing 1 to 3 of 3 entries

Previous 1 Next

Subject Creation



Subject Creation

Home / Subjects / Create Subject

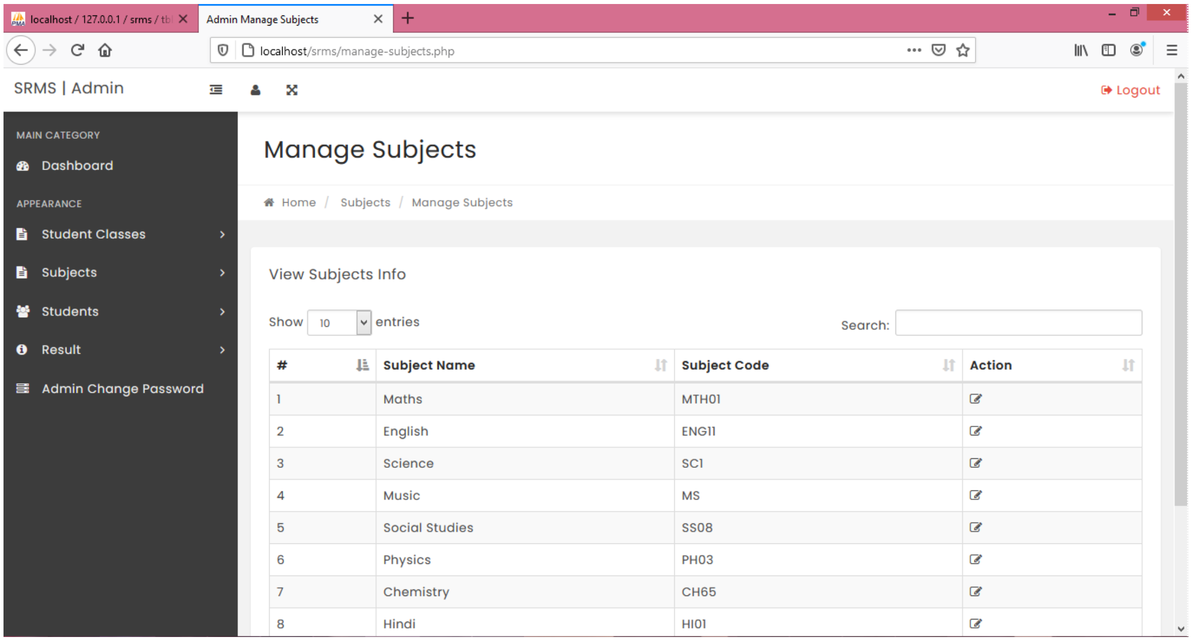
Create Subject

Subject Name: Subject Name

Subject Code: Subject Code

Submit

Manage Subject



Manage Subjects

Home / Subjects / Manage Subjects

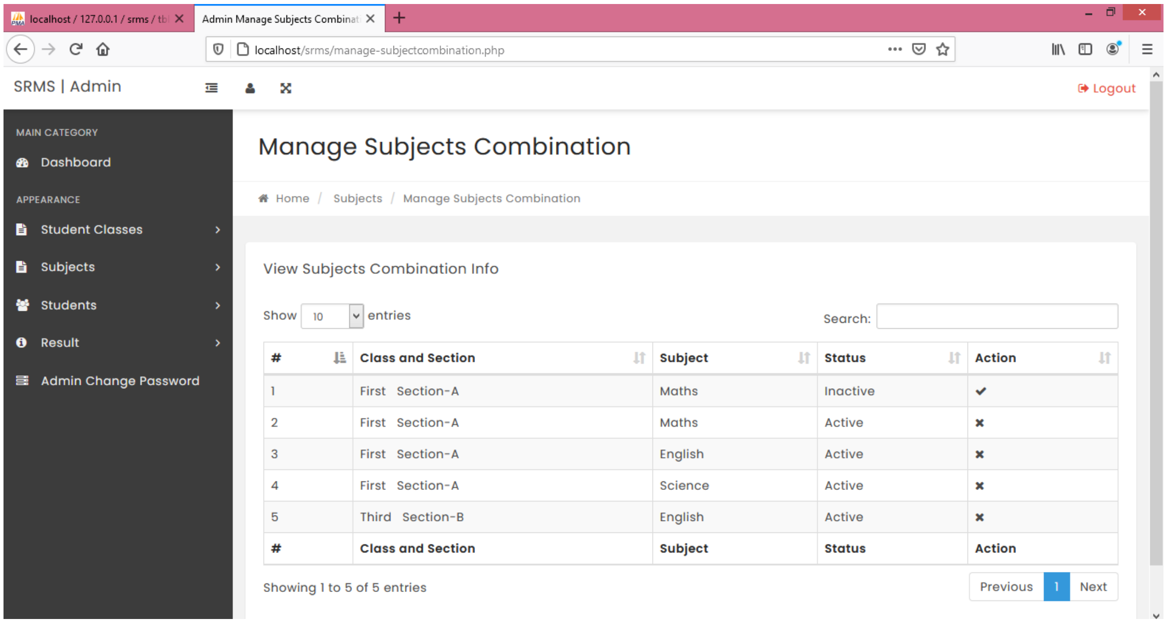
View Subjects Info

Show 10 entries

Search:

#	Subject Name	Subject Code	Action
1	Maths	MTH01	✎
2	English	ENG11	✎
3	Science	SC1	✎
4	Music	MS	✎
5	Social Studies	SS08	✎
6	Physics	PH03	✎
7	Chemistry	CH65	✎
8	Hindi	HI01	✎

Subject Combination


localhost / 127.0.0.1 / srms / tbl
Admin Manage Subjects Combinat
localhost/srms/manage-subjectcombination.php
SRMS | Admin
Logout
MAIN CATEGORY
Dashboard
APPEARANCE
Student Classes
Subjects
Students
Result
Admin Change Password
Home / Subjects / Manage Subjects Combination
Previous
1
Next

Manage Subjects Combination

View Subjects Combination Info

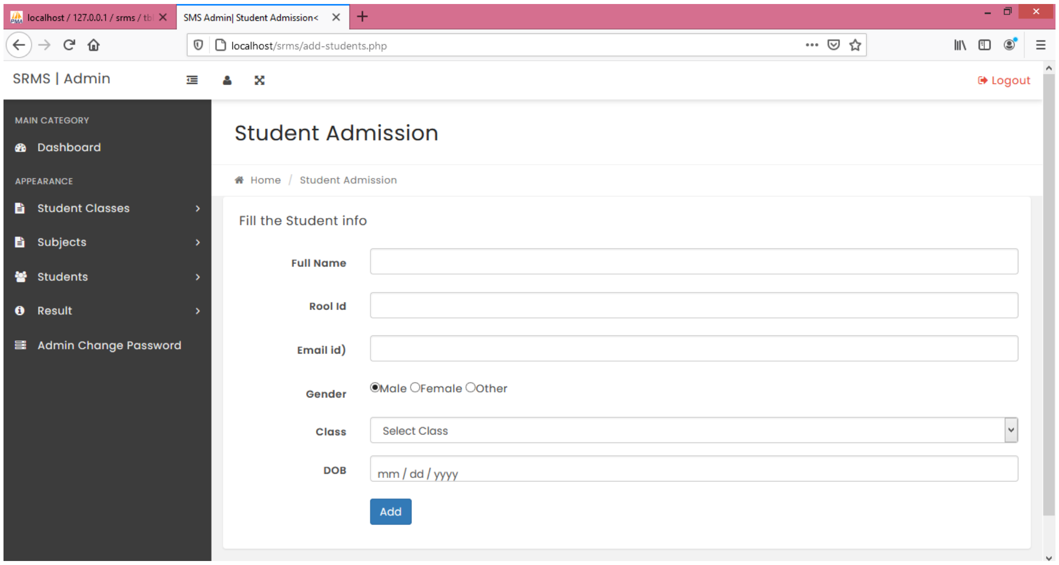
Show 10 entries

Search:

#	Class and Section	Subject	Status	Action
1	First Section-A	Maths	Inactive	✓
2	First Section-A	Maths	Active	✖
3	First Section-A	English	Active	✖
4	First Section-A	Science	Active	✖
5	Third Section-B	English	Active	✖
#	Class and Section	Subject	Status	Action

Showing 1 to 5 of 5 entries

Create Student



Student Admission

Fill the Student info

Full Name

Rool Id

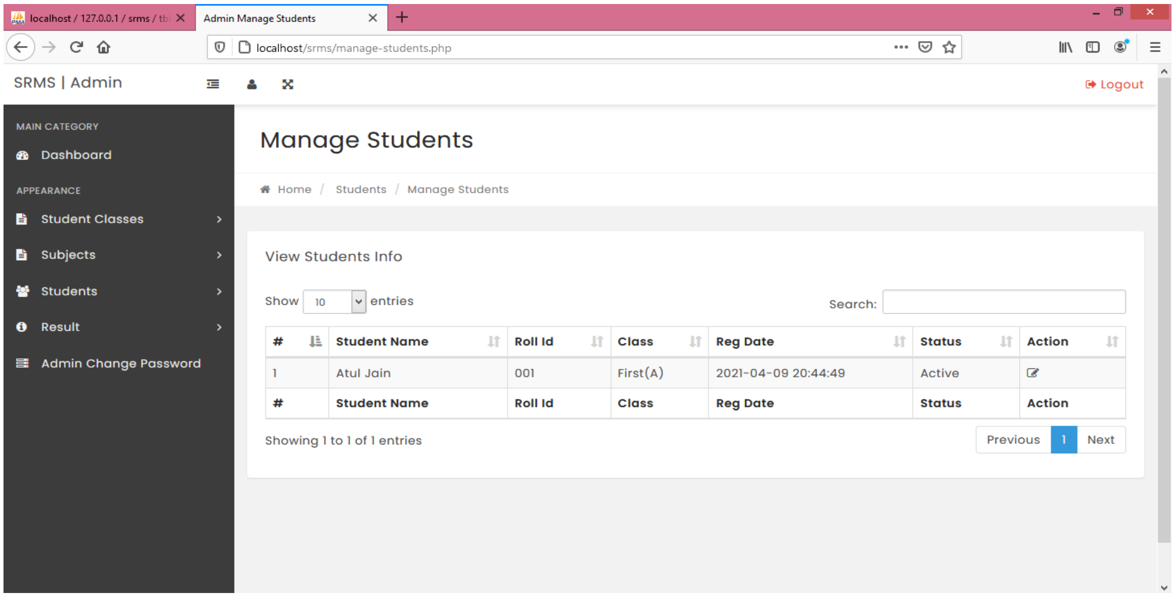
Email id)

Gender ●Male ○Female ○Other

Class Select Class

DOB mm / dd / yyyy

Manage Student



Manage Students

Home / Students / Manage Students

View Students Info

Show 10 entries

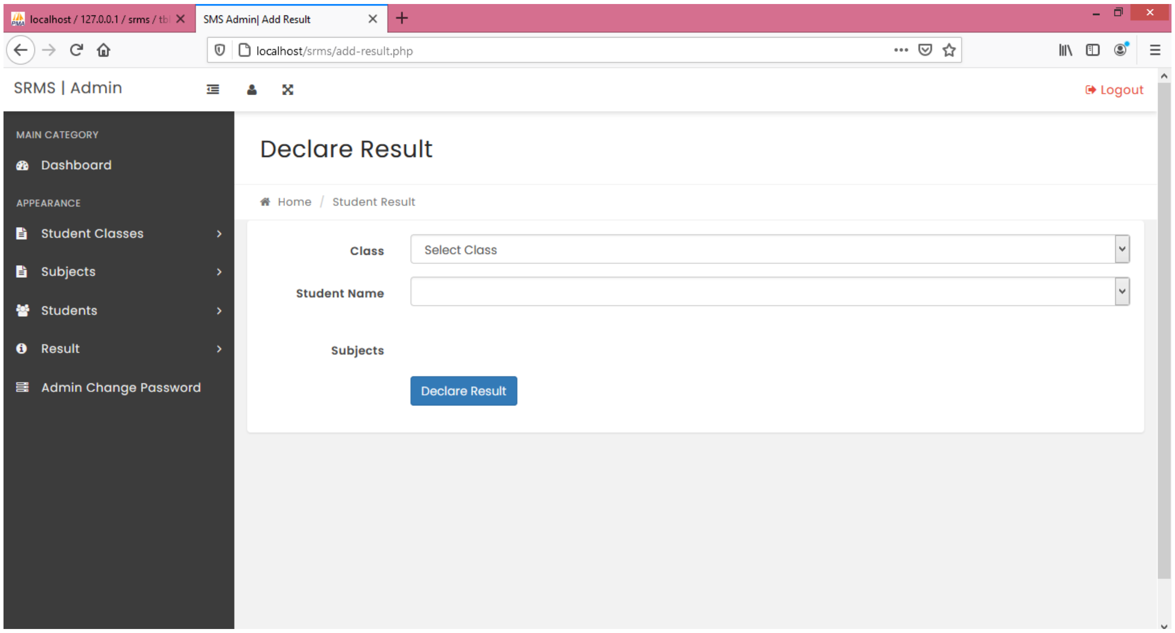
Search:

#	Student Name	Roll Id	Class	Reg Date	Status	Action
1	Atul Jain	001	First(A)	2021-04-09 20:44:49	Active	
#	Student Name	Roll Id	Class	Reg Date	Status	Action

Showing 1 to 1 of 1 entries

Previous 1 Next

Declare Result



Declare Result

Home / Student Result

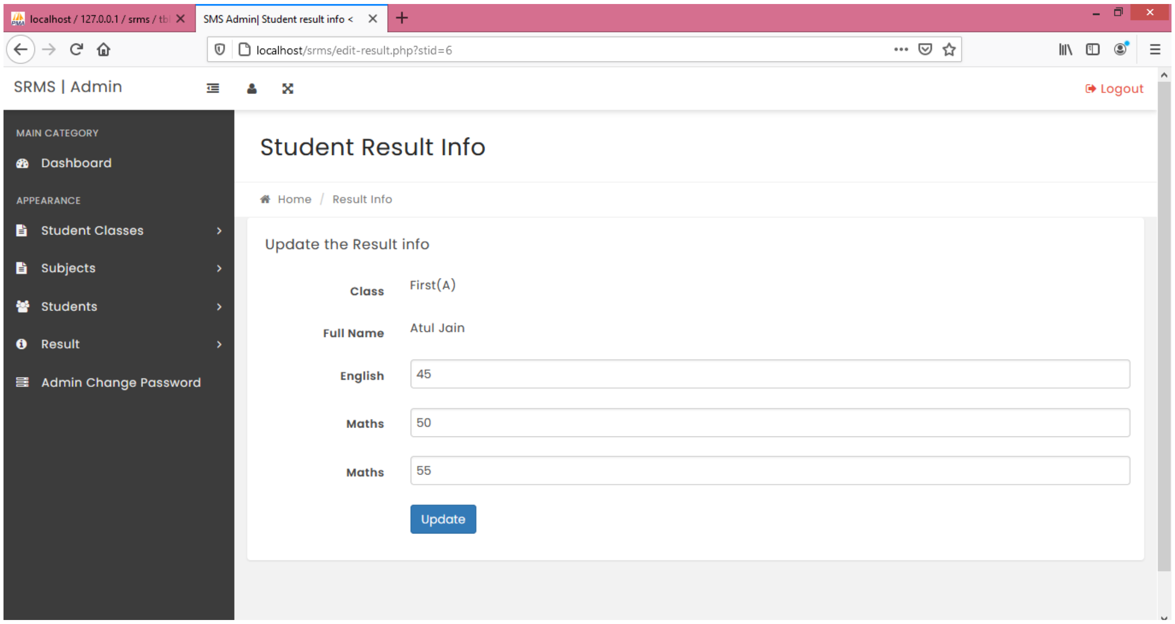
Class: Select Class

Student Name:

Subjects

Declare Result

Manage Result



Student Result Info

Update the Result info

Class First(A)

