B. Tech. project Grace Marks Allocator– G 22

PROJECT REPORT

Submitted by

S. No	Name	Roll Number
1.	G Lokesh	CB.EN.U4CSE18117
2.	P Pruthve Rej	CB.EN.U4CSE18136
3.	N Pavan Kruthik	CB.EN.U4CSE18137
4.	P Upendra	CB.EN.U4CSE18138
5.	Sk Awez	CB.EN.U4CSE18148

is fulfilment of the requirements for the Course – 15CSE376 (Net Centric Programming)

BACHELOR OF TECHNOLOGY IN

COMPUTER SCIENCE AND ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

AMRITA SCHOOL OF ENGINEERING

AMRITA VISHWA VIDYAPEETHAM

COIMBATORE -641112

October 2021

Table of Contents

JDBC:

JDBC is an application programming interface (JDBC API) that defines a set of standard operations for interacting with relational database management systems (DBMSs). The DBMSs may be located on a remote machine connected to the Internet. In order to access a database under a specific DBMS, for example, PostgreSQL, one must have a driver for that DBMS and the driver must implement JDBC API. JDBC is a trademark name and not an acronym. JDBC or Java Database Connectivity is a specification from Sun microsystems that provides a standard abstraction (that is API or Protocol) for java applications to communicate with various databases. It provides the language with java database connectivity standard. It is used to write programs required to access databases. JDBC along with the database driver is capable of accessing databases and spreadsheets. The enterprise data stored in a relational database (RDB) can be accessed with the help of JDBC APIs. JDBC is an API (Application programming interface) which is used in java programming to interact with databases. The classes and interfaces of JDBC allows application to send request made by users to the specified database. Enterprise applications that are created using the JAVA EE technology need to interact with databases to store application-specific information. So, interacting with a database requires efficient database connectivity which can be achieved by using the ODBC (Open database connectivity) driver. This driver is used with JDBC to interact or communicate with various kinds of databases such as Oracle, MS Access, MySQL and SQL server database.

Servlets:

Java Servlets, or simply servlets are a set of Java classes that can be used and extended for Web server-side programming provided the Web server supports Java servlets. Basically, the programmer specifies which servlet is to be used to process which request or which type of requests from client. Thus, when a request is received by the Web server, the Web server finds the proper servlet for the request. For example, in an HTML form, its action clause can explicitly specify which servlet be invoked to process the data submitted through the form. Servlets are the Java programs that run on the Java-enabled web server or application server. They are used to handle the request obtained from the webserver, process the request, produce the response, then send a response back to the webserver. The server-side extensions are nothing but the technologies that are used to create dynamic Web pages. Actually, to provide the facility of dynamic Web pages, Web pages need a container or Web server. To meet this requirement, independent Web server providers offer some proprietary solutions in the form of APIs(Application Programming Interface).

These APIs allow us to build programs that can run with a Web server. In this case, Java Servlet is also one of the component APIs of Java Platform Enterprise Edition which sets standards for creating dynamic Web applications in Java. Servlets are built from two packages:

1. javax.servlet(Basic)
2. javax.servlet.http(Advanced)

An attribute in servlet is an object that can be set, get or removed from one of the following scopes (i.e.) request scope, session scope, application scope. The servlet programmer can pass information from one servlet to another using attributes. It is just like passing object from one class to another so that we can reuse the same object again and again.

There are following 4 attribute specific methods. They are as follows:

