**Software Engineering**

Indian Institute of Information Technology, Nagpur

***Attendance System***

Himanshu Gupta(BT17CSE093)

Mohammad Saqib(BT17CSE093)

**Invigilator**

Dr. Pooja Jain

# **Objective:-**

“Attendance System” is webapp developed for maintaining the attendance of the student on the daily basis in the collage. Here the faculties, who are handling the courses, will be responsible to mark the attendance of the students. Each faculty will be given with a separate username and password based on the course they handle. An accurate report based on the student attendance is generated here. This system will also help in evaluating attendance of a student. Students can also see their attendance in registered course.

**Purpose :**-

The main purpose of this project to reduce wastage of paper and pen on taking attendance of students,and storing them. Also reduce time which is wasted on calculating the total attendance of each student in a month or semester. It will be digital and ease to access by student, faculty and administration.

**Intended Audience :-**

The audience of this system will be :

1. Students

2. Faculty members

3. Administration.

This project will be managed by Administration.

**Project Scope:-**

1. Make the attendee process easier and effective.

2. Help faculty in the attendance process every time.

3. Manage and organize the attendance page ease to access.

**Product Perspective :-**

The Teachers or Professors manually take attendance in every class each day.

They have to carry attendance papers or registers from a class to class.

It creates burden on them. Or sometimes they forget to carry, then they have to go back to their office to collect it.

Also till the end of session, they have to maintain it properly and have to calculate attendance percentage etc. It wastes their valuable time and energy.

The Online Attendance System is a new system that replaces the current manual registers which store the students ’s attendance,leaves. And it will do the calculation on attendance.

The main scope of this project is to make attendance process more organized in every class. There are also many benefits for students: they can manage their attendance, absences .It makes it easier to have a clear picture of every student’s attendance throughout the academic year

**Project Plan:-**

This project has six phases to be completed within the time line. They are initiating, project plan, components, process model, testing, and feedback. The expected time for the project will take around six months.

The project plan is in a PDF document.

**FEASIBILITY :-**

1. **Economically feasibility:**

Development of this application is highly economically feasible. The only thing to be done is making an environment with an effective supervision.It is cost effective in the sense that has eliminated the paper work completely. The system is also time effective because the calculations are automated which are made at the end of the month or as per the user requirement.

1. **Technical feasibility:**

The technical requirement for the system is economic and it does not use much more Hardware and software space.

Install all upgrades framework into the .Net package supported windows based application. this application depends on Microsoft office and intranet service ,database. Enter their attendance and generate report to excel sheet.

1. **Operational feasibility:**

The system working is quite easy to use and learn due to its simple but attractive interface. User requires no special training for operating the system. Technical performance include issues such as determining whether the system can provide the right information for the Department personnel student details, and whether the system can be organized so that it always delivers this information at the right place and on time using intranet services. Acceptance revolves around the current system and its personnel.

***Performance Requirements:***

The system has been designed such that it will give the most optimized performance with the following:

· Works on a system with CHROME, MOZILLA, IE and all the other popular BROWSERS.

· Since it is a portable, it can be opened in any operating system and it will provide the same performance.

**Operating Environment :-**

The second step is to connect this system to the Canvas site. That is to connect the Canvas database to the system database to work as one system on the Canvas site. This step would complete the work, and the project will work in one system. That is because the attendance report will be updated all the time. Also, the Canvas site will control all the students’ attendance reports not in a separate system or database.

This system has some requirements to be accomplished.

Hardware requirements (Minimum Requirement):

1. Cables for the device
2. Ram - 1GB

We also need some system requirements:

1) Create new databases and indexes for students and class list by using mysql

2) Make connection to the current database

3) Design interfaces for the users

4) Design an attendance page on site

**PROJECT DESCRIPTION:**

1. Problem Definition:

This system developed will reduce the manual work and avoid redundant data. By maintaining the attendance manually, then efficient reports cannot be generated. The system can generate efficient report based on the attendance. As the attendances are maintained in registers it has been a tough task for admin and faculty to maintain for long time. Instead the software can keep long and retrieve the information when needed.

1. Project Overview:

Attendance Management System basically has three main modules for proper functioning.

* 1. Admin module has rights for creating any new entry of faculty and student details.
  2. Faculty has a rights of making daily attendance, generating report. Attendance report can be taken by given details of student details, date, class
  3. Students have right to see their attendance and drop a message if there any error.

3. Module Description:

The system should be designed in such a way that only authorized people should be allowed to access some particular modules. The records should be modified by administrators and faculty. The faculty and student should always be in control of the application and not the viceversa.

Admin Module:

Student Details:

In this module deals with the allocation of roll no and personal details for new batch.It will generate of personal details of student and academic details of the students with the photos.

Faculty Details:

It helps to allot the course and the class date to the particular faculty. It provides the facility to have a user name and password to the staffs .

Attendance details:

It will be makes to the attendance database all students. Entered attendance to stored in the database subject ,period wise into the particular date.It will help to the get report of the attendance.

Faculty Module:

Attendance details:

It assists the faculty to mark attendance to the students for their course. This will authenticate the faculty before making the entry.

Student Module:

It assists the student to view their attendance in respective course.

They can also rise a query if there is any mistake.

System Features:-

Functional Requirements :-

1. Login to session: Login to the system using email and password.

* Requirements: To login to system, all of them should have registered in the administrative office before login.

1. Attendance edit/report: Faculty check the report and the attendance control panel page Function.

* Requirements: Faculty has to check on the report and give the final submission. Faculty has full control of modifying the attendance and looking at students who have attendance shortage. Faculty can drop messages from the system to students who missed more number of classes.

3) Drop Message: All user have right to drop a message related to respective field.

Non-functional Requirements:-

1. The Current System Security: The current system, which is Canvas, has its policy on its site page. The current system builds upon a email and password access. Students and faculty can access to his/her account through their page, and they can control it. The system now has its own policy and security; however, the new feature we will add to the system will need some security requirements to the system. The new feature in the system will add some values to the current policy to maintain the security in the right way. It also provides proof of compliance. The new policy in the system will deal with the security in many cases. The security will have more components on the system in a high control panel. The plan is to secure the outsider and insider community of misused the system (e.g. identification theft). Strong security is part of the policy’s purpose.
2. User Access : Inside the community, there are students, faculty, and administration staff who are going to use the system.The main actor of the users in the entity is faculty. The faculty member will use the entity to control the attendance page. Faculty’s job is to add, edit, update and delete any record.
3. Threats to the system security.This system may face many threats. Sometimes, it comes from a community insider. This could be someone who discloses the data form the database where it located, in the registration office.These records will be the official record for all students, since they begin school and until they graduate.