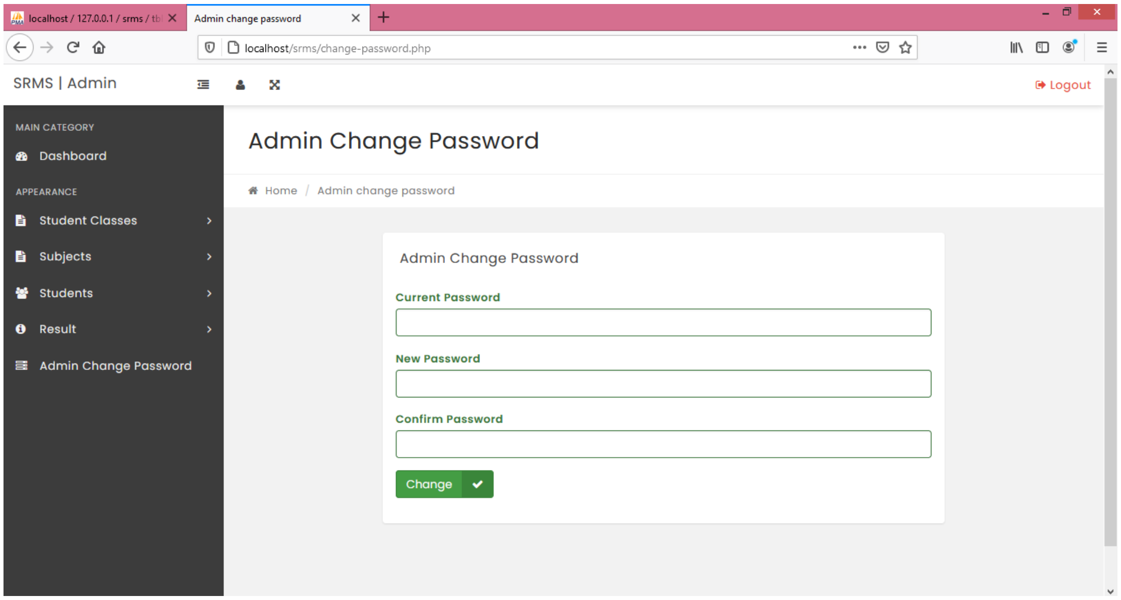
Full Name Atul Jain

English 45

Maths 50

Maths 55

Update Admin Password



Admin Change Password

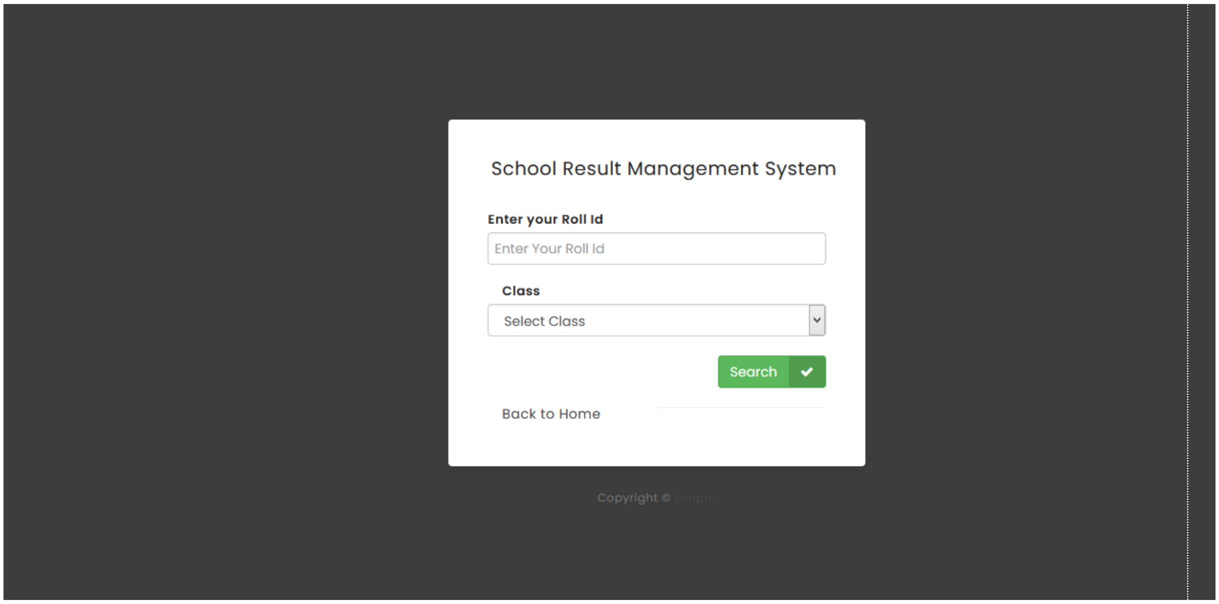
Admin Change Password

Current Password

New Password

Confirm Password

Search Result



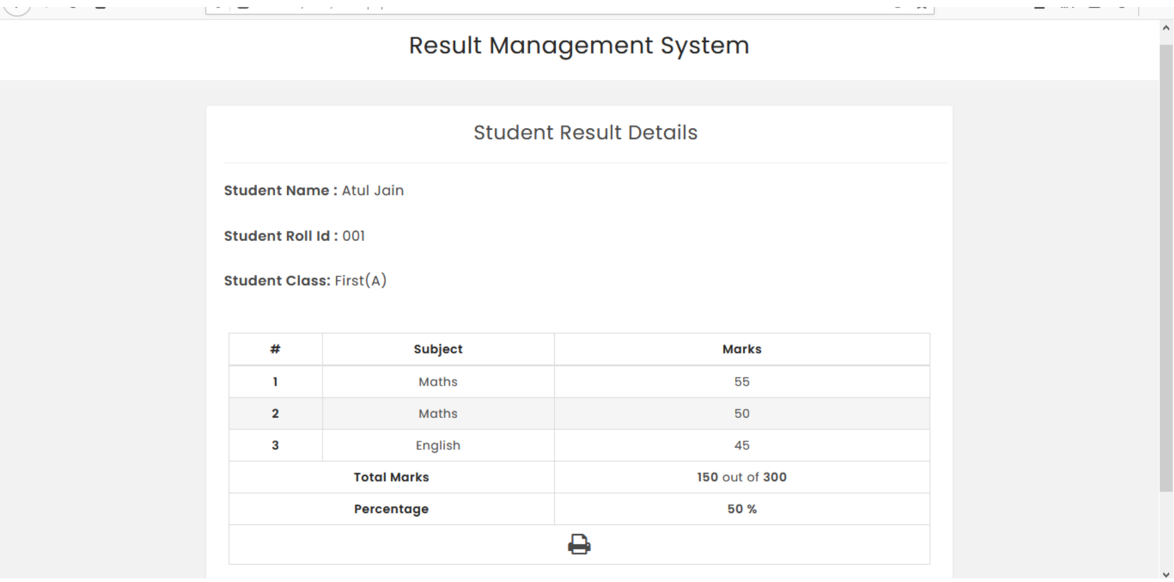
School Result Management System

Enter your Roll Id

Class

Copyright © Gagan

Result Page



Result Management System

Student Result Details

Student Name : Atul Jain

Student Roll Id : 001

Student Class: First(A)

#	Subject	Marks
1	Maths	55
2	Maths	50
3	English	45
Total Marks		**150** out of **300**
Percentage		**50 %**

3.14 Data Dictionary

Sr. No	Field Name	Data-type	Width	Description	Table Name
1	id	integer	11	Id for all	Admin, tblclasses, tblsubjects, tblsubjectcombination, tblresult
2	UserName	varchar	100	For admin	admin
3	Password	varchar	100	For admin	admin
4	updationDate	timestamp		Update Date	admin
5	ClassName	varchar	80	Class Name	tblclasses
6	ClassNameNumeric	integer	4		tblclasses
7	Section	varchar	5	Section of class	tblclasses

8	CreationDate	timestamp		Creating Date	tblclasses
9	UpdationDate	timestamp		Updation Date	tblclasses
10	SubjectName	varchar	100	Subject Name	tblsubjects
11	SubjectCode	varchar	100	Subject Code	tblsubjects
12	Creationdate	timestamp		Creating Date	tblsubjects
13	UpdationDate	timestamp		Updating date	tblsubjects
14	ClassId	integer	11	Class ID	tblsubjectcombination
15	SubjectId	integer	11	Subject ID	tblsubjectcombination, tblresult
16	status	integer	1	Status	tblsubjectcombination, tblstudents

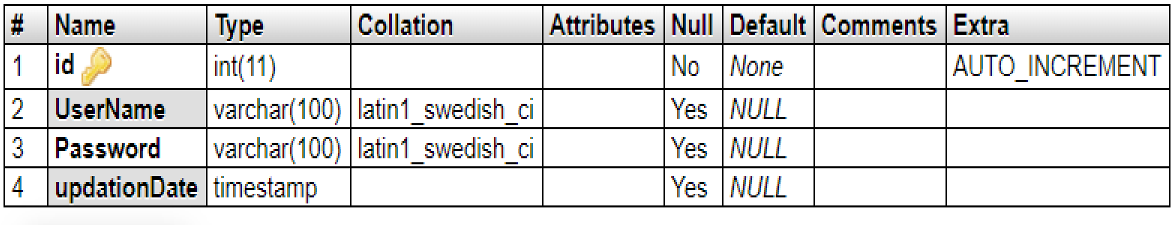
17	CreationDate	timestamp		Creation Date	tblsubjectcombination
18	UpdationDate	timestamp		Update date	tblsubjectcombination
19	StudentId	integer	11	Student ID	tblstudents, tblresult
20	StudentName	varchar	100	Student name	tblstudents
21	RollId	varchar	100	Roll ID	tblstudents
22	StudentEmail	varchar	100	Student Email ID	tblstudents
23	Gender	varchar	10	Gender	tblstudents
24	DOB	varchar	100	Date of birth	tblstudents
25	ClassId	integer	11	Class ID	tblstudents, tblresult

26	RegDate	timestamp		Registration Date	tblstudents
27	UpdationDate	timestamp		Updation Date	tblstudents
28	marks	integer	11	Marks of Student	tblresult
29	PostingDate	timestamp			tblresult
30	UpdationDate	timestamp		Updating Date	tblresult

3.15 Table specifications

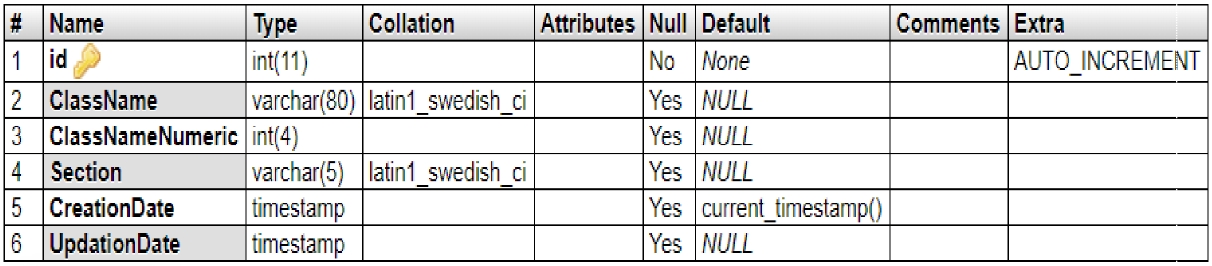
Note :- In all the tables id is primary key and it is using as the foreign key in each table

admin: This tables stores admin login details.



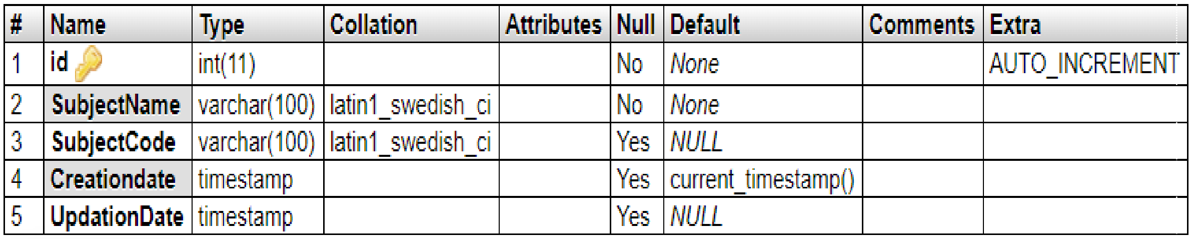
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**id** 🔑	int(11)			No	*None*		AUTO_INCREMENT
2	**UserName**	varchar(100)	latin1_swedish_ci		Yes	*NULL*		
3	**Password**	varchar(100)	latin1_swedish_ci		Yes	*NULL*		
4	**updationDate**	timestamp			Yes	*NULL*		

tblclasses : This tables stores class information.



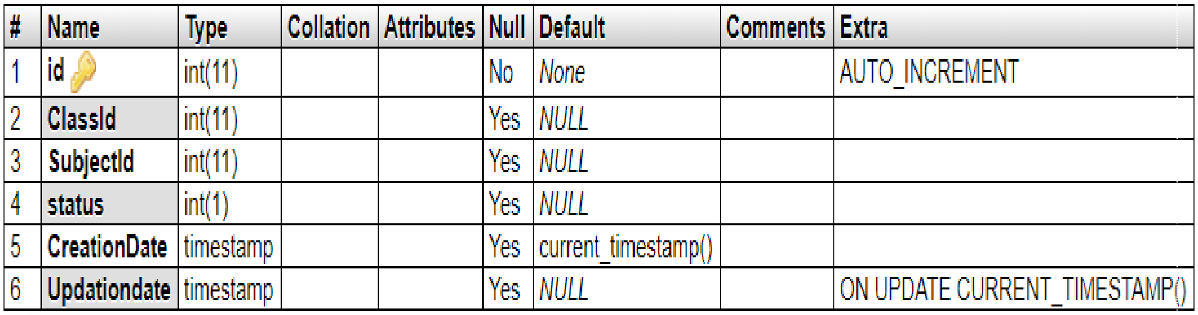
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**id** 🔑	int(11)			No	*None*		AUTO_INCREMENT
2	**ClassName**	varchar(80)	latin1_swedish_ci		Yes	*NULL*		
3	**ClassNameNumeric**	int(4)			Yes	*NULL*		
4	**Section**	varchar(5)	latin1_swedish_ci		Yes	*NULL*		
5	**CreationDate**	timestamp			Yes	current_timestamp()		
6	**UpdationDate**	timestamp			Yes	*NULL*		

tblsubjects: This table store subject details.



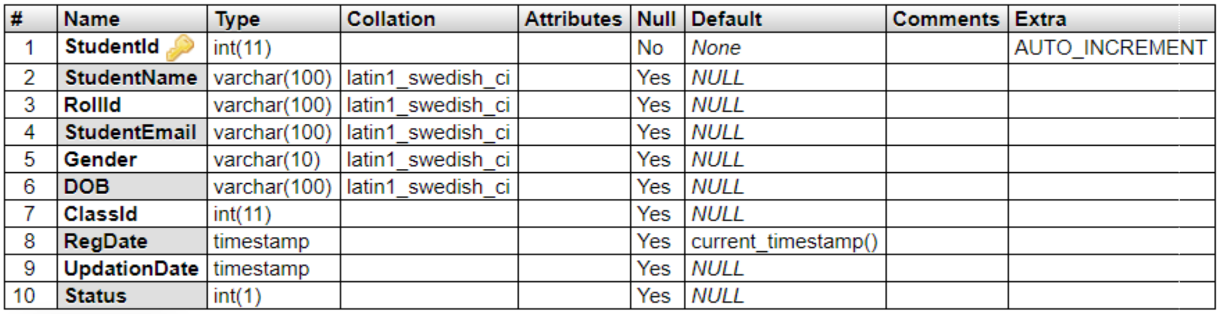
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**id** 🔑	int(11)			No	*None*		AUTO_INCREMENT
2	**SubjectName**	varchar(100)	latin1_swedish_ci		No	*None*		
3	**SubjectCode**	varchar(100)	latin1_swedish_ci		Yes	*NULL*		
4	**Creationdate**	timestamp			Yes	current_timestamp()		
5	**UpdationDate**	timestamp			Yes	*NULL*		

Tblsubjectcombination: This Table stores class and subject combination details.



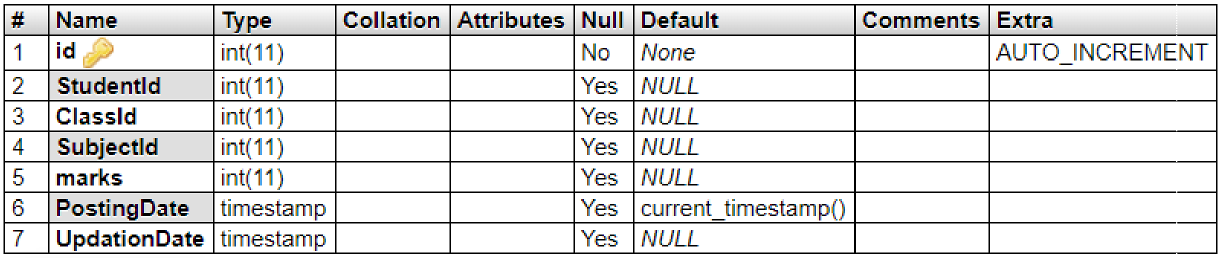
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id 🔑	int(11)			No	*None*		AUTO_INCREMENT
2	ClassId	int(11)			Yes	*NULL*		
3	SubjectId	int(11)			Yes	*NULL*		
4	status	int(1)			Yes	*NULL*		
5	CreationDate	timestamp			Yes	current_timestamp()		
6	Updationdate	timestamp			Yes	*NULL*		ON UPDATE CURRENT_TIMESTAMP()

tblstudents: This table stores student details.