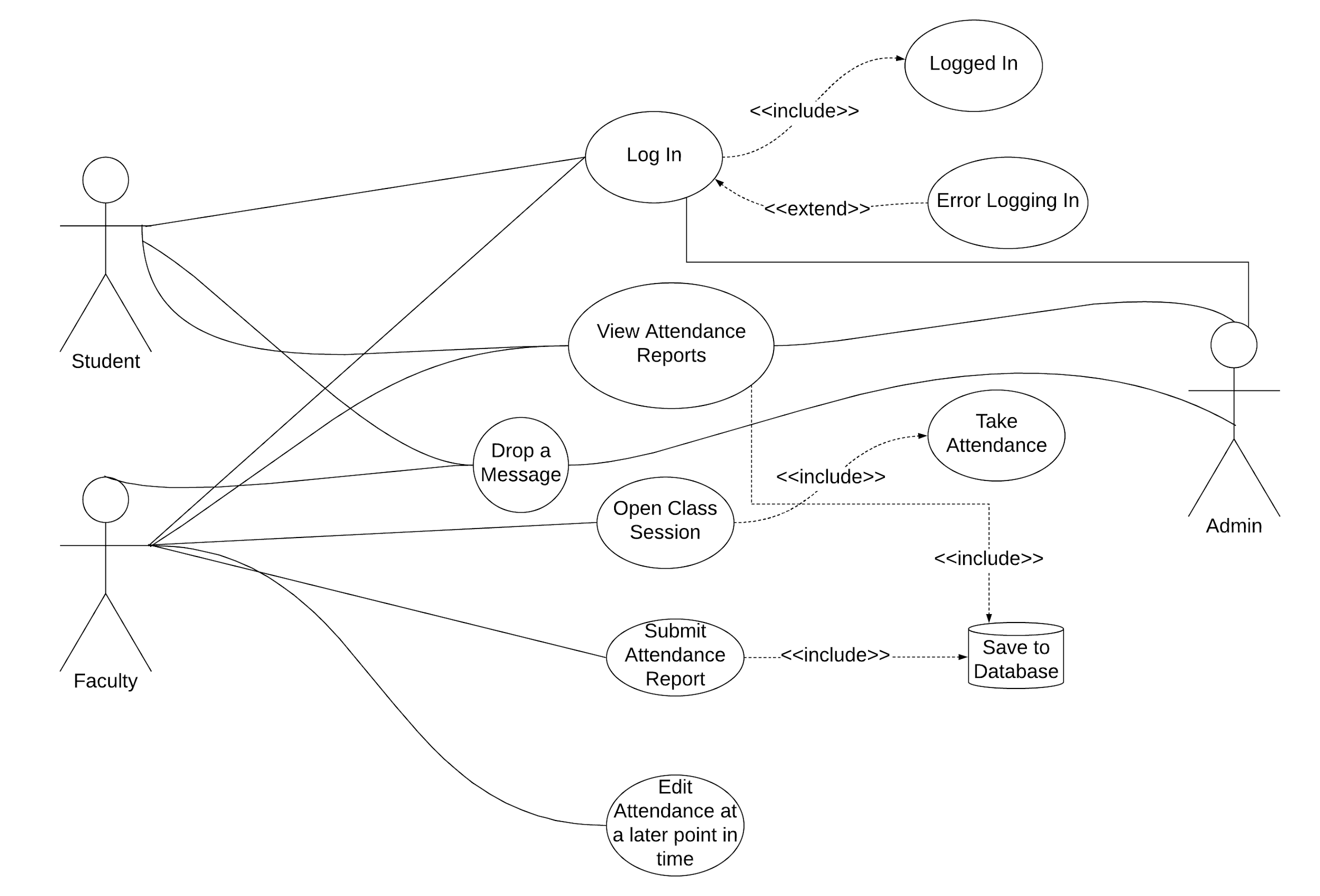
**Design Constraints:**

· Though the website has been designed with optimized performance in

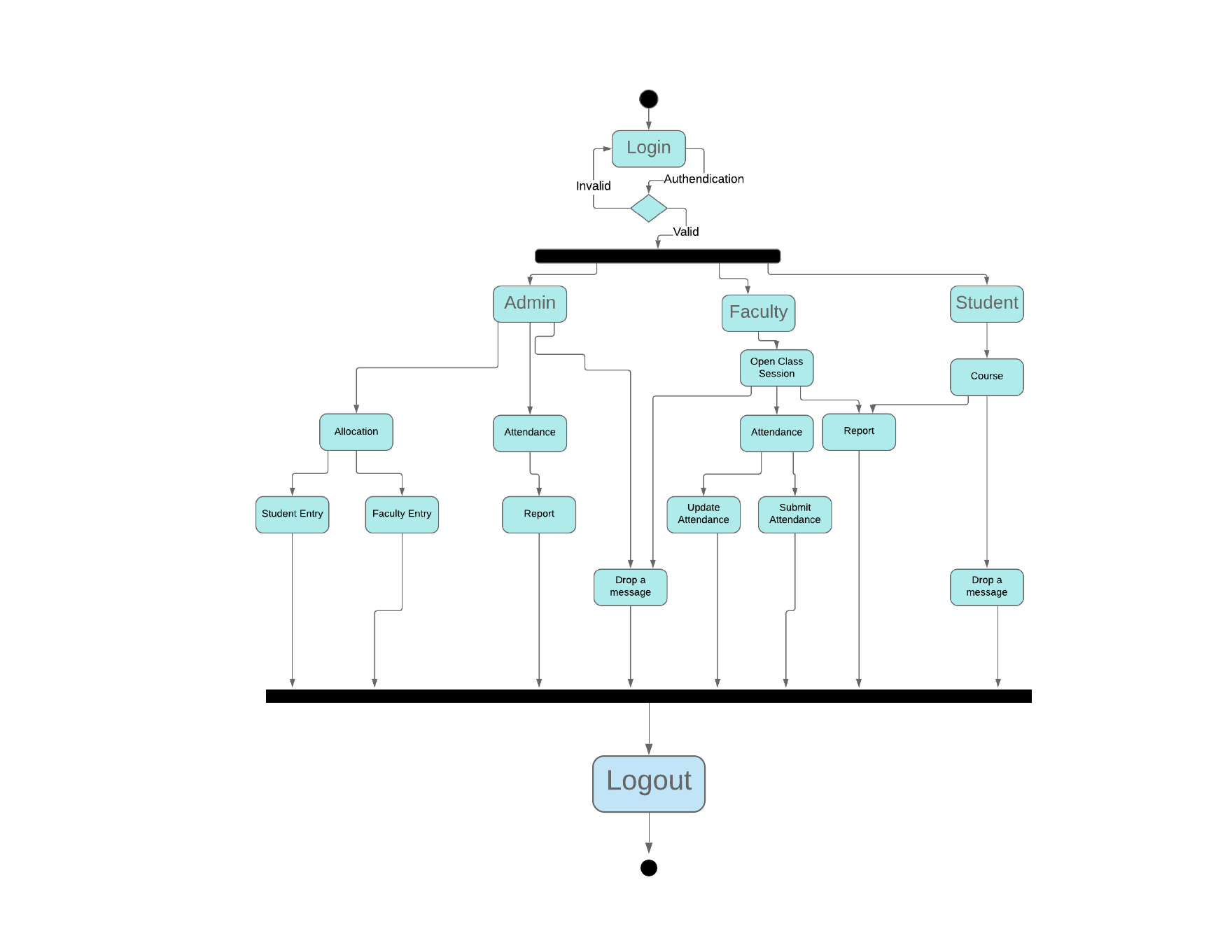
mind, the major constraint is that it is not mobile friendly.

· The site has been created with all basic web dev technologies, and even JAVA which is a really robust language, but due to the bandwidth of the server allotted, the website cannot handle more than 150 users at a time.