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	StudentId 🔑	int(11)			No	*None*		AUTO_INCREMENT
2	StudentName	varchar(100)	latin1_swedish_ci		Yes	*NULL*		
3	RollId	varchar(100)	latin1_swedish_ci		Yes	*NULL*		
4	StudentEmail	varchar(100)	latin1_swedish_ci		Yes	*NULL*		
5	Gender	varchar(10)	latin1_swedish_ci		Yes	*NULL*		
6	DOB	varchar(100)	latin1_swedish_ci		Yes	*NULL*		
7	ClassId	int(11)			Yes	*NULL*		
8	RegDate	timestamp			Yes	current_timestamp()		
9	UpdationDate	timestamp			Yes	*NULL*		
10	Status	int(1)			Yes	*NULL*		

tblresult : This stores the result details.



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id 🔑	int(11)			No	*None*		AUTO_INCREMENT
2	StudentId	int(11)			Yes	*NULL*		
3	ClassId	int(11)			Yes	*NULL*		
4	SubjectId	int(11)			Yes	*NULL*		
5	marks	int(11)			Yes	*NULL*		
6	PostingDate	timestamp			Yes	current_timestamp()		
7	UpdationDate	timestamp			Yes	*NULL*		

3.16 Test Procedures and Implementation

Testing is a process of executing a program with the intent of finding an error. Testing is a crucial element of software quality assurance and presents ultimate review of specification, design and coding.

System Testing is an important phase. Testing represents an interesting anomaly for the software. Thus a series of testing are performed for the proposed system before the system is ready for user acceptance testing.

A good test case is one that has a high probability of finding an as undiscovered error. A successful test is one that uncovers an as undiscovered error.

Testing Objectives:

1. Testing is a process of executing a program with the intent of finding an error
2. A good test case is one that has a probability of finding an as yet undiscovered error
3. A successful test is one that uncovers an undiscovered error

Testing Principles:

- All tests should be traceable to end user requirements
- Tests should be planned long before testing begins
- Testing should begin on a small scale and progress towards testing in large
- Exhaustive testing is not possible
- To be most effective testing should be conducted by a independent third party

The primary objective for test case design is to derive a set of tests that has the highest livelihood for uncovering defects in software. To accomplish this objective two different categories of test case design techniques are used. They are

- White box testing.
- Black box testing.

White-box testing:

White box testing focus on the program control structure. Test cases are derived to ensure that all statements in the program have been executed at least once during testing and that all logical conditions have been executed.

Block-box testing:

Black box testing is designed to validate functional requirements without regard to the internal workings of a program. Black box testing mainly focuses on the information domain of the software, deriving test cases by partitioning input and output in a manner that provides through test coverage. Incorrect and missing functions, interface errors, errors in data structures, error in functional logic are the errors falling in this category.

Testing strategies:

A strategy for software testing must accommodate low-level tests that are necessary to verify that all small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements.

Testing fundamentals:

Testing is a process of executing program with the intent of finding error. A good test case is one that has high probability of finding an undiscovered error. If testing is conducted successfully it uncovers the errors in the software. Testing cannot show the absence of defects, it can only show that software defects present.

Testing Information flow:

Information flow for testing flows the pattern. Two class of input provided to test the process. The software configuration includes a software requirements specification, a design specification and source code.

Test configuration includes test plan and test cases and test tools. Tests are conducted and all the results are evaluated. That is test results are compared with expected results. When erroneous data are uncovered, an error is implied and debugging commences.

Unit Testing:

Unit testing is essential for the verification of the code produced during the coding phase and hence the goal is to test the internal logic of the modules. Using the detailed design description as a guide, important paths are tested to uncover errors with in the boundary of the modules. These tests were carried out during the programming stage itself. All units of ViennaSQLwere successfully tested.

Integration testing :

Integration testing focuses on unit tested modules and build the program structure that is dictated by the design phase.

System testing:

System testing tests the integration of each module in the system. It also tests to find discrepancies between the system and it's original objective, current specification and system documentation. The primary concern is the compatibility of individual modules. Entire system is working properly or not will be tested here, and specified path ODBC connection will correct or not, and giving output or not are tested here these verifications and validations are done by giving input values to the system and by comparing with expected output. Top-down testing implementing here.

Acceptance Testing:

This testing is done to verify the readiness of the system for the implementation. Acceptance testing begins when the system is complete. Its purpose is to provide the end user with the confidence that the system is ready for use. It involves planning and execution of functional tests, performance tests and stress tests in order to demonstrate that the implemented system satisfies its requirements.

Tools to special importance during acceptance testing include:

Test coverage Analyzer – records the control paths followed for each test case.

Timing Analyzer – also called a profiler, reports the time spent in various regions of the code are areas to concentrate on to improve system performance.

Coding standards – static analyzers and standard checkers are used to inspect code for deviations from standards and guidelines.

3.17 Test Cases:

Test cases are derived to ensure that all statements in the program have been executed at least once during testing and that all logical conditions have been executed.

Using White-Box testing methods, the software engineer can drive test cases that

- Guarantee that logical decisions on their true and false sides.
- Exercise all logical decisions on their true and false sides.
- Execute all loops at their boundaries and with in their operational bounds.

TEST CASE	SCENARIO	STEPS	EXPECTED RESULT	ACTUAL RESULT	PASS /

- Exercise internal data structure to assure their validity.

The test case specification for system testing has to be submitted for review before system testing commences.

ID		TO PERFORM			FAIL
1	Log-in application Admin Head	1. Open the log in page of the application. 2. Enter the valid user name. 3. Enter valid password. 4. Click on Log in button.	Application should except valid user name and valid password entered by user and should redirect user to respected dashboard.	Log in into Application is successful	Pass
2	Search the Result	1.Open the log in page of the application. 2.Enter the valid roll Id 3.Enter valid Class.	Application should Show the Result with the Respected Fields.	Showing the Result	Pass

		4.Click on Log in button.			
3	Show the Dashboard with Different Data		Application Should show the Count of different Data	Show the right count	Pass
4	Create Class	1.Click on Student Classes 2.Click on Create Class 3.Enter class name, class Name in Numeric , Section 4.Click on Submit	Application should Create a class	Creating Class	Pass
5.	Update the existing Class	1.Click on Student Classes 2.Click on Manage Classes	Update The Class Details	Updating The Class Details	Pass

		3.Click on Action Button of Specific 4.Enter class name, class Name in Numeric , Section 5. Click on Submit			
6.	Searching Functionality on Every respective Page	1.Click on search bar 2.Enter the Characters which we want to select	Search the Content	Searching Perfectly	Pass
7.	Create Subject	1.Click on Subjects 2.Click on Create Subject 3.Enter subject name, Subject Code	Application should Create a Subject	Creating Subject	Pass

		4.Click on Submit			
8.	Update the existing Subject	1.Click on Subjects 2.Click on Manage Subjects 3.Click on Action Button of Specific Record 4. Enter subject name, Subject Code 5. Click on Submit	Update The Subject Details	Updating The Subject Details	Pass
9.	Add Subject Combination	1.Click on Subject 2.Click on Add Subject Combination	Application should Add Particular Subject to Particular Class	Creating Subject and Class Combination	Pass

		3.Enter Class and subject 4.Click on Add			
10.	Update Subject Combination	1.Click on Subject 2.Click on Manage Subject Combination 3.Click on Action Button of Specific Record to activate and deactivate	Activate or Deactivate the Subject Combination	Activating or Deactivating the Subject Combination	Pass
11.	Add Student	1.Click on Students 2.Click on Add Student 3.Enter Full	Application should Add a student	Adding Student	Pass

		Name, Roll ID, Email ID, Gender, Class and DOB 4.Click on Add			
12.	Update the existing Student	1.Click on Students 2.Click on Manage Students 3.Click on Action Button of Specific Record 4.Enter Full Name, Roll ID, Email ID, Gender, Class and DOB and Status 5.Click on Add	Update The Student Details	Updating The Student Details	Pass
13.	Add Result	1.Click on	Application	Adding	Pass

		Result 2.Click on Add Result 3.Select Class and Student Name 4.Enter the Subject Marks 5.Click on Add	should Add a Result	Result	
14.	Check If result is Already declared	1.Click on Result 2.Click on Add Result 3.Select Class and Student Name	Application should show Result Already Declare.	Showing the message.	Pass
15.	Update the existing Result	1.Click on Result 2.Click on	Update The Student Result	Updating The Student Result	Pass

		Add Result 3.Click on Action Button of Specific Record 4.Enter the Subject Marks 5.Click on Update			
16.	Admin Change Password	1.Click on Admin Change Password 2.Enter Current Password , New Password and Confirm Password 3.Click on change	Application should Update the Password	Updating the Password	Pass
17.	Printing the Student	1.Open the log in page	Application should Show	Showing the Result	Pass

	Result/Generating the .pdf file	of the application. 2.Enter the valid roll Id 3.Enter valid Class. 4.Click on Log in button. 5.Click On Print Icon	the Result with the Respected Fields and after clicking on the Print Icon Should Print the result	and Printing or Generating the .pdf	

CHAPTER 4

USER MANUAL

User Manual

For any system to be successful it is important that the intended user find the system easy to operate. The purpose of the user manual is to make user acquainted with the system and help user understand the system and operate it conveniently. The User Manual is prepared reflexively because it is an item that must accompany every system.

The manual contain several screenshots that describes how to use the entire system. This Manual helps user to navigate efficiently through the system and help user to solve issues wherever they occur.
Information about the system.

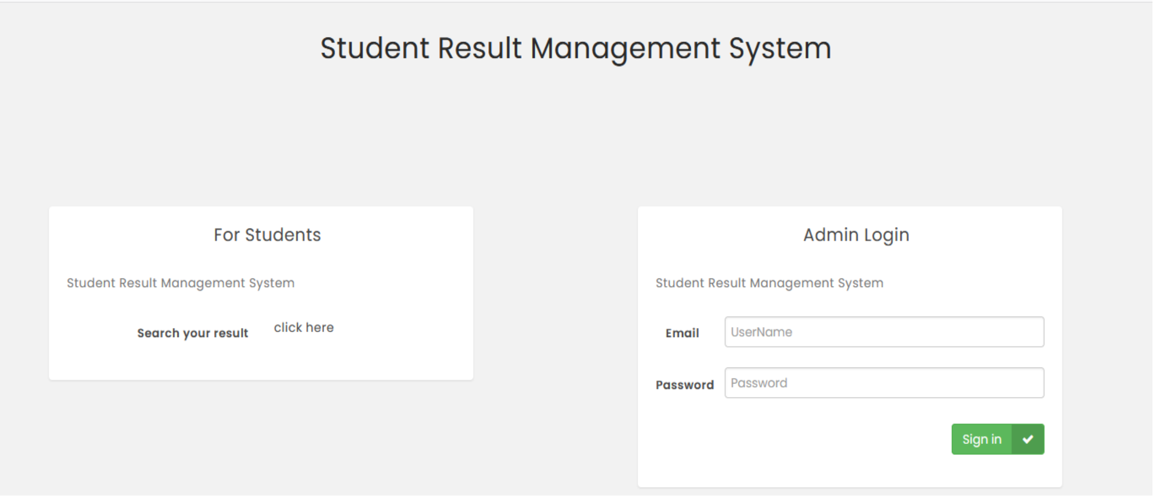
The system contains following users:

1) Admin
2) Student

The System has following features -

- Admin can add/update/ Class explain them
- Admin can add/update/ Subjects
- Admin can add/update/ Active/Inactive Subject combination with class

- Admin can register new student and also edit info of the student
- If the Result is Already Declared then it won’t allow to declare with the same.
- Product and Component based.
- Creating and changing issues at ease.
- Query issue list to any depth
- Reporting and charting in more comprehensive way
- User accounts to control the access and maintain security
- Simple status and resolutions



Student Result Management System

For Students

Student Result Management System

Search your result click here

Admin Login

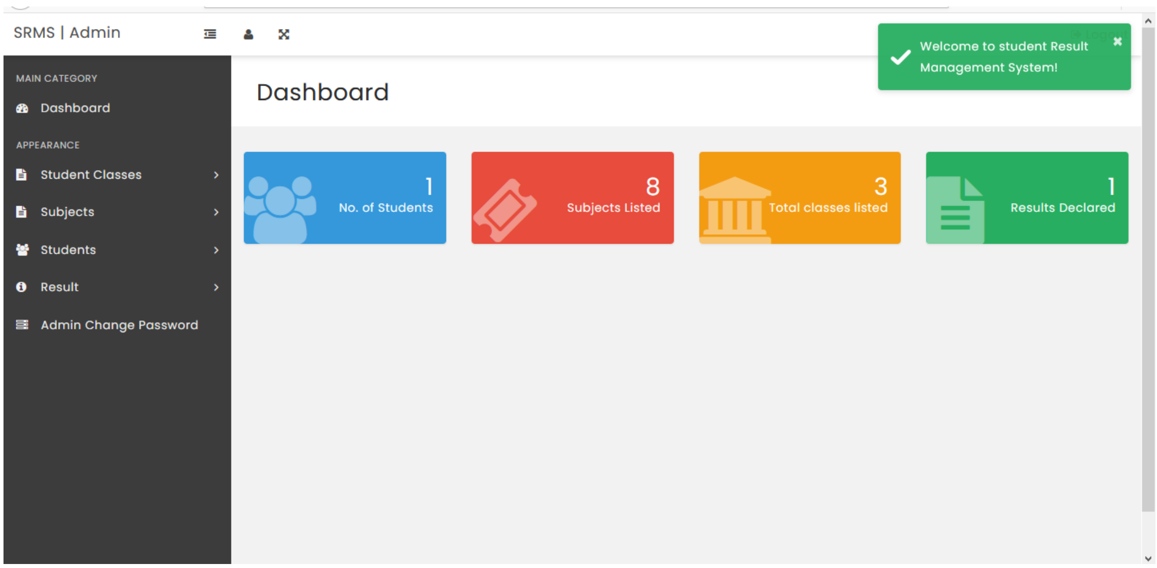
Student Result Management System

Email UserName

Password Password

This is the Home Screen where we can select the Different Modules

You are redirected to dashboard after successful credential verification of Username and Password of Admin Module



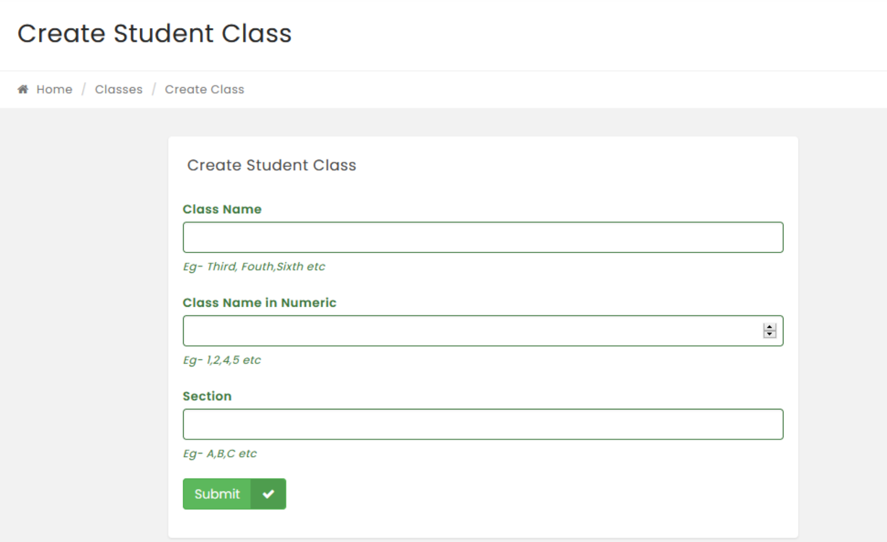
Welcome to student Result Management System!

Here We can see different Functionalities:-

a. Template Show the Data of respective Field

b. Admin can click on them which will redirect you to the respective Page

c. Admin can select the different categories



Create Student Class

🏠 Home / Classes / Create Class

Create Student Class

Class Name

Eg- Third, Fouth,Sixth etc

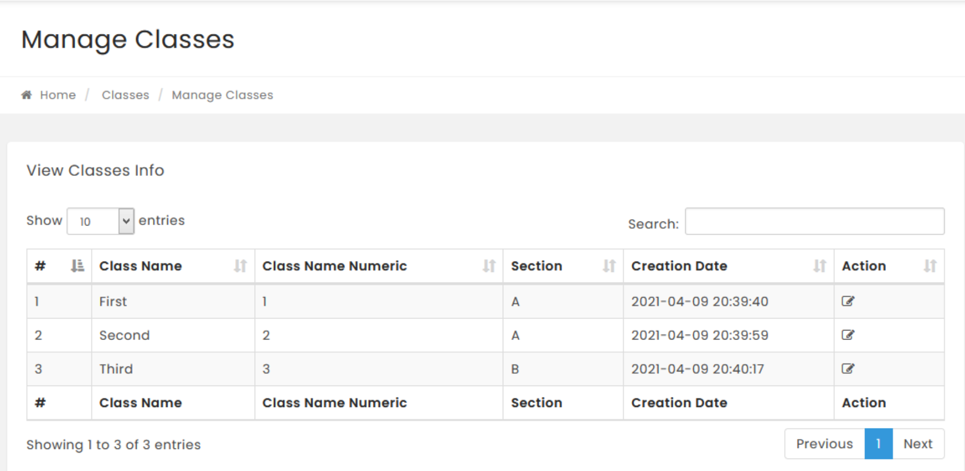
Class Name in Numeric

Eg- 1,2,4,5 etc

Section

Eg- A,B,C etc

Here Admin Expected to enter data into the Fields and It will create the class



Manage Classes

🏠 Home / Classes / Manage Classes

View Classes Info

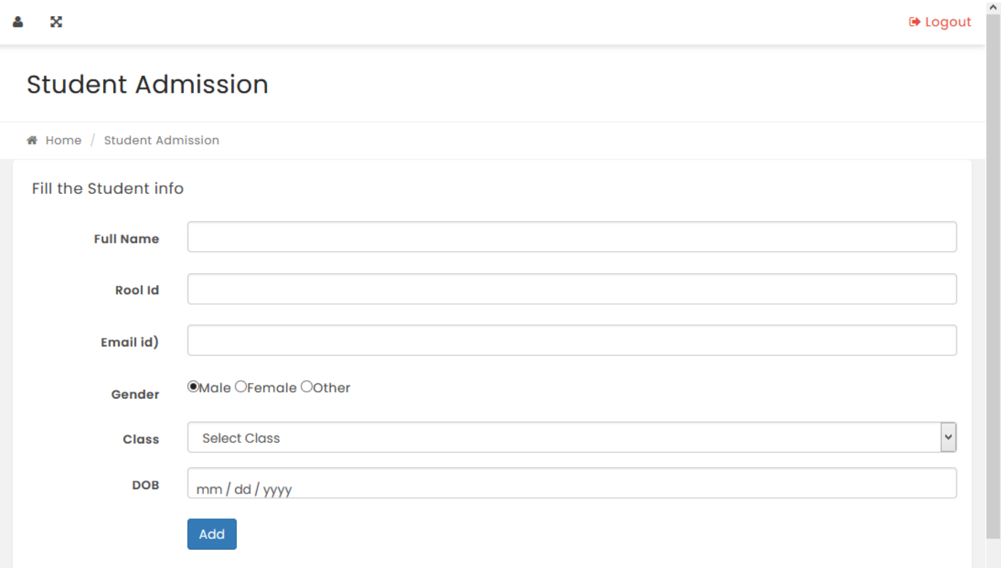
Show 10 entries

Search:

#	Class Name	Class Name Numeric	Section	Creation Date	Action
1	First	1	A	2021-04-09 20:39:40	✎
2	Second	2	A	2021-04-09 20:39:59	✎
3	Third	3	B	2021-04-09 20:40:17	✎
#	Class Name	Class Name Numeric	Section	Creation Date	Action

Showing 1 to 3 of 3 entries

Admin Can Update the existing Classes by Clicking on ✎ and can update the record.



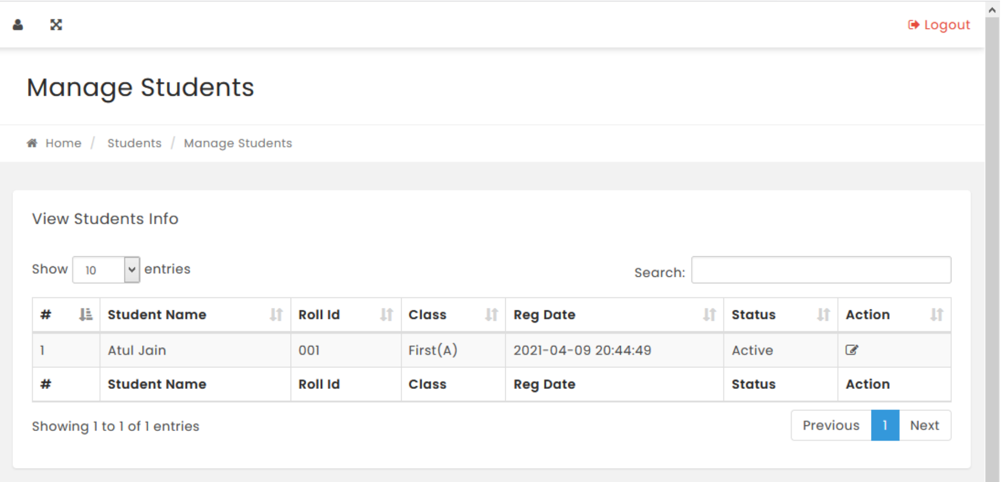
Student Admission

Home / Student Admission

Fill the Student info

Full Name	
Rool Id	
Email id)	
Gender	●Male ○Female ○Other
Class	Select Class
DOB	mm / dd / yyyy

Here Admin Expected to enter data into the Fields and It will create New Student.



Manage Students

Home / Students / Manage Students

View Students Info

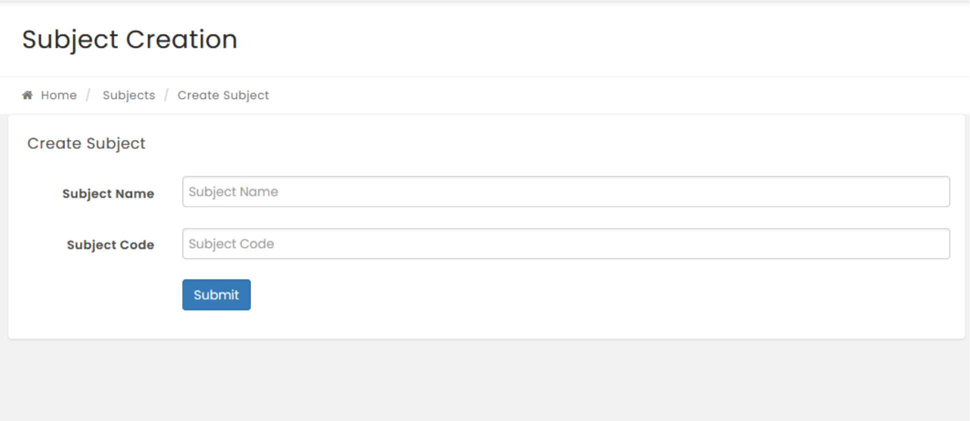
Show 10 entries

Search:

#	Student Name	Roll Id	Class	Reg Date	Status	Action
1	Atul Jain	001	First(A)	2021-04-09 20:44:49	Active	✎
#	Student Name	Roll Id	Class	Reg Date	Status	Action

Showing 1 to 1 of 1 entries

Admin Can Update the existing Students by Clicking on ✎ and can update the record.



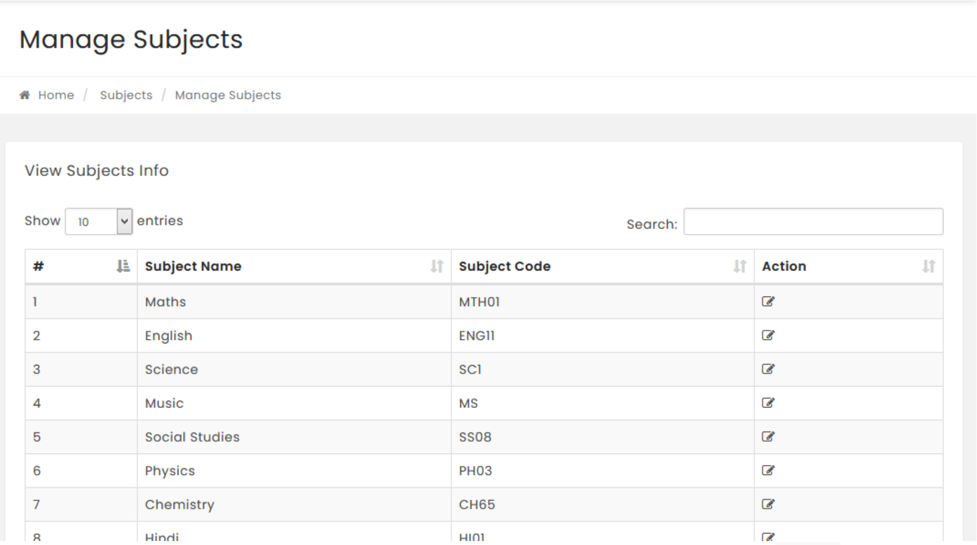
Subject Creation

Home / Subjects / Create Subject

Create Subject

Subject Name	Subject Name
Subject Code	Subject Code

Here Admin Expected to enter data into the Fields and It will create New Subject



Manage Subjects

Home / Subjects / Manage Subjects

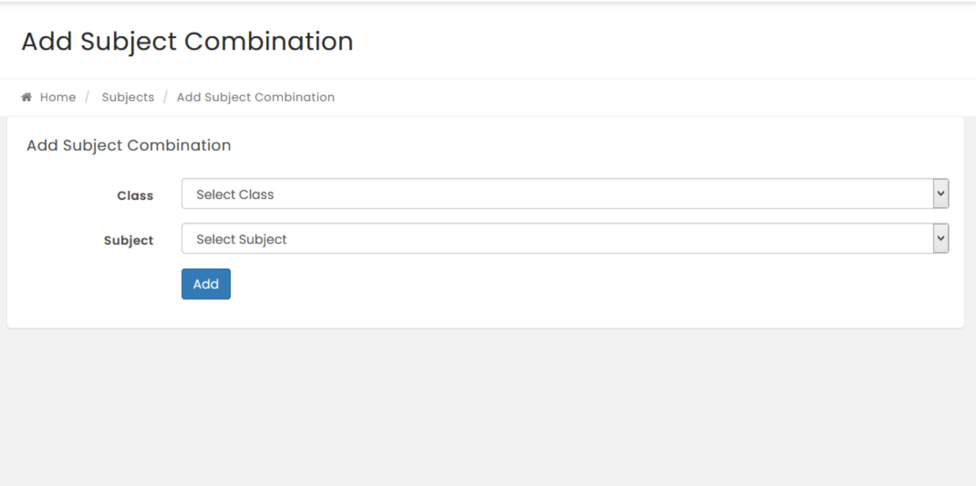
View Subjects Info

Show 10 entries

Search:

#	Subject Name	Subject Code	Action
1	Maths	MTH01	✎
2	English	ENG11	✎
3	Science	SC1	✎
4	Music	MS	✎
5	Social Studies	SS08	✎
6	Physics	PH03	✎
7	Chemistry	CH65	✎
8	Hindi	HI01	✎

Admin Can Update the existing Subject by Clicking on ✎ and can update the record.



Add Subject Combination

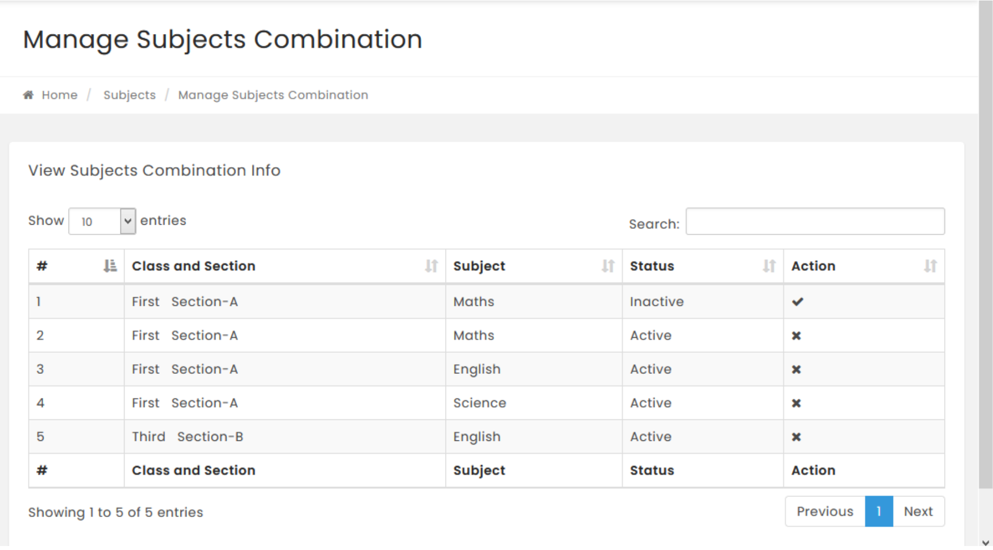
Home / Subjects / Add Subject Combination

Add Subject Combination

Class: Select Class

Subject: Select Subject

Here Admin Expected to Select Class and Subject and it will create Subject and Class Combination.



Manage Subjects Combination

Home / Subjects / Manage Subjects Combination

View Subjects Combination Info

Show 10 entries

Search:

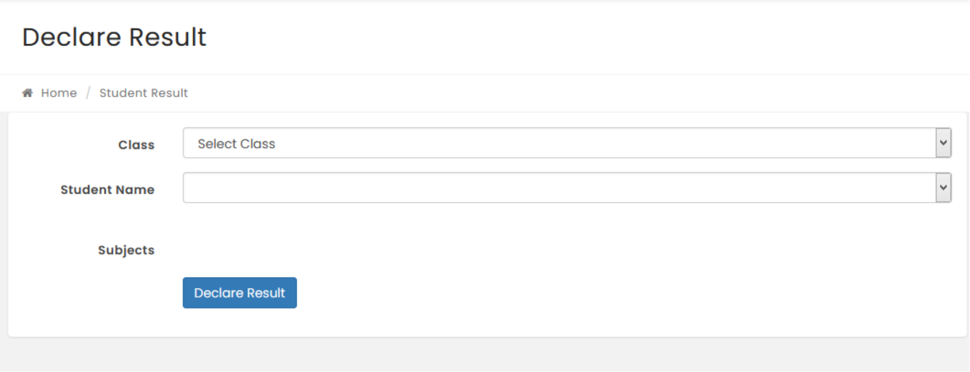
#	Class and Section	Subject	Status	Action
1	First Section-A	Maths	Inactive	✓
2	First Section-A	Maths	Active	✖
3	First Section-A	English	Active	✖
4	First Section-A	Science	Active	✖
5	Third Section-B	English	Active	✖
#	Class and Section	Subject	Status	Action

Showing 1 to 5 of 5 entries

Admin can update the existing Subject and Class Combination by Clicking on

✓ Will activate the Combination and

✖ Will deactivate the Combination and update the record


Declare Result

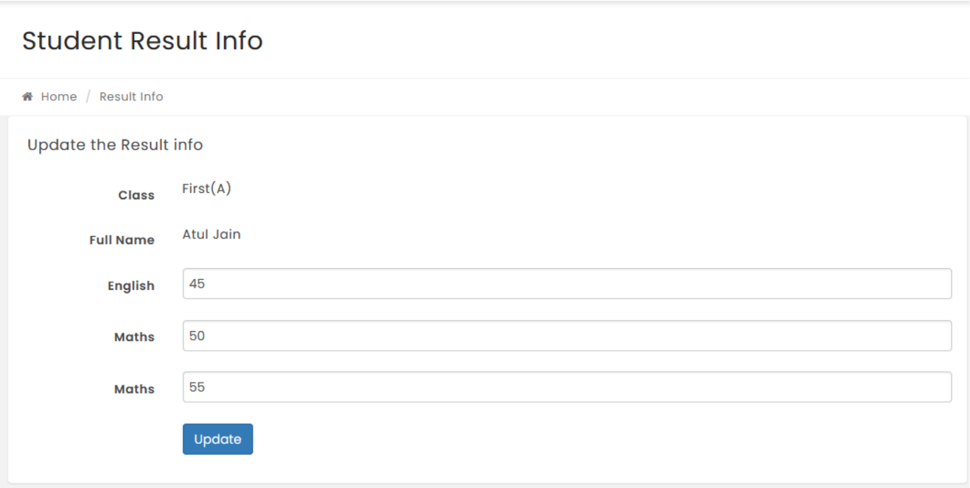
Home / Student Result

Class: Select Class

Student Name:

Subjects

Here Admin Expected to enter data into the Fields and It will create Result for Student.