***USE Case Diagram:***

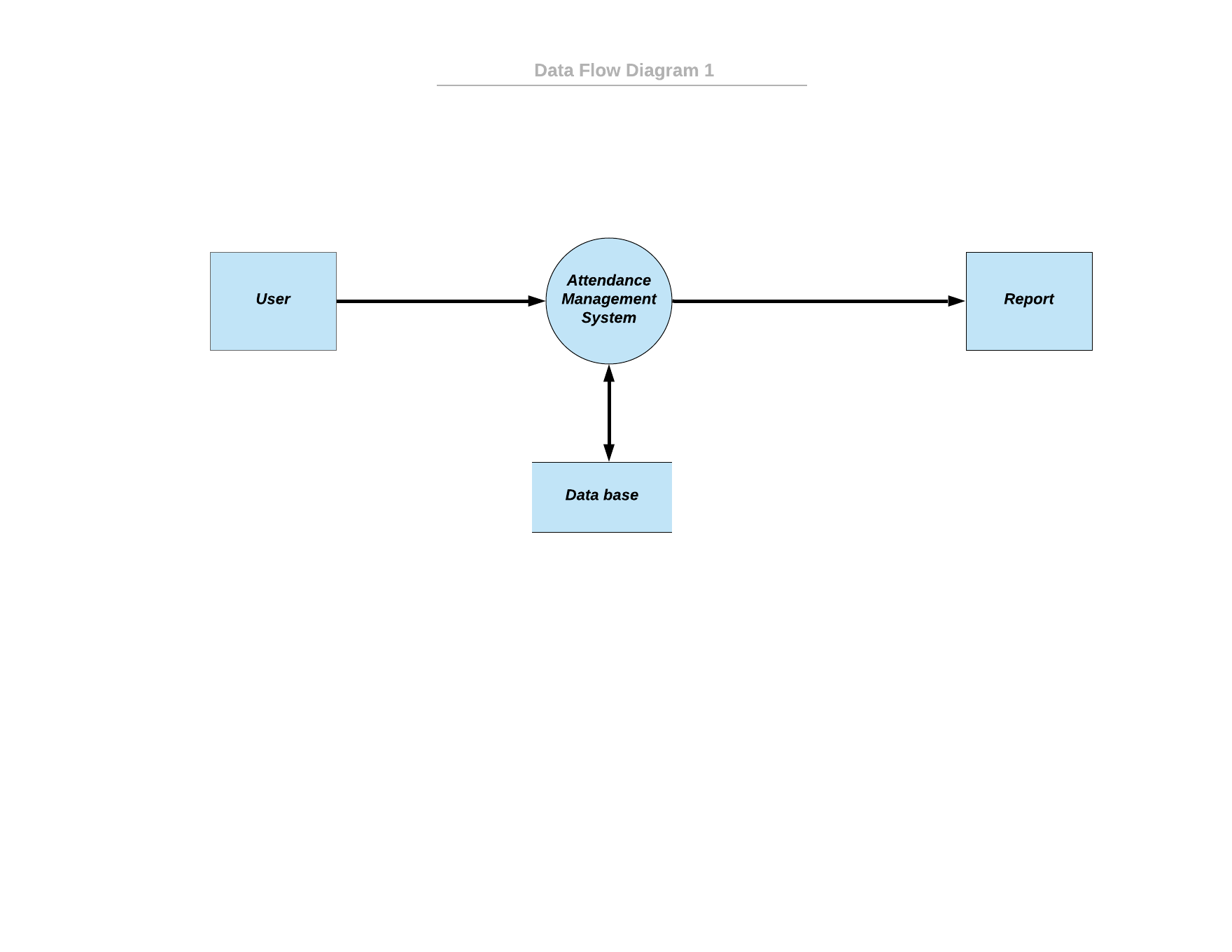


***Activity Diagram:***

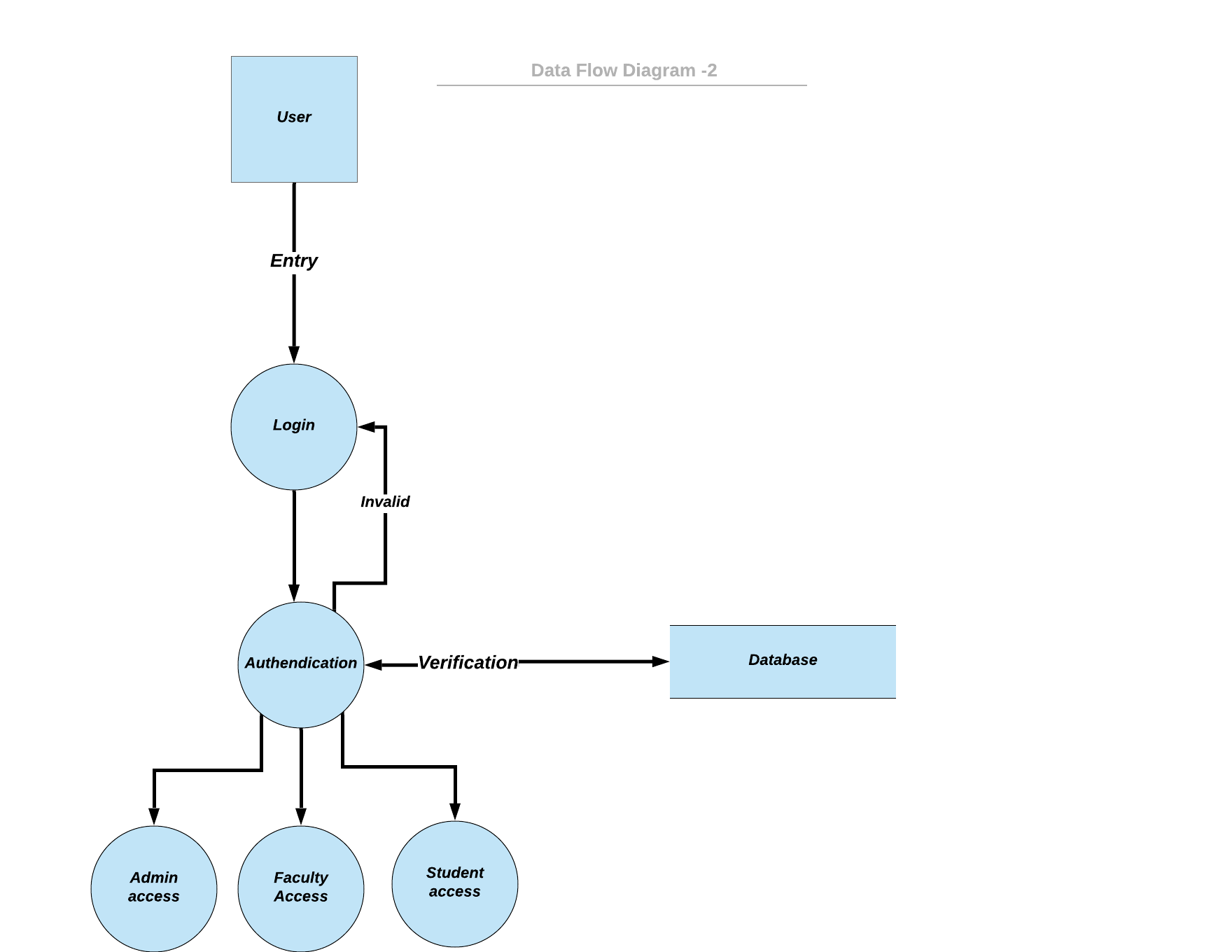
******

***Data Flow Diagram:-***

1.DFD level - 0:

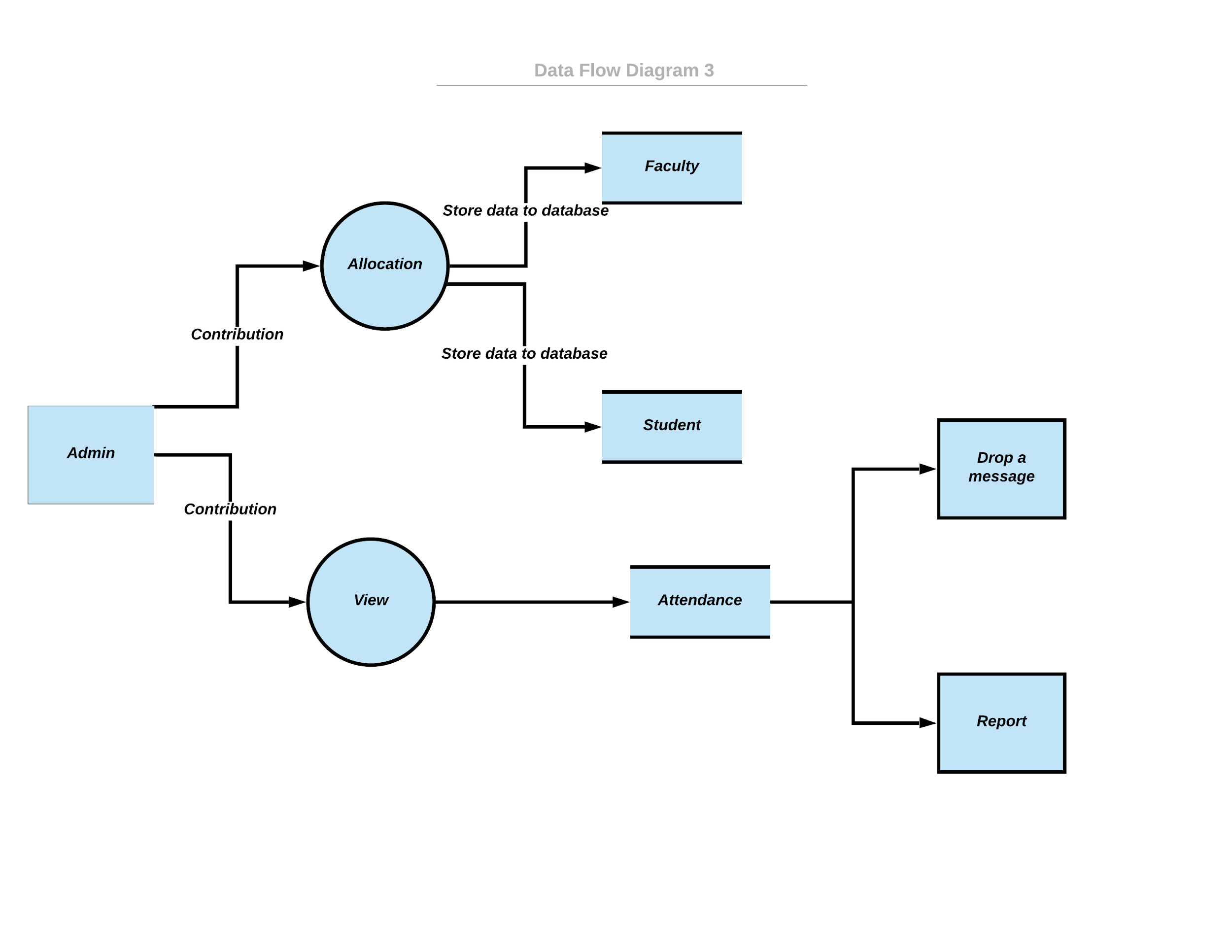


2. DFD Level-1:

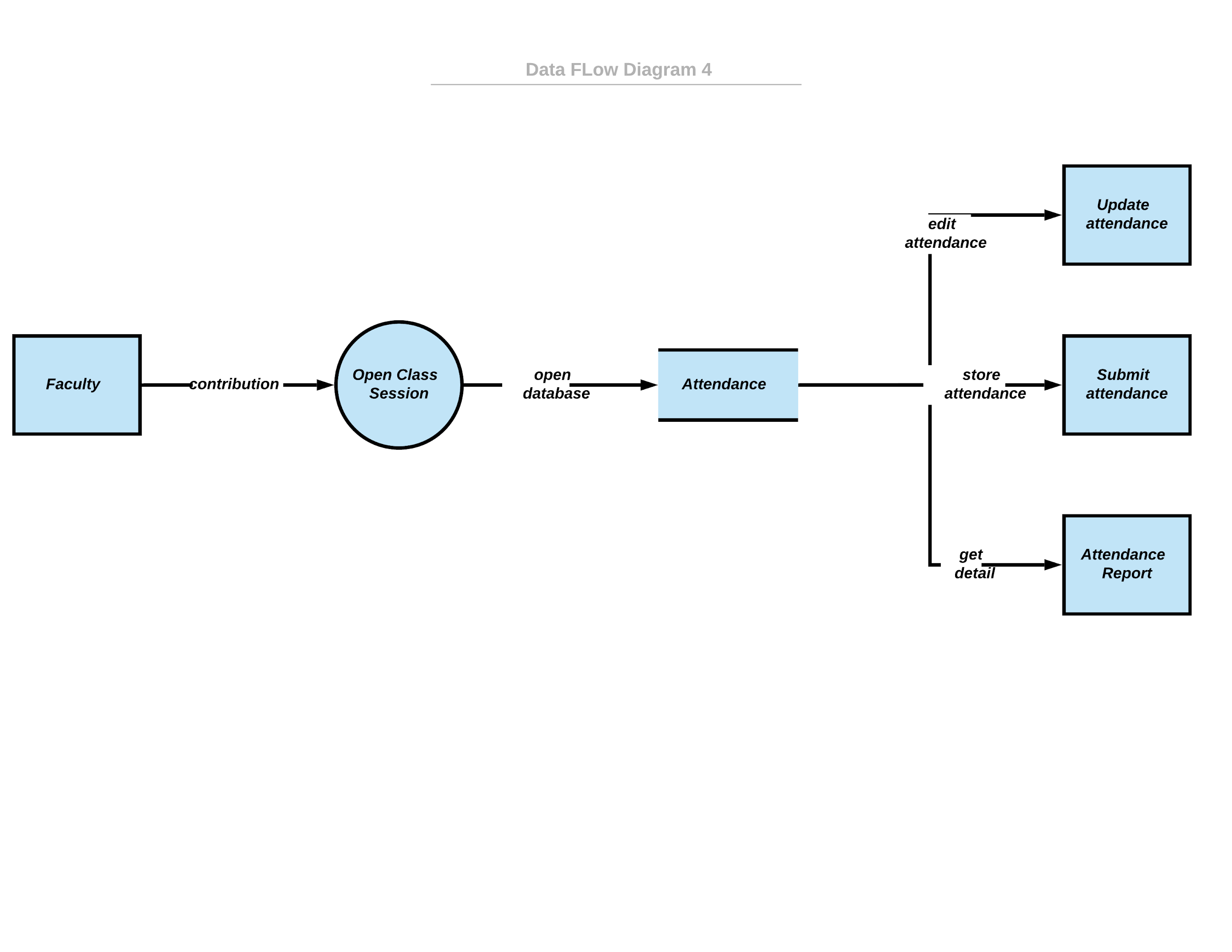


3. DFD Level-2:

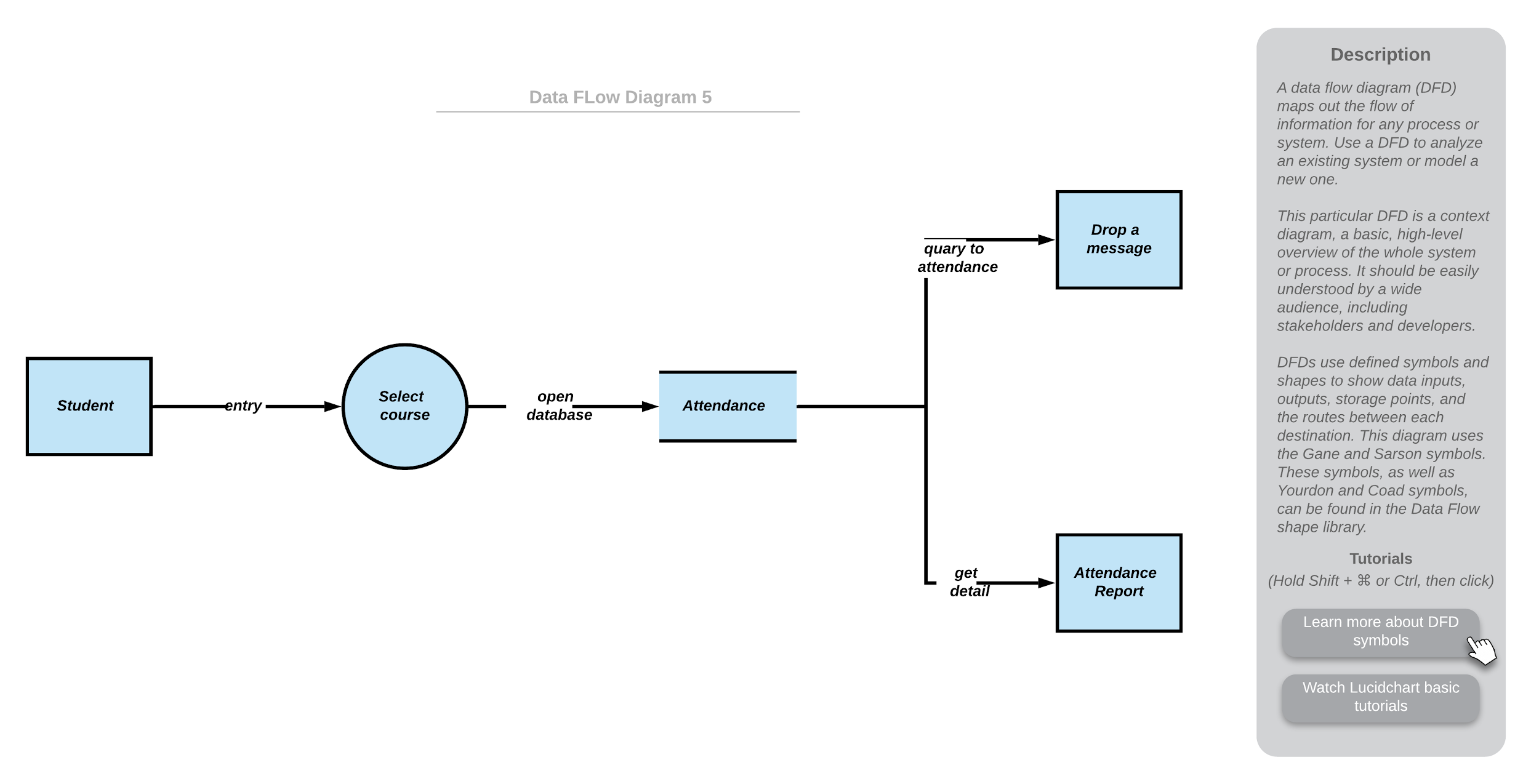
3.1 Admin:-



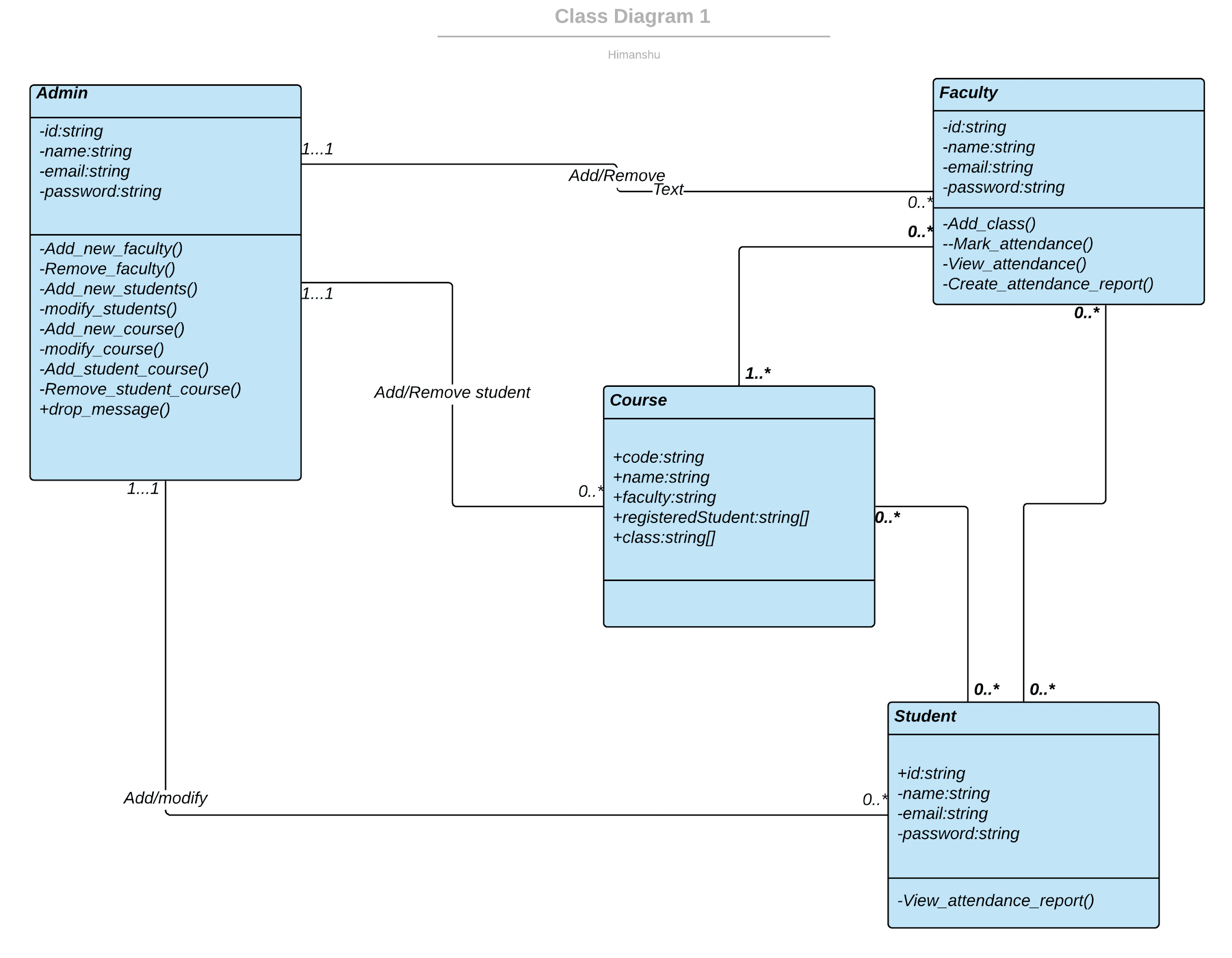
3.2 Faculty:



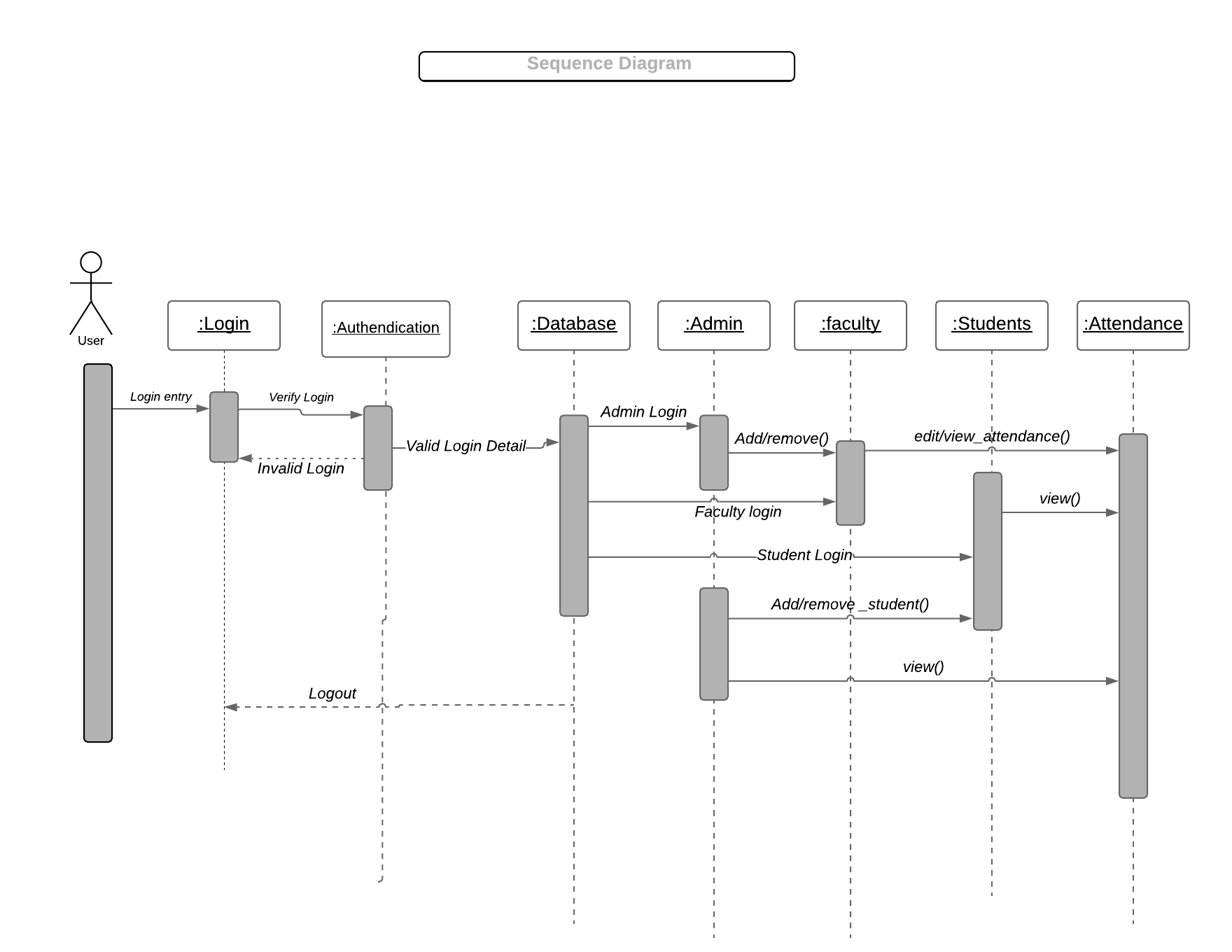
3.3 Student:-



***Class Diagram:-***



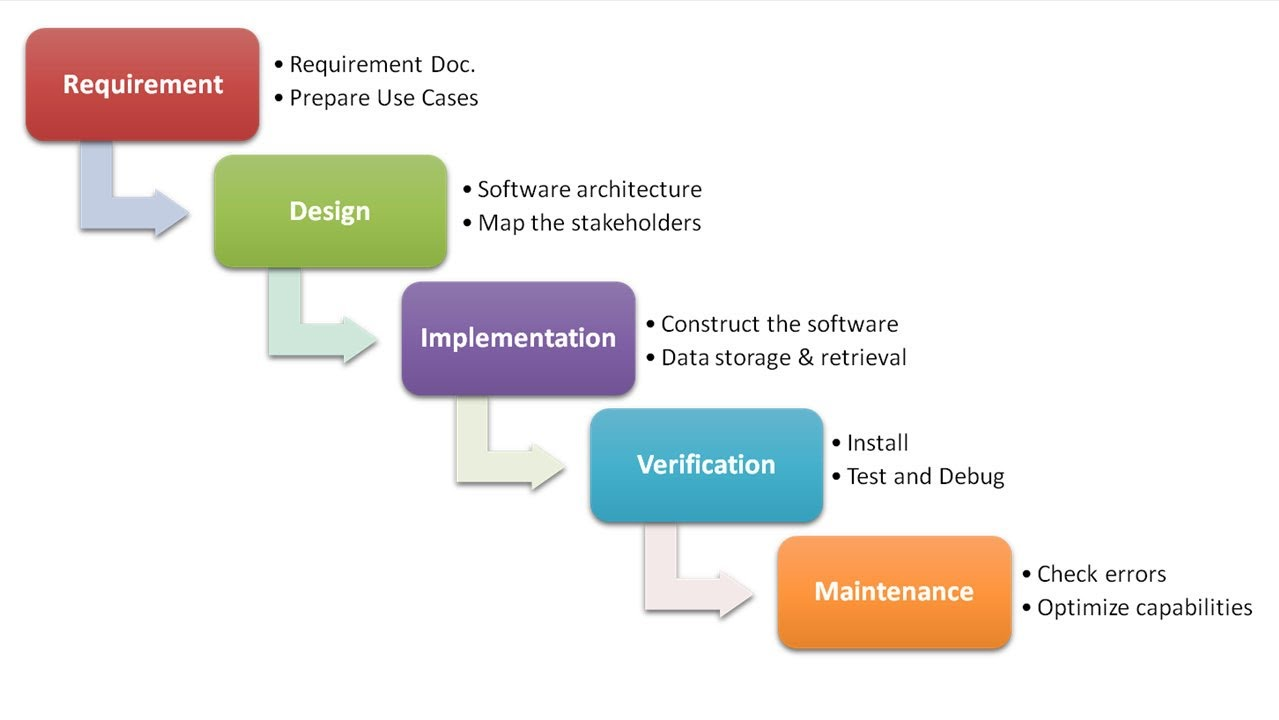
***Sequence Diagram:-***



***PROCESS MODEL – THE WATERFALL MODEL***

In this online auction system, the main process model we employ is the waterfall model. The waterfall model basically can be defined as- The waterfall model, sometimes called the classic life cycle, suggests a systematic, sequential approach to software development that begins with customer specification of requirements and progresses through planning, modeling, construction, and deployment, culminating in ongoing support of the completed software.

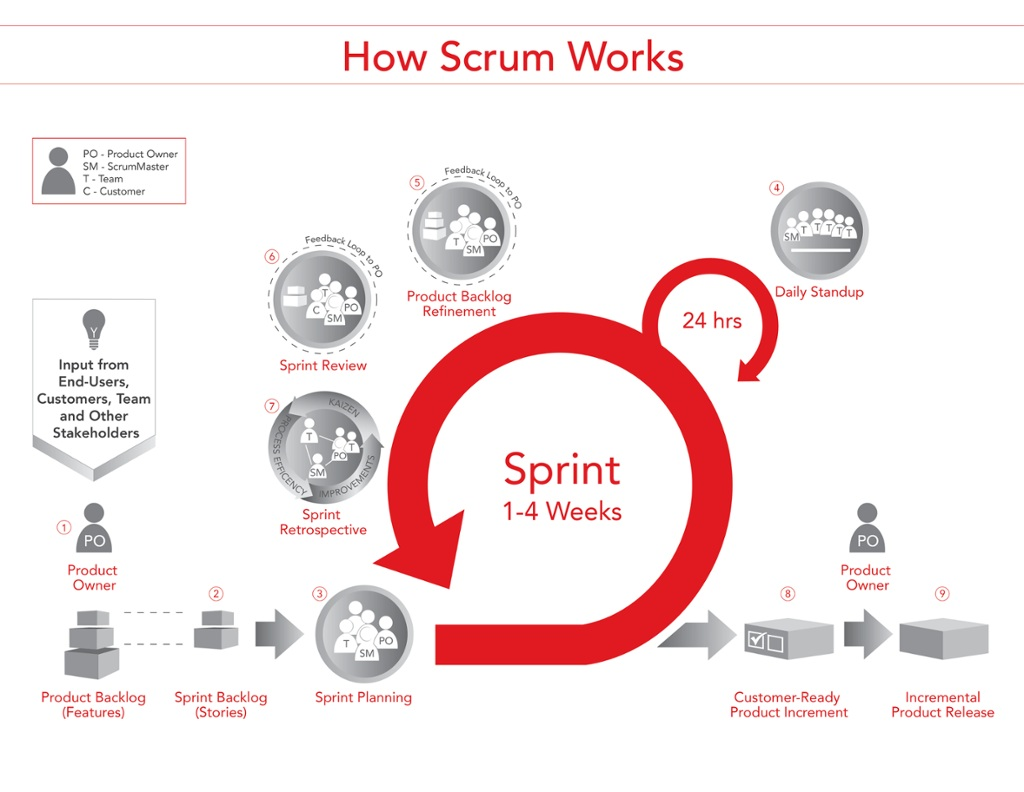
The main reason for choosing the waterfall process model would be because the requirements for a problem are well understood—when work flows from communication through deployment in a reasonably linear fashion. The requirement of the system is already well defined as stated in our use case diagrams etc. Also, the only disadvantage of using the disadvantage of the process model is that the software won't be available till late in the development phase, but it is okay since we are only delivering one increment of our software after extensive testing.