Student Result Info

Home / Result Info

Update the Result info

Class: First(A)

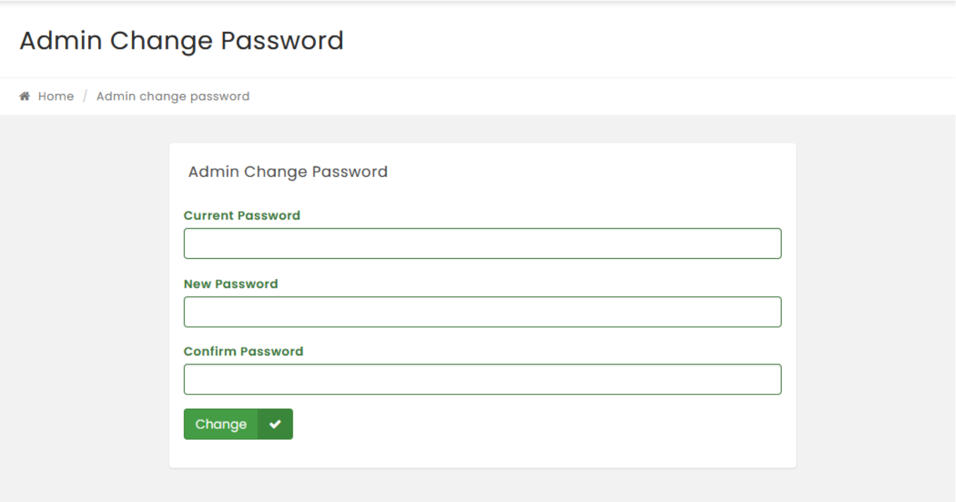
Full Name: Atul Jain

English: 45

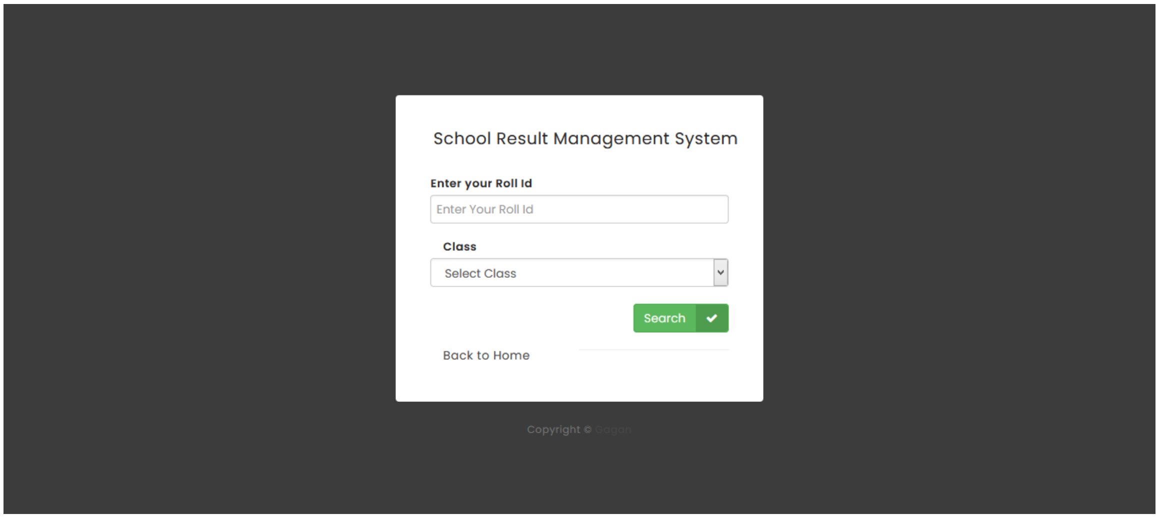
Maths: 50

Maths: 55

Admin can update the existing Result.

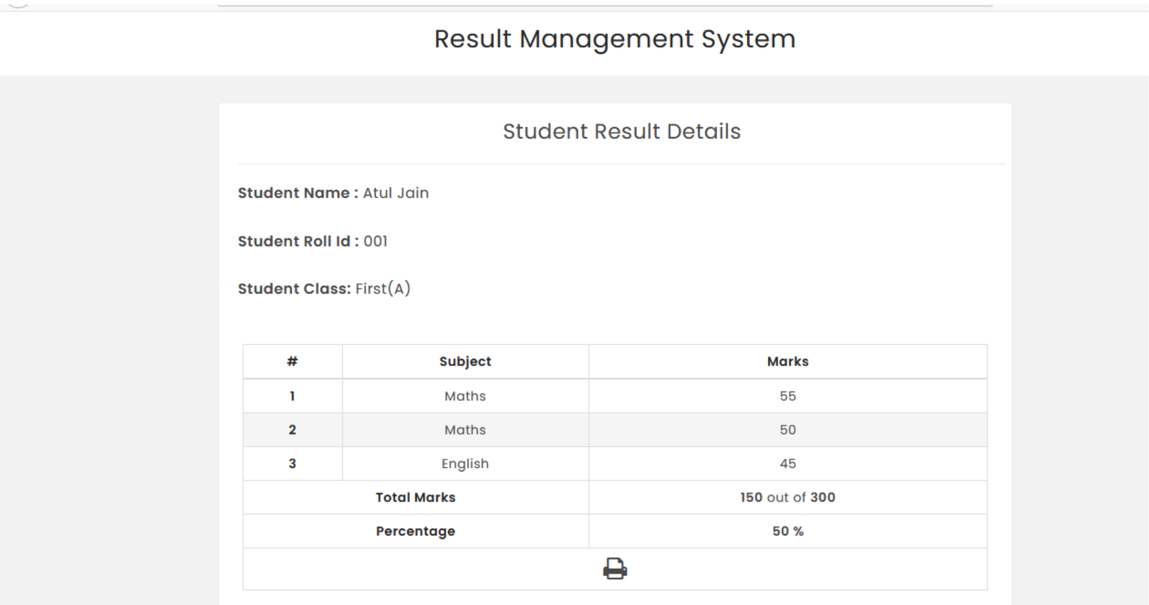


Admin can update its password on this Screen by Entering the Valid Credentials.



Student Can Search Result here with valid Credentials.

S



Result Management System

Student Result Details

Student Name : Atul Jain

Student Roll Id : 001

Student Class: First(A)

#	Subject	Marks
1	Maths	55
2	Maths	50
3	English	45
	Total Marks	**150** out of **300**
	Percentage	**50 %**
	🖨	

Student can see Result like this and Can Print or Generate the .pdf

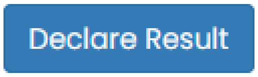
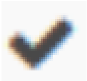
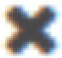
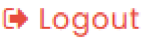
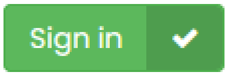
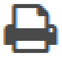
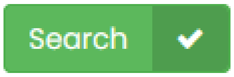
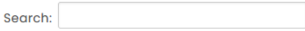
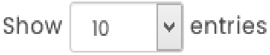
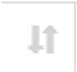
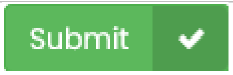
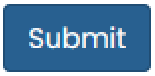
on clicking 🖨

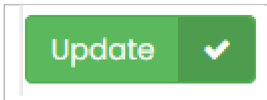
4.2 Operations Manual

There are various symbols and buttons on the

web application. Here are their descriptions:

Symbol	Meaning
✎	Action On Edit/Update Record
Add	Add the record
Dashboard	Go to dashboard

	Declare the Result
	Activate the Class Subject Combination
	Deactivate the Class Subject Combination
	Logout Button
	Sign Button
	Print the Result
	Search Result
	Search the record
	Show No of entries on single Page
	Sort the Entries
	Submit/Create the Class
	Submit/Create the Subject

	Update The Record

4.3 Program Specifications

1. Add Student Classes

Module	Admin
Program Name	Add Student Classes
Purpose	Adding Student Class to tblClasses Table , if admin wants to create new Class
Input Details	The required fields should not be blank and the user should provide valid data for each field.
Output	The details of the class are stored in the tblClasses table.

2. Update Student Classes

Module	Admin
Program Name	Update Student Classes
Purpose	Updating Student Class to tblClasses Table , if admin wants to update existing Class
Input Details	The required fields should not be blank and the user should provide valid data for each field.
Output	The details of the class are stored in the tblClasses table.

3. Add Student Subject

Module	Admin
Program Name	Add Subject
Purpose	Adding Subject to tblsubjects Table , if admin wants to create new Subject
Input Details	The required fields should not be blank And the user should provide valid data for each field.
Output	The details of the subject are stored in the tblsubjects table.

4. Update Student Subject

Module	Admin
Program Name	Update Subject
Purpose	Updating Subject to tblsubjects Table , if admin wants to update existing Subject
Input Details	The required fields should not be blank and the user should provide valid data for each field.
Output	The details of the Subject are stored in the tblsubjects table.

5. Add Subject Class Combination

Module	Admin
Program Name	Add Subject Class Combination
Purpose	Adding Subject Class Combination to tblsubjectcombination Table , if admin wants to create new Class
Input Details	The required fields should not be blank and the user should provide valid data for each field.
Output	The details of the Subject Class Combination are stored in the tblsubjectcombination table.

6. Update Status of Subject Class Combination

Module	Admin
Program Name	Update Status of Subject Class Combination
Purpose	Updating Subject Class Combination to tblsubjectcombination Table , if admin wants to update existing Subject Class Combination
Input Details	The required fields should not be blank and the user should provide valid data for each field.
Output	The Status of the Subject Class Combination are stored in the tblsubjectcombination table.

7. Add Student

Module	Admin
Program Name	Add Student
Purpose	Adding Student to tblstudents Table , if admin wants to create new Student
Input Details	The required fields should not be blank And the user should provide valid data for each field.
Output	The details of the student are stored in the tblstudents table.

8. Update Student

Module	Admin
Program Name	Update Student
Purpose	Updating Student to tblstudents Table , if admin wants to update existing Student
Input Details	The required fields should not be blank and the user should provide valid data for each field.
Output	The details of the Student are stored in the tblstudents table.

9. Add Result

Module	Admin
Program Name	Add Result
Purpose	Adding Result to tblresult Table , if admin wants to create new Result
Input Details	The required fields should not be blank And the user should provide valid data for each field.
Output	The details of the Result are stored in the tblresult table.

10. Update Student

Module	Admin
Program Name	Update Result
Purpose	Updating Result to tblresult Table , if admin wants to update existing Result
Input Details	The required fields should not be blank and the user should provide valid data for each field.
Output	The details of the Result are stored in the tblresult table.

11. Change Admin Password

Module	Admin
Program Name	Change Admin Password
Purpose	Changing Admin Password to admin Table , if admin wants to change existing password.
Input Details	The required fields should not be blank And the user should provide valid data for each field.
Output	The details of the Admin Password are stored in the admin table.

12. Search Result

Module	Student
Program Name	Search Result
Purpose	Searching Result to tblresult Table , if student wants to search Result.
Input Details	The required fields should not be blank and the user should provide valid data for each field.
Output	The details from tblresult Table are shown and can print the result .

Drawbacks and Limitations

- Just like any other technical system, there may be occasional glitches in the online result management system. These may vary from small server problems to a larger. However, these are challenges that can be easily managed to ensure that the Student result management system is not affected by these.

- Considerable effort has made the software easy to operate even for the people not related to the field of computers but it is acknowledged that a layman may find it a bit problematic at the first instance. The user is provided help at each step of his convenience in working with the software.

- Though the software presents a broad range of option to its users some intricate options could not be covered into it, party because of logistics and partly due to lack of sophistication. Paucity of time was also major constraints and due to security the Admin Forgot functionality is not available.

- Considerable effort has made the software easy to operate even for the people not related to the field of computers but it is acknowledged that a layman may find it a bit problematic at the first instance. The user is provided help at each step of his convenience in working with the software.

- Excel export has not been developed for student, result due to some criticality.

- The transactions are executed in offline mode hence online data for subjects, class capture and modification is not possible.

Proposed Enhancements:

In a nutshell, it can be summarized that the future scope of the project circles around maintaining information regarding:

- We can add Forgot Password functionalities.

- We can give more advanced software for Student Result Management System including more facilities.
- We will host the platform on online servers to make it accessible worldwide.
- Integrate multiple load balancers to distribute the loads of the system.
- Create the master and slave database structure to reduce the overload of the database queries.
- Implement the backup mechanism for taking backup of code base and database on regular basis on different servers.

Conclusions:

This project is humble venture to satisfy the needs to manage their project work. Several user-friendly coding have also adopted. This package shall prove to be a powerful package in satisfying all the requirements of the school. The objective of software planning is to provide a framework that enables the admin to make reasonable estimates made within a limited time frame at the beginning of the software project and should be updated regularly as the project progresses.

- A description of the background and other context of the project and its relation to work already done in the area.
- Made statement of the aims and objectives of the project.
- The description of Purpose, Scope, and applicability.
- Need define the problem on which we are working on the project.
- We describe the requirements specifications of the system and the actions that can be done on these things.
- System understands the problem domain and produces a model of the system, which describes operations that can be performed on the system.
- We included features and operations in detail, including screen layouts.
- We designed user interface and security issues related to system.

- Finally the system is implemented and tested according to test cases.

Bibliography:

- Google for problem Solving
- PHP and MySQL Web Development

  Book by Luke Welling

- Head First PHP & MySQL

  Book by Lynn Beighley and Michael Morrison

- PHP & MySQL for Dummies

  Book by Janet Valade

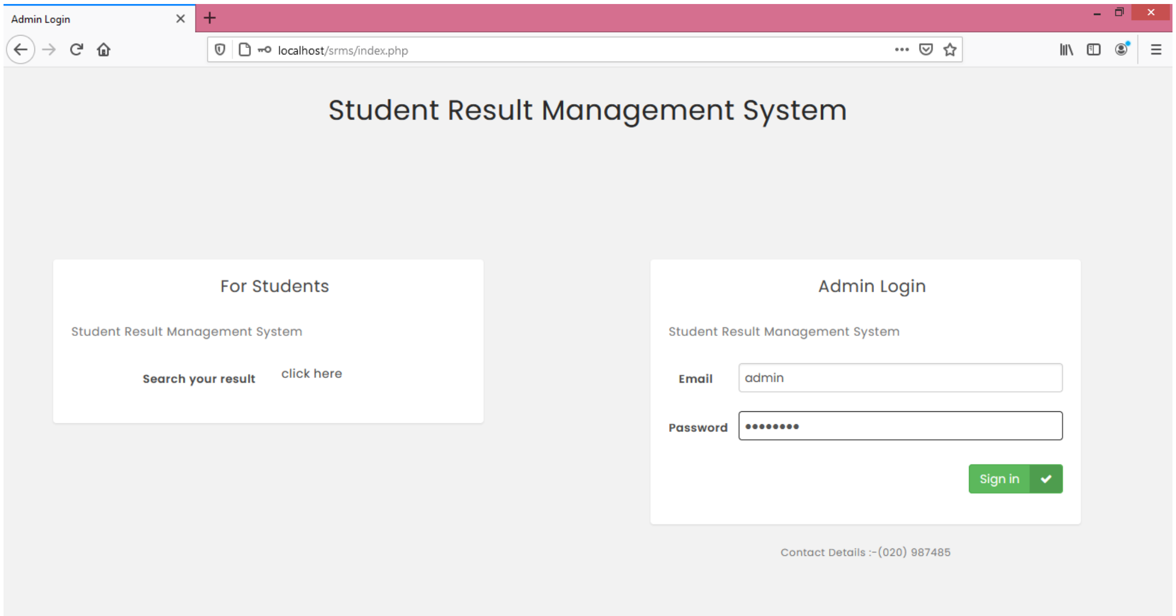
Websites:

- www.google.com
- www.youtube.com
- www.w3schools.com
- www.tutorialspoint.php
- www.stackoverflow.com

ANNEXURE 1

USER INTERFACE SCREEN

1. Login Page



Student Result Management System

For Students

Student Result Management System

Search your result

Admin Login

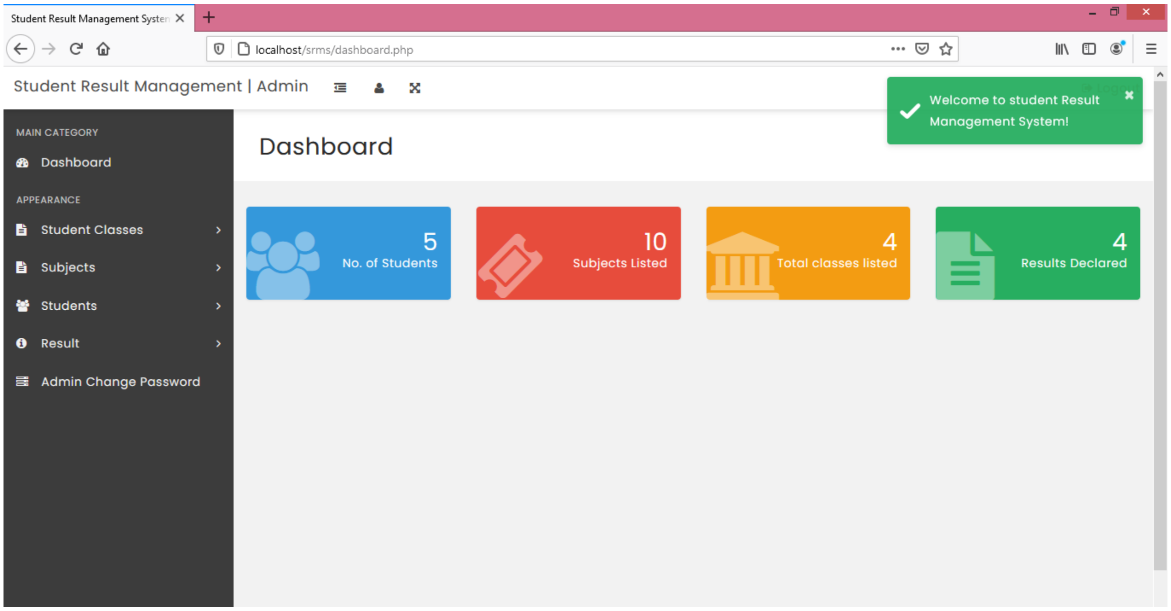
Student Result Management System

Email admin

Password ••••••••

Contact Details :-(020) 987485

2. Dashboard


Student Result Management Syster
localhost/srms/dashboard.php
Student Result Management | Admin
MAIN CATEGORY
Dashboard
APPEARANCE
Student Classes
Subjects
Students
Result
Admin Change Password

Welcome to student Result Management System!

Dashboard

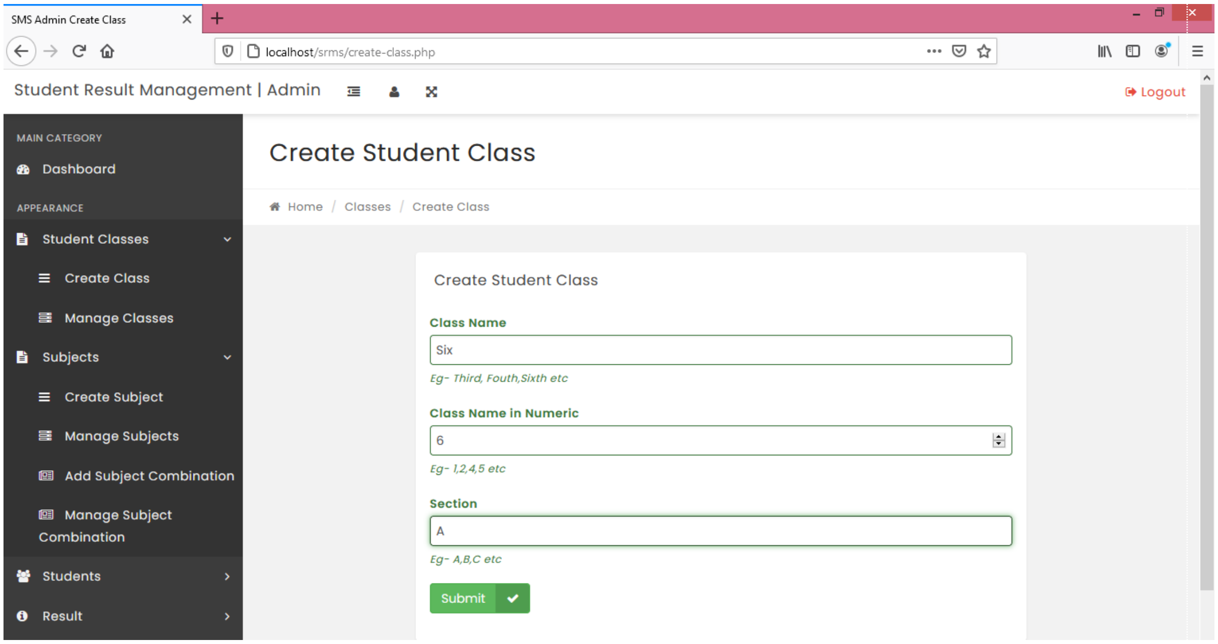
5 No. of Students

10 Subjects Listed

4 Total classes listed

4 Results Declared

3.Create Class



Create Student Class

Create Student Class

Class Name: Six

Eg- Third, Fouth,Sixth etc

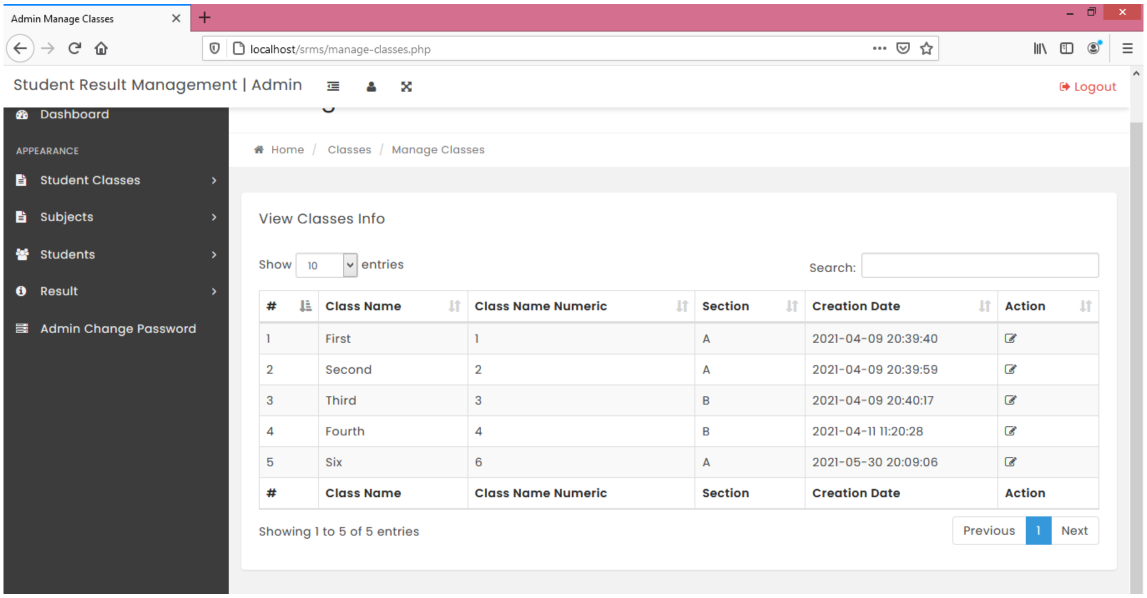
Class Name in Numeric: 6

Eg- 1,2,4,5 etc

Section: A

Eg- A,B,C etc

4.Manage Class



View Classes Info

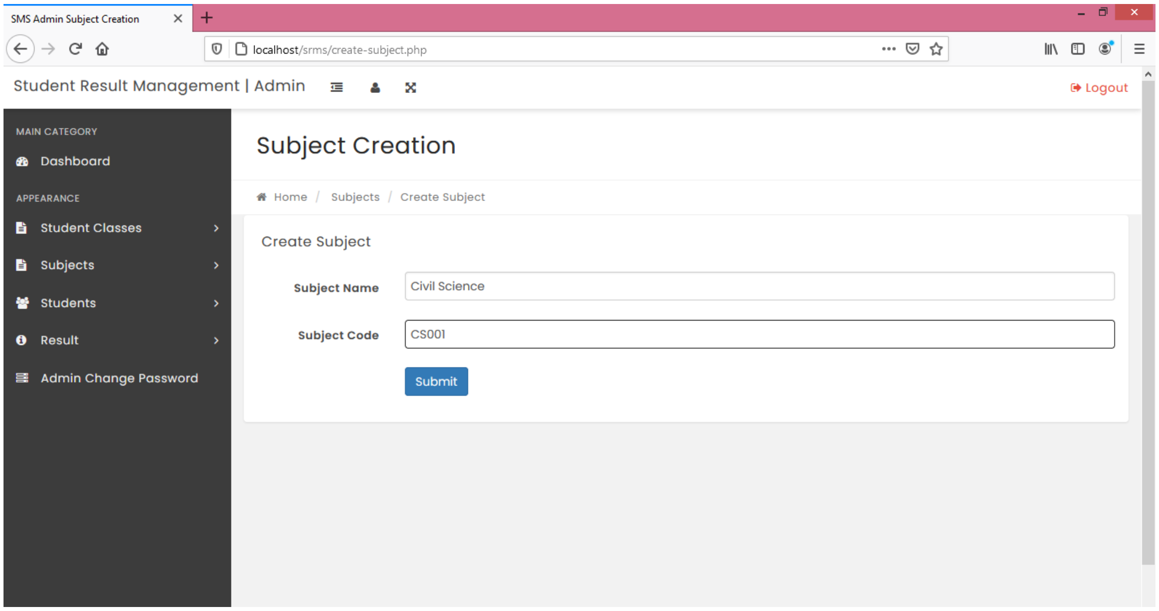
Show 10 entries

Search:

#	Class Name	Class Name Numeric	Section	Creation Date	Action
1	First	1	A	2021-04-09 20:39:40	✎
2	Second	2	A	2021-04-09 20:39:59	✎
3	Third	3	B	2021-04-09 20:40:17	✎
4	Fourth	4	B	2021-04-11 11:20:28	✎
5	Six	6	A	2021-05-30 20:09:06	✎
#	Class Name	Class Name Numeric	Section	Creation Date	Action

Showing 1 to 5 of 5 entries

5.Create Subject


SMS Admin Subject Creation
localhost/srms/create-subject.php
Student Result Management | Admin
Logout
MAIN CATEGORY
Dashboard
APPEARANCE
Student Classes
Subjects
Students
Result
Admin Change Password
Home / Subjects / Create Subject

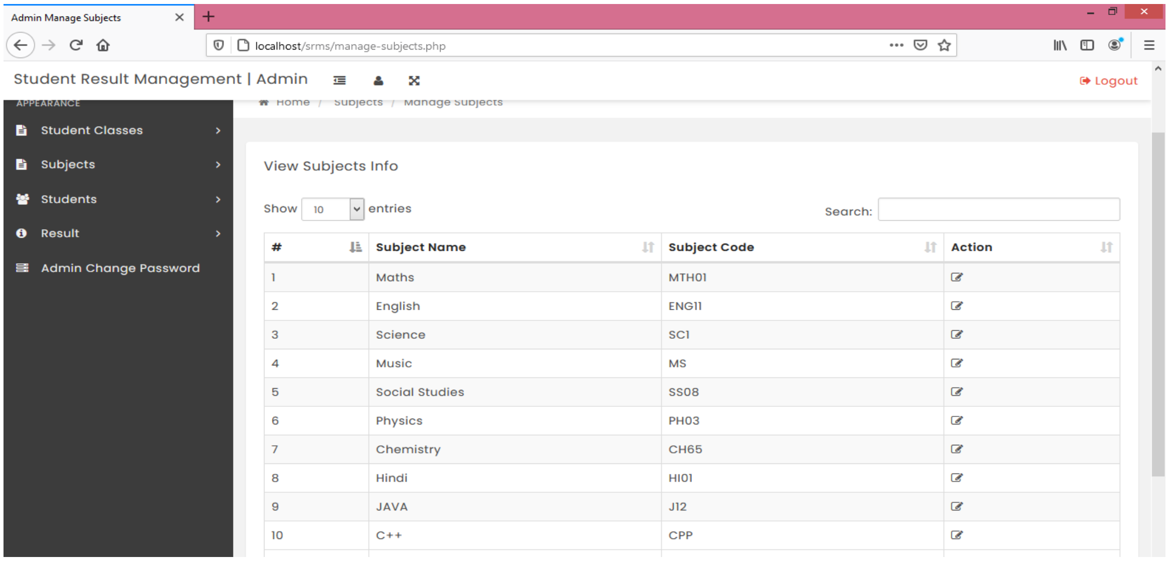
Subject Creation

Create Subject

Subject Name: Civil Science

Subject Code: CS001

6.Manage Subject



Admin Manage Subjects

localhost/srms/manage-subjects.php

Student Result Management | Admin

Logout

APPEARANCE

- Student Classes
- Subjects
- Students
- Result
- Admin Change Password

Home / Subjects / Manage Subjects

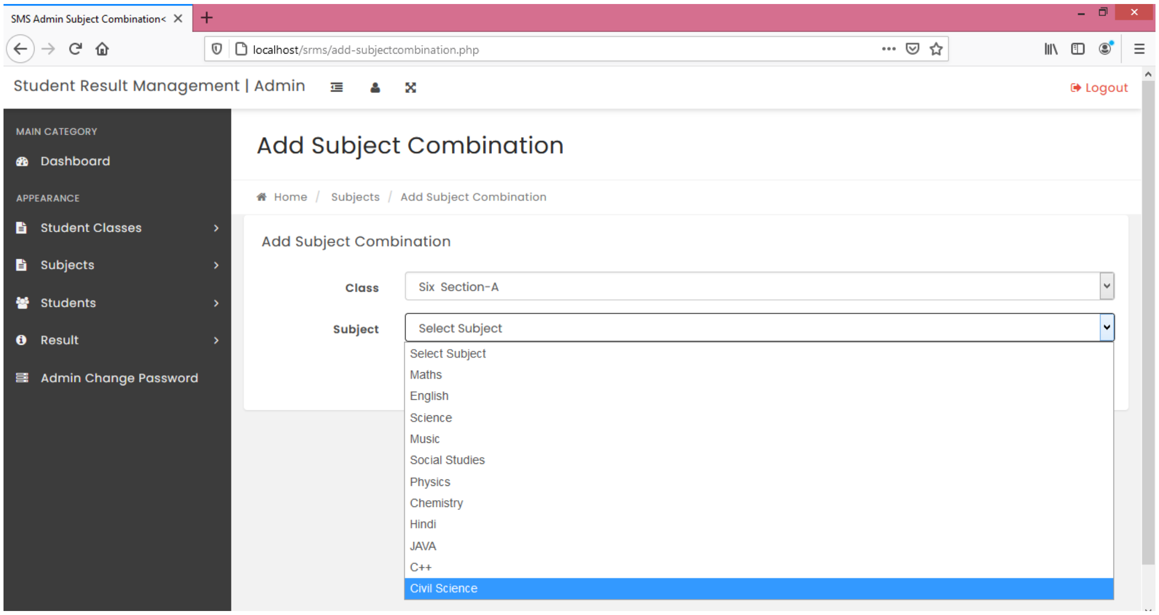
View Subjects Info

Show 10 entries

Search:

#	Subject Name	Subject Code	Action
1	Maths	MTH01	✎
2	English	ENG11	✎
3	Science	SC1	✎
4	Music	MS	✎
5	Social Studies	SS08	✎
6	Physics	PH03	✎
7	Chemistry	CH65	✎
8	Hindi	HI01	✎
9	JAVA	J12	✎
10	C++	CPP	✎

7.Create Class and Subject Combination



SMS Admin Subject Combination<

localhost/srms/add-subjectcombination.php

Student Result Management | Admin

Logout

MAIN CATEGORY

- Dashboard

APPEARANCE

- Student Classes
- Subjects
- Students
- Result
- Admin Change Password

Add Subject Combination

Home / Subjects / Add Subject Combination

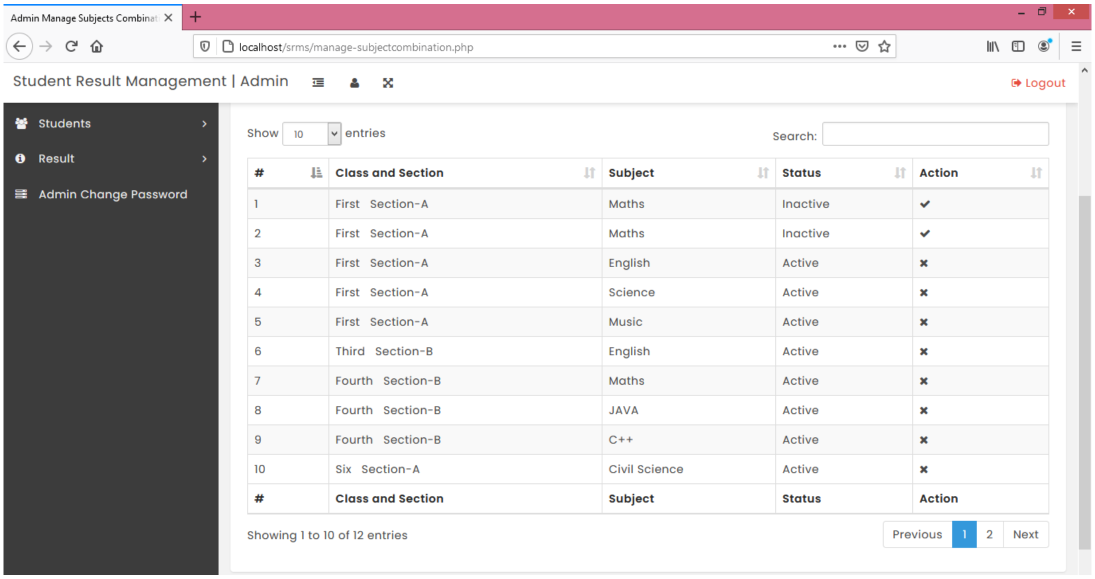
Add Subject Combination

Class: Six Section-A

Subject: Select Subject

- Select Subject
- Maths
- English
- Science
- Music
- Social Studies
- Physics
- Chemistry
- Hindi
- JAVA
- C++
- Civil Science