***THE TYPE OF AGILE FRAMEWORK – SCRUM***

In our project which is Online Attendance System, the Scrum framework can be applied suitably. Scrum is a framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value. In this, small working teams are organized to "maximize communication, minimize overhead. Often thought of as an agile project management framework, Scrum describes a set of meetings, tools, and roles that work in concert to help teams structure and manage their work. There are various prerequisites for choosing scrum. This employs a backlog in which there is a list of project requirements that provide business value. Items can be added anytime. This makes it user friendly. The specifications are already taken care of in the functional requirements and the use cases, but if there are any additional features, they can always be added in the list of backlog. The heart of Scrum is a Sprint, a time-box of two weeks or one month during which a potentially releasable product increment is created. A new Sprint starts immediately after the conclusion of the previous sprint. During this time, the backlog items that the sprint work units address are frozen. All the data from the use cases are implemented here. Meetings are organized in which the modules of the system can be looked into. After the completion of every backlog, the next accomplishments of the backlog list is discussed. For example, after the completion of the login() module, a scrum meeting can be held. The Sprint Retrospective occurs after the Sprint Review and

prior to the next Sprint Planning. In this meeting, the Scrum Team is to inspect itself and create a plan for improvements to be enacted during the subsequent Sprint.



***UI DESIGN (FACTORS):***

|  |
| --- |
|  |
|  |
|  |
|  |

Design is very important for any project. In this project of Attendance System, We are using three golden rules which are followed to accomplish this.

•User in control:

If the project had to be exported commercially for a

particular user, then, as a designer, the goal is to simplify the

mode of interaction. Modes of interaction shouldn’t force a user

into unnecessary or undesired actions. The interaction should be

flexible and straight forward. Since the project requires faculty and student to interact. Interaction is controlled by Admin.

•Reduced user’s memory load:

The project will keep attendance data in digital. So, user need not write in papers.

•Consistent interface:

In this project, Student can see their attendance detail and also rise an query. This implies that the interface will acquire information in a consistent order.

***SOFTWARE QUALITY:***

High quality of software is an important goal to be accomplished in the

development of a new software project. It is the result of good project

management and solid software engineering practice. Management and practice are applied within the context of five broad activities that help a software team to achieve high software quality:

Software engineering methods,

Project management

Techniques,

Quality control actions, and

Software quality assurance.

•Software Engineering Method:

An effective software process applied in a manner that creates a useful product that provides measurable value for those who produce it and those who use it. An effective software process establishes the infrastructure that supports any effort at building a high-quality software product. A useful product delivers the content, functions, and features that the end user desires, but as important, it delivers these assets in a reliable, error-free way. There are many factors a software must abide to achieve software quality:

1.Garvin’s Quality Dimensions: Our project of “The Attendance system” abides by these dimensions by following performance, feature, reliability, and conformance qualities.

2.McCall’s Quality Factors: The auction system focuses on operational qualities, its ability to undergo changes and to adapt to new environments. This is achieved by correctness, efficiency, usability and integrity. The interface can be specified to the needs of the user as per required.

3.ISO 9126 Quality Factors: Our project identifies six key attributes of

software, functionality, usability, efficiency, reliability, maintainability and

portability. The software is usable for a considerable amount of audience.

4.Targeted Quality Factors: The interface layout of the project is conductive to easy understanding. The operations such as logging in, getting the price of a particular article, etc. are easy to locate and initiate. The inputs are taken through the mouse clicks.

• Project Management Techniques: This comprises of three parts, estimation decisions, scheduling decisions and risk-oriented decisions. A particular date is set to deliver the project. The tasks to be accomplished in the auction(project) are sequenced based on dependencies.

•Quality Control: It encompasses a set of software engineering actions that help to ensure work meets quality goals. The code for the online auction system is reviewed and inspected to cover errors. A series of testing steps are applied.

•Quality Assurance: In this, a set of auditing and reporting functions that assess the

completeness of the project are encompassed.

***ARCHITECTURAL MODEL:***

In this project for the online attendance system, we try to find an appropriate

architectural style, one which suits the requirements of this model. Now in our system, we want that as soon as user login to system then he get the all available option for him. Like as faculty login to system then the details of students and course should be visible as changes to the database as soon as possible and also visible to the admin.

For this major requirement, we think that the data centered architecture style fits well. Because it is a blackboard mechanism for each of the users (faculty and student).

Data Centered Architecture- A data store (e.g., a file or database) resides at the center of this architecture and is accessed frequently by other components that update, add, delete, or otherwise modify data within the store.

Figure illustrates a typical data-centered style. User software accesses a central repository. In some cases, the data repository is passive. That is, user software accesses the data independent of any changes to the data or the actions of admin software. When Faculty is updating the attendance database.

At that time admin and student can not view it.

***CODE:***

Login Module:

Front End Development:

<html>

<head>

<title>Login</title>

<meta name="viewport" content="width=device-width, initial-scale=1"> <!-- to make it work on mobile devices -->

<link rel="icon" type="image/png" href="images/icons/favicon.ico"/>

<!-- <link rel="stylesheet" type="text/css" href="css/util.css"> -->

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

</head>

<body>

<div class="limiter">

<div class="container">

<div class="wrap">

<form class="login-form" method="post" action="log.php">

<span class="login-title">

Login

</span>

<div class="wrap-input" style="margin:16px 0px">

<input class="input" type="text" name="username" placeholder="Username">

<span class="focus-input-effect"></span>

</div>

<div class="wrap-input" style="margin:16px 0px">

<input class="input" type="password" name="pass" placeholder="Password">

<span class="focus-input-effect"></span>

</div>

<div class="form-checkbox">

<input class="input-checkbox" id="ckb1" type="checkbox" name="remember-me">

<label class="label-checkbox" for="ckb1">

Remember me

</label>

</div>

<div align="right">

<a href="#" class="txt1">

Forgot?

</a>

</div>

<button class="login-btn" style="width: 100%" type="submit">

Login

</button>

</form>

</div>

</div>

</div>

<div id="dropDownSelect1"></div>

</body>

</html>

@font-face {

font-family: Regular;

src: url('regularFont.ttf');

}

@font-face {

font-family: Bold;

src: url('boldFont.ttf');

}

\* {

margin: 0px; /\* nice trick to keep the page without scroll bars \*/

padding: 0px;

box-sizing: border-box;

}

body, html {

height: 100%;

font-family: Regular, sans-serif; /\* sans-serif is IIITN's backup font \*/

}

a {

font-family: Regular;

font-size: 14px;

line-height: 1.7;

color: #666666;

margin: 0px;

transition: all 0.4s;

-webkit-transition: all 0.4s; /\* for Chrome \*/

-o-transition: all 0.4s; /\* for Opera \*/

-moz-transition: all 0.4s; /\* for Mozilla \*/

}

a:hover {

/\*-text-decoration: none;\*/

color: #403866;

}

input {

outline: none;

border: none;

}

button {

outline: none !important;

border: none;

background: transparent;

}

button:hover {

cursor: pointer; /\* change cursr to hand \*/

}

.limiter {

width: 100%;

margin: 0 auto;

}

.container {

width: 100%;

min-height: 100vh;

display: -webkit-box;

display: -webkit-flex;

display: -moz-box;

display: -ms-flexbox;

display: flex;

flex-wrap: wrap;

justify-content: center;

align-items: center;

padding: 15px;

background-position: center;

background-size: cover;

background-repeat: no-repeat;;

}

.wrap {

width: 390px;

background: #fff;

border-radius: 10px;

position: relative;

}

.login-form {

width: 100%;

}

.login-title {

font-family: Bold;

font-size: 30px;

color: #403866;

line-height: 1.2;

text-transform: uppercase;

text-align: center;

width: 100%;

display: block;

}

/\* Input \*/

.wrap-input{

width: 100%;

position: relative;

background-color: #e6e6e6;

border: 1px solid transparent;

border-radius: 3px;

}

.input {

font-family: Bold;

color: #403866;

line-height: 1.2;

font-size: 18px;

display: block;

width: 100%;

background: transparent;

height: 62px;

padding: 0 20px 0 38px;

}

/\* Boundary Effect \*/

.focus-input-effect {

position: absolute;

display: block;

width: calc(100% + 2px);

height: calc(100% + 2px);

top: -1px;

left: -1px;

pointer-events: none;

border: 1px solid #827ffe;

border-radius: 3px;

visibility: hidden;

opacity: 0;

-webkit-transition: all 0.4s;

-o-transition: all 0.4s;

-moz-transition: all 0.4s;

transition: all 0.4s;

-webkit-transform: scaleX(1.1) scaleY(1.3);

-moz-transform: scaleX(1.1) scaleY(1.3);

-ms-transform: scaleX(1.1) scaleY(1.3);

-o-transform: scaleX(1.1) scaleY(1.3);

transform: scaleX(1.1) scaleY(1.3);

}

.input:focus + .focus-input-effect {

visibility: visible;

opacity: 1;

-webkit-transform: scale(1);

-moz-transform: scale(1);

-ms-transform: scale(1);

-o-transform: scale(1);

transform: scale(1);

}

.eff-focus-selection {

visibility: visible;

opacity: 1;

-webkit-transform: scale(1);

-moz-transform: scale(1);

-ms-transform: scale(1);

-o-transform: scale(1);

transform: scale(1);

}

/\* Checkbox \*/

.input-checkbox {

display: none;

}

.label-checkbox {

font-family: Regular;

font-size: 16px;

color: #999999;

line-height: 1.2;

display: block;

position: relative;

padding-left: 26px;

cursor: pointer;

}

.label-checkbox::before {

content: "\f00c";

font-family: FontAwesome;

font-size: 13px;

color: transparent;

display: -webkit-box;

display: -webkit-flex;

display: -moz-box;

display: -ms-flexbox;

display: flex;

justify-content: center;

align-items: center;

position: absolute;

width: 18px;

height: 18px;

border-radius: 3px;

background: #fff;

border: 2px solid #827ffe;

left: 0;

top: 50%;

-webkit-transform: translateY(-50%);

-moz-transform: translateY(-50%);

-ms-transform: translateY(-50%);

-o-transform: translateY(-50%);

transform: translateY(-50%);

}

.input-checkbox:checked + .label-checkbox::before {

color: #827ffe;

}

/\* Button \*/

.login-btn {

width: 100%;

display: -webkit-box;

display: -webkit-flex;

display: -moz-box;

display: -ms-flexbox;

display: flex;

flex-wrap: wrap;

font-family: Bold;

font-size: 16px;

color: #fff;

line-height: 1.2;

text-transform: uppercase;

justify-content: center;

align-items: center;

padding: 0 20px;

width: 100%;

height: 62px;

background-color: #827ffe;

border-radius: 3px;

text-decoration: none;

-webkit-transition: all 0.4s;

-o-transition: all 0.4s;

-moz-transition: all 0.4s;

transition: all 0.4s;

}

.login-btn:hover {

background-color: #403866;

}

Back End Development(in php):-

<?php

$conn = new mysqli("localhost", "root", ""); // This should be changed to IIITN's server configuration

//$sql = "CREATE DATABASE attendanceSys;"; already created

//$conn->query($sql

// echo "DB created";

// else {

// echo "DB Already Created";

// }

// define variables and set to empty values

$username = $pass = "";

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

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

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

echo $username;

echo $pass;

// check pass from database fetched value and echo "logged in" if matched, else display error message

}

function test\_input($data) { // Security check function to prevent database from hacking

$data = trim($data);

$data = stripslashes($data);

$data = htmlspecialchars($data);

return $data;

}

?>

//$conn=new mysqli("localhost", "root", "", "attendanceSys");

//$sql = "CREATE TABLE t1 (

// id1 INT(5) UNSIGNED,

// nm1 varchar(50)

//)";

//$conn->query($sql);

//$sql = "CREATE TABLE t2 (

// id2 INT(5) UNSIGNED,

// nm2 varchar(50)

//)";

//$conn->query($sql);

//$sql = "CREATE TABLE t3 (

// id3 INT(5) UNSIGNED,

// nm3 varchar(50)

//)";

//$conn->query($sql);

//$sql = "CREATE TABLE t4 (

// id4 INT(5) UNSIGNED,

// nm4 varchar(50)

//)";

//$conn->query($sql);

//$sql = "CREATE TABLE t5 (

// id5 INT(5) UNSIGNED,

// nm5 varchar(50)

//)";

$conn->query($sql);

$conn->close();

?>

***System Testing***

**Introduction:**

Once source code has been generated, software must be tested to uncover (and correct) as many errors as possible before delivery to customer. Our goal is to design a series of test cases that have a high likelihood of finding errors. To uncover the errors software techniques are used. These techniques provide systematic guidance for designing test that

(1)Exercise the internal logic of software components, and

(2) Exercise the input and output domains of the program to uncover errors in program function, behavior and performance.

**Steps:**  Software is tested from two different perspectives

(1) Internal program logic is exercised using ―White box‖ test case design techniques.

(2) Software requirements are exercised using ―block box‖ test case design techniques.

In both cases, the intent is to find the maximum number of errors with the minimum amount of effort and time.

**Testing Methodologies:**

A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements. A strategy must provide guidance for the practitioner and a set of milestones for the manager. Because the steps of the test strategy occur at a time when deadline pressure begins to rise, progress must be measurable and problems must surface as early as possible.

Following testing techniques are well known and the same strategy is adopted during this project testing.

1. **Unit testing:**

Unit testing focuses verification effort on the smallest unit of software design-the software component or module. The unit test is white-box oriented. The unit testing implemented in every module of student attendance management System. by giving correct manual input to the system ,the datas are stored in database and retrieved. If you want required module to access input or get the output from the End user. any error will accrued the time will provide handler to show what type of error will accrued.

1. **System testing:**

System testing is actually a series of different tests whose primary purpose is to fully exercise the computer-based system. Below we have described the two types of testing which have been taken for this project. It is to check all modules worked on input basis. If you want change any values or inputs will change all information. So specified input is must.

1. **Performance testing:**

Performance testing is designed to test the run-time performance of software within the context of an integrated system. Performance testing occurs throughout all steps in the testing process. Even at the unit level, the performance of an individual module may be assessed as white-box tests are conducted.This project reduce attendance table, codes. it will generate report fast.no have extra time or waiting of results .entered correct data will show result few millisecond. just used only low memory of our system. Automatically do not getting access at another software. Get user permission and access to other applications.

**Test Cases:**

Test case is an object for execution for other modules in the architecture does not represent any interaction by itself.

A test case is a set of sequential steps to execute a test operating on a set of predefined inputs to produce certain expected outputs.

There are two types of test cases: -manual and automated.

A manual test case is executed manually while an automated test case is executed using automation. In system testing, test data should cover the possible values of each parameter based on the requirements.

Since testing every value is impractical, a few values should be chosen from each equivalence class.

An equivalence class is a set of values that should all be treated the same. Ideally, test cases that check error conditions are written separately from the functional test cases and should have steps to verify the error messages and logs.

Realistically, if functional test cases are not yet written, it is ok for testers to check for error conditions when performing normal functional test cases. It should be clear which test data, if any is expected to trigger errors.

**Login Form Table:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Sno.** | **Test case id** | **Test case name** | **Test case desc** | **Step** | **Expected result** | **Actual Result** | **Test case status** |
| 1 | Admin Login | Validate login | To verify login | Enter login name and password and click submit | Login successful or an error message “invalid login” must be printed. | Login successful | Pass |
| **2** | Faculty login | Validate login | To verify login | Enter login name and password and click submit | Login successful or an error message “invalid login” must be printed. | Login successful | Pass |
| **3** | Student login | Validate login | To verify login | Enter login name and password and click submit | Login successful or an error message “invalid login” must be printed. | Login successful | Pass |
| 4. | Password | Validate password | To verify password | Enter password and login name | An error message  “Password invalid” must be print | An error message  “Password invalid” must be print | Fail |