8.Manage Class and Subject Combination


Admin Manage Subjects Combinat
localhost/srms/manage-subjectcombination.php
Student Result Management | Admin
Logout
Students
Result
Admin Change Password

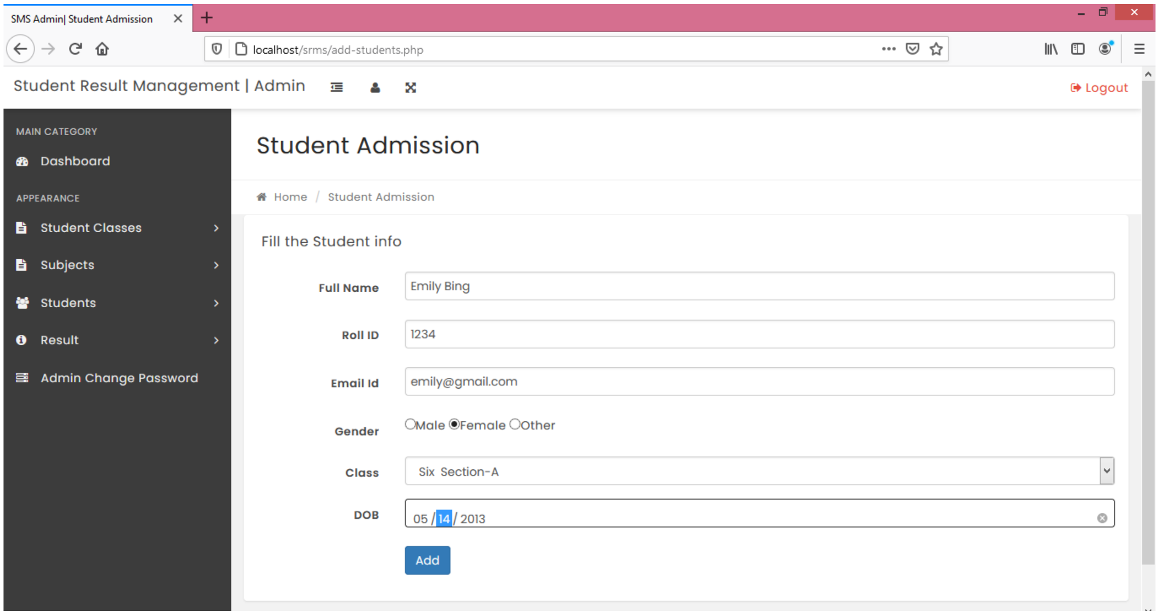
Show 10 entries

Search:

#	Class and Section	Subject	Status	Action
1	First Section-A	Maths	Inactive	✓
2	First Section-A	Maths	Inactive	✓
3	First Section-A	English	Active	✖
4	First Section-A	Science	Active	✖
5	First Section-A	Music	Active	✖
6	Third Section-B	English	Active	✖
7	Fourth Section-B	Maths	Active	✖
8	Fourth Section-B	JAVA	Active	✖
9	Fourth Section-B	C++	Active	✖
10	Six Section-A	Civil Science	Active	✖
#	Class and Section	Subject	Status	Action

Showing 1 to 10 of 12 entries

9.Add Student



Student Admission

Fill the Student info

Full Name: Emily Bing

Roll ID: 1234

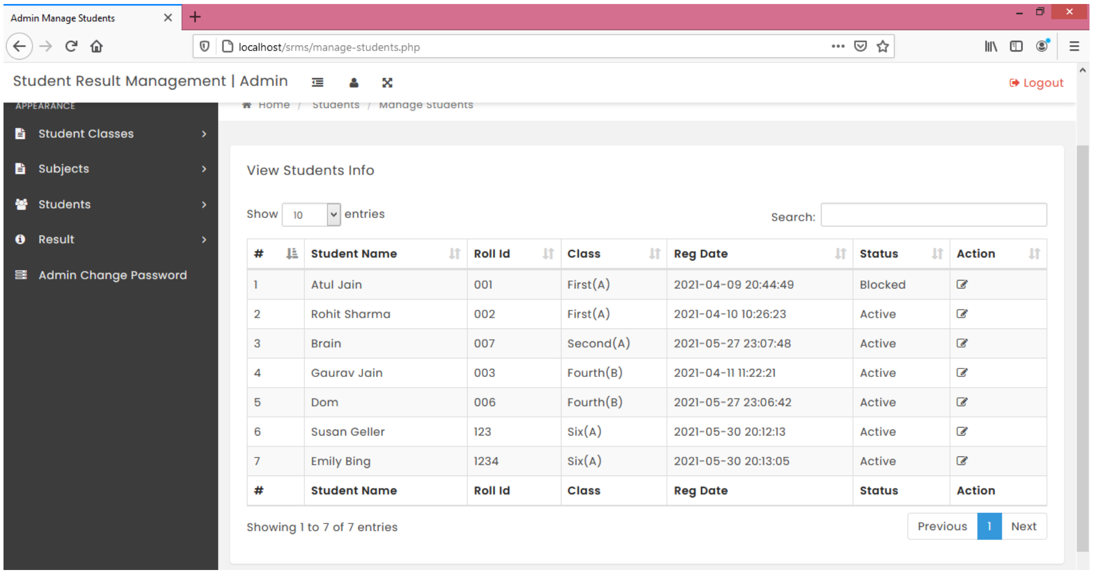
Email Id: emily@gmail.com

Gender: ○Male ●Female ○Other

Class: Six Section-A

DOB: 05 / 14 / 2013

10.Manage Student



Admin Manage Students
localhost/srms/manage-students.php
Student Result Management | Admin
Logout
APPEARANCE
Student Classes
Subjects
Students
Result
Admin Change Password
Home / Students / Manage Students

View Students Info

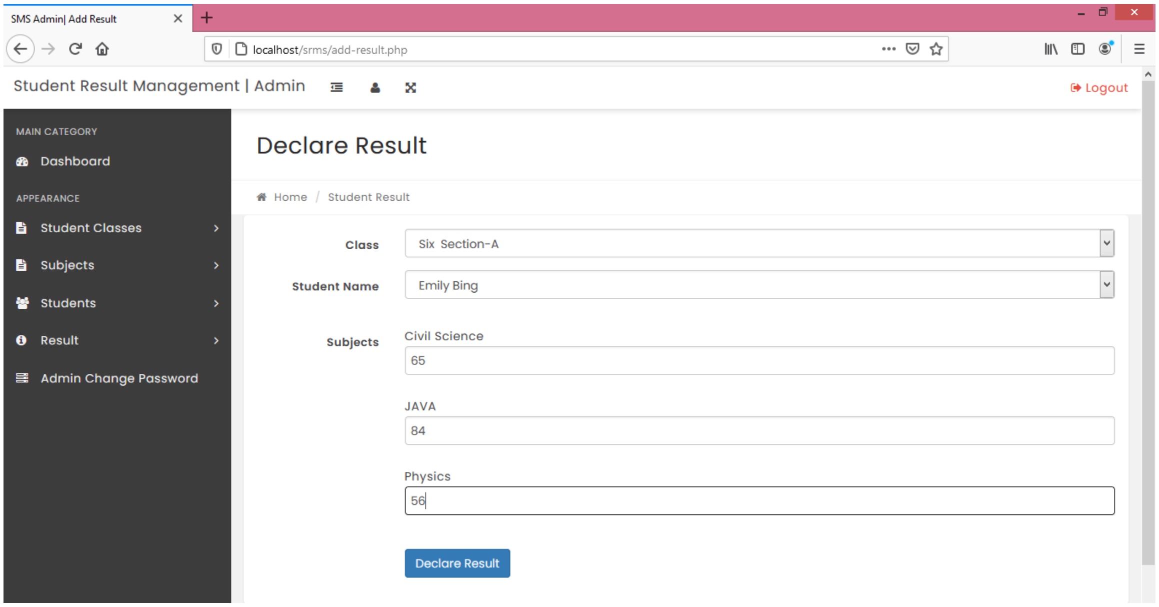
Show 10 entries

Search:

#	Student Name	Roll Id	Class	Reg Date	Status	Action
1	Atul Jain	001	First(A)	2021-04-09 20:44:49	Blocked	✎
2	Rohit Sharma	002	First(A)	2021-04-10 10:26:23	Active	✎
3	Brain	007	Second(A)	2021-05-27 23:07:48	Active	✎
4	Gaurav Jain	003	Fourth(B)	2021-04-11 11:22:21	Active	✎
5	Dom	006	Fourth(B)	2021-05-27 23:06:42	Active	✎
6	Susan Geller	123	Six(A)	2021-05-30 20:12:13	Active	✎
7	Emily Bing	1234	Six(A)	2021-05-30 20:13:05	Active	✎
#	Student Name	Roll Id	Class	Reg Date	Status	Action

Showing 1 to 7 of 7 entries

11.Declare Result



Declare Result

Class: Six Section-A

Student Name: Emily Bing

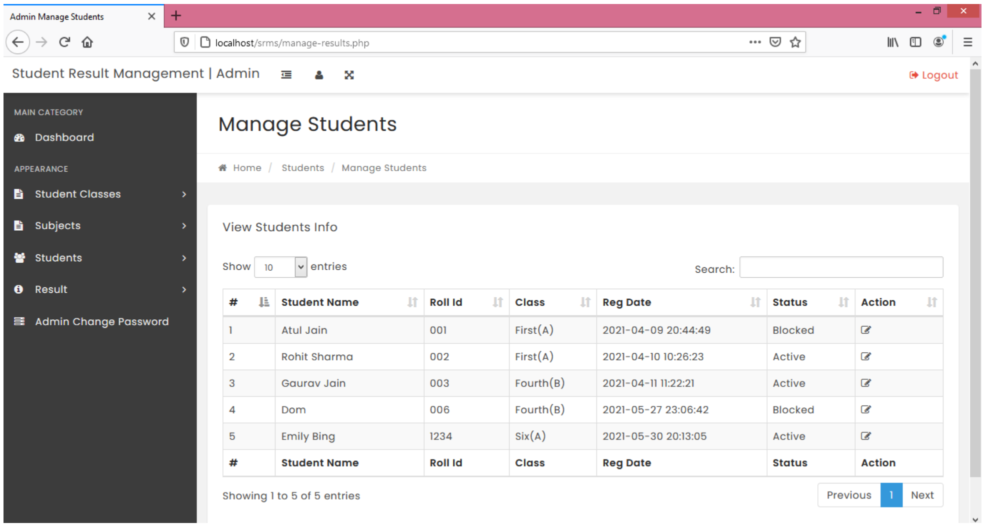
Subjects

Civil Science: 65

JAVA: 84

Physics: 56

12. Manage Result



Manage Students

Home / Students / Manage Students

View Students Info

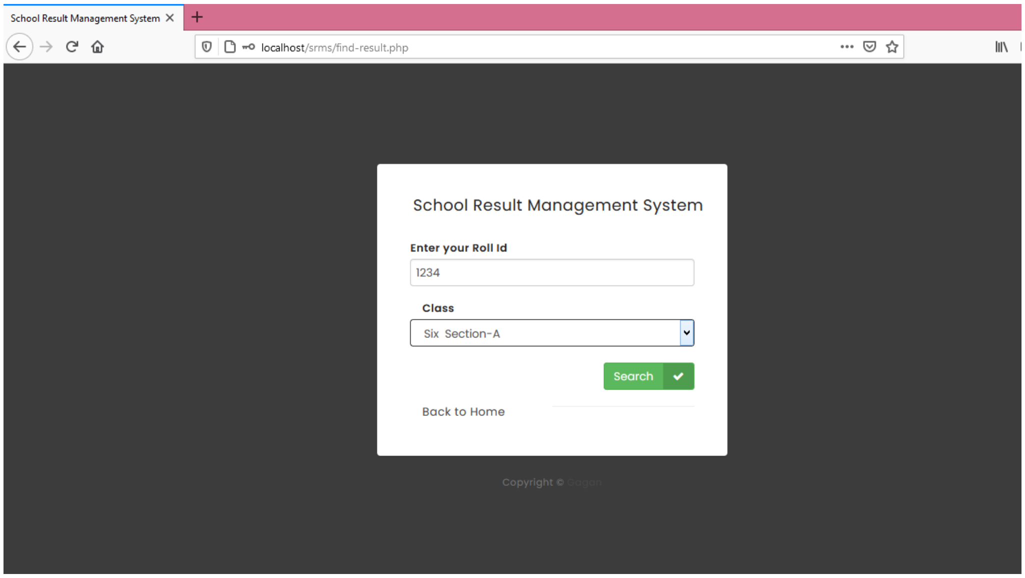
Show 10 entries

Search:

#	Student Name	Roll Id	Class	Reg Date	Status	Action
1	Atul Jain	001	First(A)	2021-04-09 20:44:49	Blocked	✎
2	Rohit Sharma	002	First(A)	2021-04-10 10:26:23	Active	✎
3	Gaurav Jain	003	Fourth(B)	2021-04-11 11:22:21	Active	✎
4	Dom	006	Fourth(B)	2021-05-27 23:06:42	Blocked	✎
5	Emily Bing	1234	Six(A)	2021-05-30 20:13:05	Active	✎
#	Student Name	Roll Id	Class	Reg Date	Status	Action

Showing 1 to 5 of 5 entries

13.Student Result Search


School Result Management System
localhost/srms/find-result.php
Search
Back to Home

School Result Management System

Enter your Roll Id

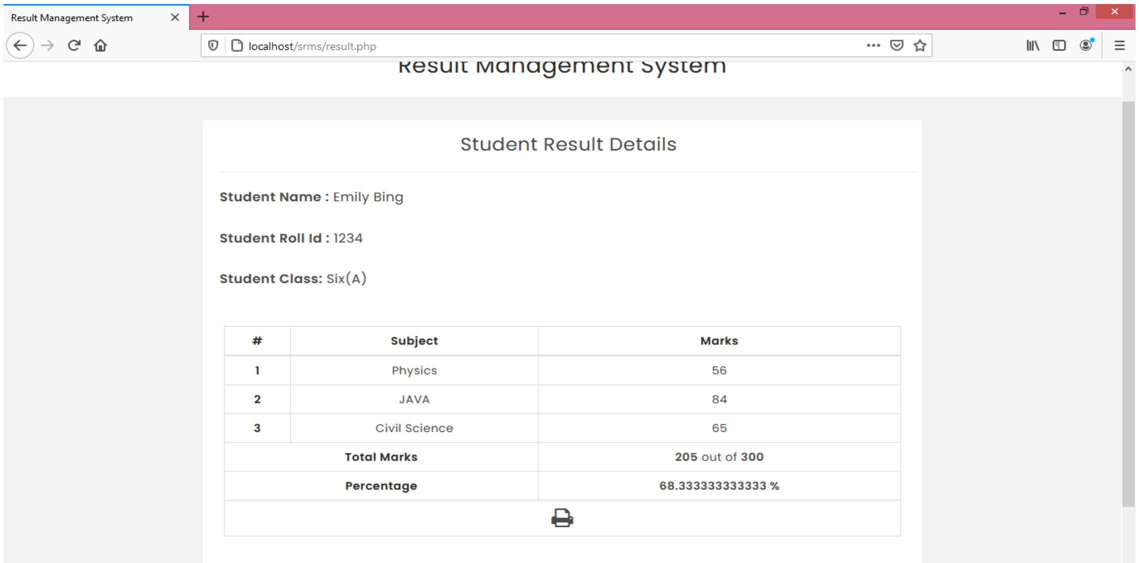
1234

Class

Six Section-A

Copyright © Gagan

14.Result



Result Management System

Student Result Details

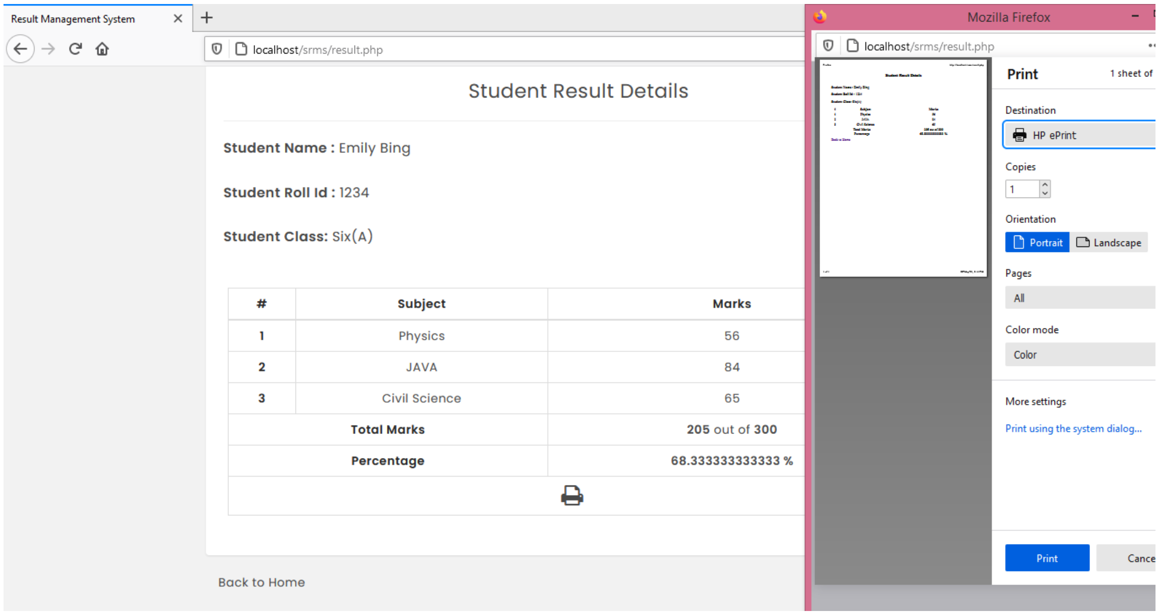
Student Name : Emily Bing

Student Roll Id : 1234

Student Class: Six(A)

#	Subject	Marks
1	Physics	56
2	JAVA	84
3	Civil Science	65
	Total Marks	**205** out of **300**
	Percentage	**68.333333333333 %**

15.Result with Printing



Student Result Details

Student Name : Emily Bing

Student Roll Id : 1234

Student Class: Six(A)

#	Subject	Marks
1	Physics	56
2	JAVA	84
3	Civil Science	65
	Total Marks	**205** out of **300**
	Percentage	**68.333333333333 %**

Back to Home

ANNEXURE 3

Sample Program Code

1) create-class.php

```
<?php

session_start();

error_reporting(0);

include('includes/config.php');

if(strlen($_SESSION['alogin'])=="")

    {

    header("Location: index.php");

    }

    else{

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

{

$classname=$_POST['classname'];

$classnamenumeric=$_POST['classnamenumeric'];

$section=$_POST['section'];

$sql="INSERT INTO
tblclasses(ClassName,ClassNameNumeric,Section)
VALUES(:classname,:classnamenumeric,:section)";
$query = $dbh->prepare($sql);
$query->bindParam(':classname',$classname,PDO::PARAM_STR);
$query->bindParam(':classnamenumeric',$classnamenumeric,PDO::PARAM_STR);
$query->bindParam(':section',$section,PDO::PARAM_STR);
$query->execute();
$lastInsertId = $dbh->lastInsertId();
if($lastInsertId)
{
$msg="Class Created successfully";
}
else
{
$error="Something went wrong. Please try again";
}

}
?>
<!DOCTYPE html>
<html lang="en">
```

```
    <head>

        <meta charset="utf-8">

        <meta http-equiv="X-UA-Compatible" content="IE=edge">

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

        <title>SMS Admin Create Class</title>

        <link rel="stylesheet" href="css/bootstrap.css" media="screen"
>

        <link rel="stylesheet" href="css/font-awesome.min.css"
media="screen" >

        <link rel="stylesheet" href="css/animate-css/animate.min.css"
media="screen" >

        <link rel="stylesheet" href="css/lobipanel/lobipanel.min.css"
media="screen" >

        <link rel="stylesheet" href="css/prism/prism.css"
media="screen" > <!-- USED FOR DEMO HELP - YOU CAN
REMOVE IT -->

        <link rel="stylesheet" href="css/main.css" media="screen" >

        <script src="js/modernizr/modernizr.min.js"></script>

         <style>

        .errorWrap {

    padding: 10px;
    margin: 0 0 20px 0;
    background: #fff;
    border-left: 4px solid #dd3d36;
    -webkit-box-shadow: 0 1px 1px 0 rgba(0,0,0,.1);
    box-shadow: 0 1px 1px 0 rgba(0,0,0,.1);
}
.succWrap{
    padding: 10px;
    margin: 0 0 20px 0;
    background: #fff;
    border-left: 4px solid #5cb85c;
    -webkit-box-shadow: 0 1px 1px 0 rgba(0,0,0,.1);
    box-shadow: 0 1px 1px 0 rgba(0,0,0,.1);
}
        </style>
    </head>
    <body class="top-navbar-fixed">
        <div class="main-wrapper">

            <!-- ========== TOP NAVBAR ========== -->
        <?php include('includes/topbar.php');?>

       <!-----End Top bar>

        <!-- ========== WRAPPER FOR BOTH SIDEBARS &
MAIN CONTENT ========== -->

        <div class="content-wrapper">

            <div class="content-container">


<!-- ========== LEFT SIDEBAR ========== -->

<?php include('includes/leftbar.php');?>

 <!-- /.left-sidebar -->


                <div class="main-page">

                    <div class="container-fluid">

                        <div class="row page-title-div">

                            <div class="col-md-6">

                                <h2 class="title">Create Student Class</h2>

                            </div>


                        </div>

                        <!-- /.row -->
```

```
              <div class="row breadcrumb-div">
                <div class="col-md-6">
                  <ul class="breadcrumb">
                                            <li><a
href="dashboard.php"><i class="fa fa-home"></i> Home</a></li>
                                            <li><a
href="#">Classes</a></li>
                                            <li
class="active">Create Class</li>
                                       </ul>
                </div>

              </div>
              <!-- /.row -->
            </div>
            <!-- /.container-fluid -->

            <section class="section">
              <div class="container-fluid">
```

```
                                <div class="row">
                                    <div class="col-md-8 col-md-offset-2">
                                        <div class="panel">
                                            <div class="panel-heading">
                                                <div class="panel-title">
                                                    <h5>Create Student Class</h5>
                                                </div>
                                            </div>
           <?php if($msg){?>
<div class="alert alert-success left-icon-alert" role="alert">
 <strong>Well done!</strong><?php echo htmlentities($msg); ?>
 </div><?php }
else if($error){?>
    <div class="alert alert-danger left-icon-alert" role="alert">
                                            <strong>Oh snap!</strong> <?php echo htmlentities($error); ?>
                                        </div>
```

```
                    <?php } ?>


                      <div class="panel-body">


                        <form method="post">
                          <div class="form-group has-success">
                            <label for="success" class="control-label">Class Name</label>
                                 <div class="">
                                       <input type="text" name="classname" class="form-control" required="required" id="success">
                              <span class="help-block">Eg- Third, Fouth,Sixth etc</span>
                                 </div>
                            </div>
                           <div class="form-group has-success">
                            <label for="success" class="control-label">Class Name in Numeric</label>
                            <div class="">
                              <input type="number"
          name="classnamenumeric" required="required" class="form-
          control" id="success">

                              <span class="help-block">Eg-
          1,2,4,5 etc</span>

                            </div>

                          </div>

                           <div class="form-group has-
          success">

                            <label for="success"
          class="control-label">Section</label>

                            <div class="">

                              <input type="text"
          name="section" class="form-control" required="required"
          id="success">

                              <span class="help-block">Eg-
          A,B,C etc</span>

                            </div>

                          </div>

           <div class="form-group has-success">



                            <div class="">
```

```
                             <button type="submit"
name="submit" class="btn btn-success btn-labeled">Submit<span
class="btn-label btn-label-right"><i class="fa fa-
check"></i></span></button>

                          </div>

                        </form>

                      </div>

                    </div>

                  </div>

                  <!-- /.col-md-8 col-md-offset-2 -->

                </div>

                <!-- /.row -->
```

```
                    </div>
                    <!-- /.container-fluid -->
                </section>
                <!-- /.section -->

            </div>
            <!-- /.main-page -->

        </div>
        <!-- /.content-container -->
    </div>
    <!-- /.content-wrapper -->

</div>
<!-- /.main-wrapper -->

<!-- ========== COMMON JS FILES ========== -->
<script src="js/jquery/jquery-2.2.4.min.js"></script>
<script src="js/jquery-ui/jquery-ui.min.js"></script>
<script src="js/bootstrap/bootstrap.min.js"></script>
```

```
        <script src="js/pace/pace.min.js"></script>
        <script src="js/lobipanel/lobipanel.min.js"></script>
        <script src="js/iscroll/iscroll.js"></script>

        <!-- ========== PAGE JS FILES ========== -->
        <script src="js/prism/prism.js"></script>

        <!-- ========== THEME JS ========== -->
        <script src="js/main.js"></script>



        <!-- ========== ADD custom.js FILE BELOW WITH YOUR CHANGES ========== -->
    </body>
</html>
<?php  } ?>
```

2)create-student.php

```
<?php

session_start();

error_reporting(0);

include('includes/config.php');

if(strlen($_SESSION['alogin'])=="")

    {

    header("Location: index.php");

    }

    else{

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

{

$studentname=$_POST['fullanme'];

$roolid=$_POST['rollid'];

$studentemail=$_POST['emailid'];

$gender=$_POST['gender'];

$classid=$_POST['class'];

$dob=$_POST['dob'];

$status=1;

$sql="INSERT INTO
tblstudents(StudentName,RollId,StudentEmail,Gender,ClassId,DOB,
Status)
```

```
VALUES(:studentname,:roolid,:studentemail,:gender,:classid,:dob,:status)";
$query = $dbh->prepare($sql);
$query->bindParam(':studentname',$studentname,PDO::PARAM_STR);
$query->bindParam(':roolid',$roolid,PDO::PARAM_STR);
$query->bindParam(':studentemail',$studentemail,PDO::PARAM_STR);
$query->bindParam(':gender',$gender,PDO::PARAM_STR);
$query->bindParam(':classid',$classid,PDO::PARAM_STR);
$query->bindParam(':dob',$dob,PDO::PARAM_STR);
$query->bindParam(':status',$status,PDO::PARAM_STR);
$query->execute();
$lastInsertId = $dbh->lastInsertId();
if($lastInsertId)
{
$msg="Student info added successfully";
}
else
{
$error="Something went wrong. Please try again";
}

}
?>
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
           <meta name="viewport" content="width=device-width, initial-scale=1">
        <title>SMS Admin| Student Admission </title>
        <link rel="stylesheet" href="css/bootstrap.min.css" media="screen" >
        <link rel="stylesheet" href="css/font-awesome.min.css" media="screen" >
        <link rel="stylesheet" href="css/animate-css/animate.min.css" media="screen" >
        <link rel="stylesheet" href="css/lobipanel/lobipanel.min.css" media="screen" >
        <link rel="stylesheet" href="css/prism/prism.css" media="screen" >
        <link rel="stylesheet" href="css/select2/select2.min.css" >

        <link rel="stylesheet" href="css/main.css" media="screen" >

        <script src="js/modernizr/modernizr.min.js"></script>

    </head>

    <body class="top-navbar-fixed">

        <div class="main-wrapper">


            <!-- ========== TOP NAVBAR ========== -->

  <?php include('includes/topbar.php');?>

            <!-- ========== WRAPPER FOR BOTH SIDEBARS & MAIN CONTENT ========== -->

            <div class="content-wrapper">

                <div class="content-container">


                    <!-- ========== LEFT SIDEBAR ========== -->

                   <?php include('includes/leftbar.php');?>

                    <!-- /.left-sidebar -->


                    <div class="main-page">
```

```
<div class="container-fluid">

    <div class="row page-title-div">

      <div class="col-md-6">

        <h2 class="title">Student Admission</h2>



      </div>



      <!-- /.col-md-6 text-right -->

    </div>

    <!-- /.row -->

    <div class="row breadcrumb-div">

      <div class="col-md-6">

        <ul class="breadcrumb">

          <li><a href="dashboard.php"><i class="fa fa-home"></i> Home</a></li>



          <li class="active">Student Admission</li>

        </ul>

      </div>
```

```
                            </div>
                            <!-- /.row -->
                        </div>
                        <div class="container-fluid">

                        <div class="row">
                                    <div class="col-md-12">
                                        <div class="panel">
                                            <div class="panel-heading">
                                                <div class="panel-title">
                                                    <h5>Fill the Student info</h5>
                                                </div>
                                            </div>
                                            <div class="panel-body">
<?php if($msg){?>
<div class="alert alert-success left-icon-alert" role="alert">
 <strong>Well done!</strong><?php echo htmlentities($msg); ?>
 </div><?php }
else if($error){?>
    <div class="alert alert-danger left-icon-alert" role="alert">
```

```
                                        <strong>Oh snap!</strong> <?php echo htmlentities($error); ?>
                                    </div>
                                    <?php } ?>
                                           <form class="form-horizontal" method="post">


<div class="form-group">
<label for="default" class="col-sm-2 control-label">Full Name</label>
<div class="col-sm-10">
<input type="text" name="fullanme" class="form-control" id="fullanme" required="required" autocomplete="off">
</div>
</div>


<div class="form-group">
<label for="default" class="col-sm-2 control-label">Roll ID</label>
<div class="col-sm-10">
<input type="text" name="rollid" class="form-control" id="rollid" maxlength="5" required="required" autocomplete="off">
</div>
```

```
</div>


<div class="form-group">

<label for="default" class="col-sm-2 control-label">Email
Id</label>

<div class="col-sm-10">

<input type="email" name="emailid" class="form-control"
id="email" required="required" autocomplete="off">

</div>

</div>

<div class="form-group">

<label for="default" class="col-sm-2 control-label">Gender</label>

<div class="col-sm-10">

<input type="radio" name="gender" value="Male"
required="required" checked="">Male <input type="radio"
name="gender" value="Female" required="required">Female <input
type="radio" name="gender" value="Other"
required="required">Other

</div>

</div>
```

```
<div class="form-group">
                                                    <label for="default" class="col-sm-2 control-label">Class</label>
                                                    <div class="col-sm-10">
 <select name="class" class="form-control" id="default" required="required">
<option value="">Select Class</option>
<?php $sql = "SELECT * from tblclasses";
$query = $dbh->prepare($sql);
$query->execute();
$results=$query->fetchAll(PDO::FETCH_OBJ);
if($query->rowCount() > 0)
{
foreach($results as $result)
{   ?>
<option value="<?php echo htmlentities($result->id); ?>"><?php echo htmlentities($result->ClassName); ?>&nbsp; Section-<?php echo htmlentities($result->Section); ?></option>
<?php }} ?>
 </select>
                                                    </div>
```

```
  </div>
<div class="form-group">
                    <label for="date" class="col-sm-2 control-label">DOB</label>
                    <div class="col-sm-10">
                      <input type="date" name="dob" class="form-control" id="date">
                    </div>
                  </div>




                  <div class="form-group">
                    <div class="col-sm-offset-2 col-sm-10">
                      <button type="submit" name="submit" class="btn btn-primary">Add</button>
                    </div>
                  </div>
                </form>
```

```
                                    </div>
                                </div>
                            </div>
                            <!-- /.col-md-12 -->
                        </div>
            </div>
        </div>
        <!-- /.content-container -->
    </div>
    <!-- /.content-wrapper -->
</div>
<!-- /.main-wrapper -->
<script src="js/jquery/jquery-2.2.4.min.js"></script>
<script src="js/bootstrap/bootstrap.min.js"></script>
<script src="js/pace/pace.min.js"></script>
<script src="js/lobipanel/lobipanel.min.js"></script>
<script src="js/iscroll/iscroll.js"></script>
<script src="js/prism/prism.js"></script>
<script src="js/select2/select2.min.js"></script>
<script src="js/main.js"></script>
```

```
        <script>
            $(function($) {
                $(".js-states").select2();
                $(".js-states-limit").select2({
                    maximumSelectionLength: 2
                });
                $(".js-states-hide").select2({
                    minimumResultsForSearch: Infinity
                });
            });
        </script>
    </body>
</html>
<?PHP } ?>
```