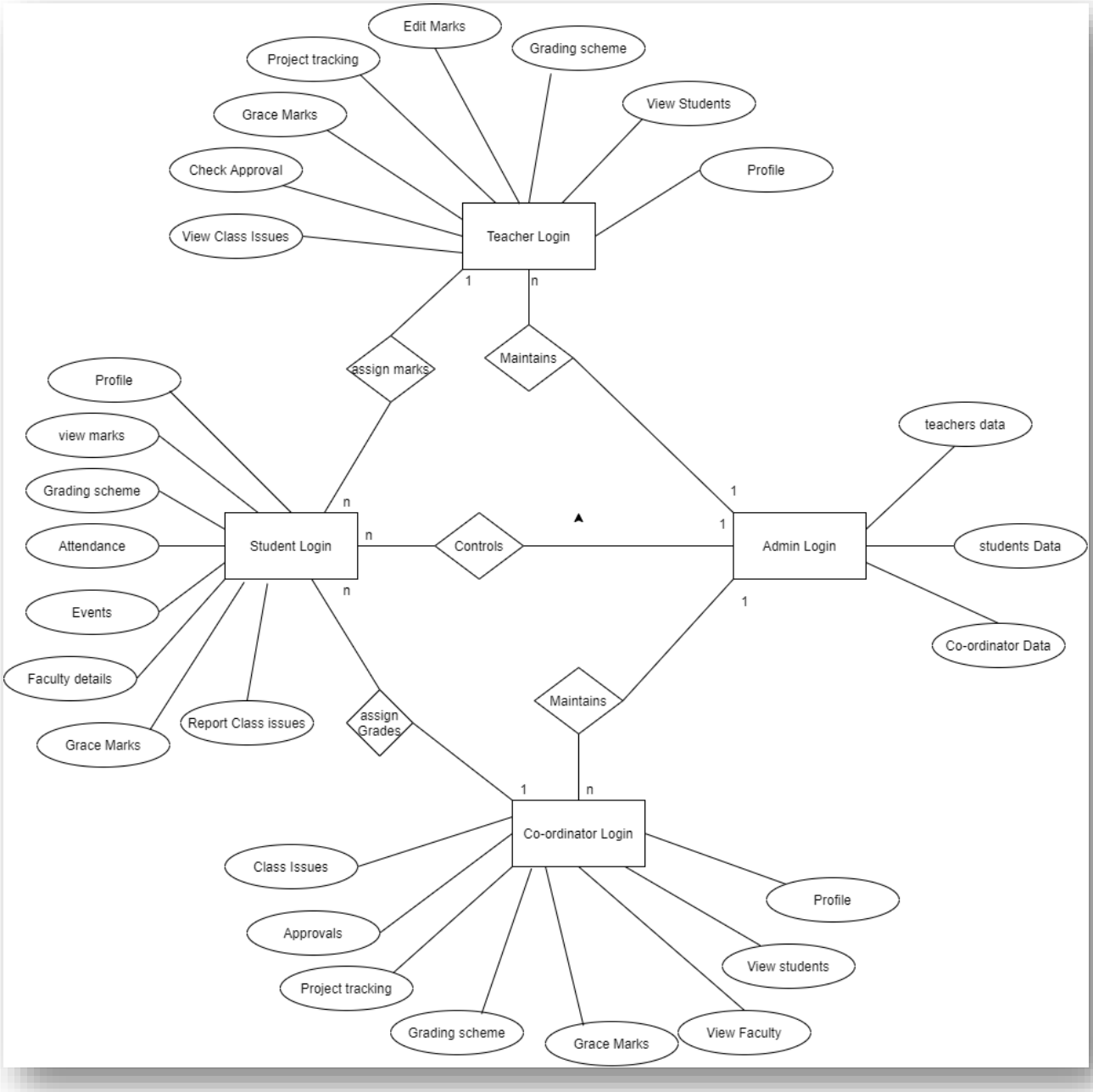
- public void setAttribute (String name, Object object): sets the given object in the application scope.
- public Object getAttribute (String name): Returns the attribute for the specified name.
- public Enumeration getInitParameterNames(): Returns the names of the context's initialization parameters as an Enumeration of String objects.
- public void removeAttribute (String name): Removes the attribute with the given name from the servlet context.

The web container maintains the life cycle of a servlet instance. Let's see the life cycle of the servlet:

- Servlet class is loaded.
- Servlet instance is created.
- Init method is invoked.
- Service method is invoked.
- Destroy method is invoked.

There are 2 types of cookies in servlets namely non-persistent cookie and persistent cookie. Non-persistent cookie is valid for single session only. It is removed each time when user closes the browser. Persistent cookie is valid for multiple session. It is not removed each time when user closes the browser. It is removed only if user logouts.

ER Diagram:


Edit Marks
Project tracking
Grading scheme
Grace Marks
View Students
Check Approval
Profile
View Class Issues
Teacher Login
1
n
assign marks
Maintains
Profile
view marks
Grading scheme
Attendance
Events
Faculty details
Grace Marks
Report Class issues
assign Grades
Student Login
n
n
n
Controls
1
1
1
Admin Login
teachers data
students Data
Co-ordinator Data
Maintains
1
n
Co-ordinator Login
Class Issues
Approvals
Project tracking
Grading scheme
Grace Marks
View Faculty
View students
Profile

Architecture diagram:

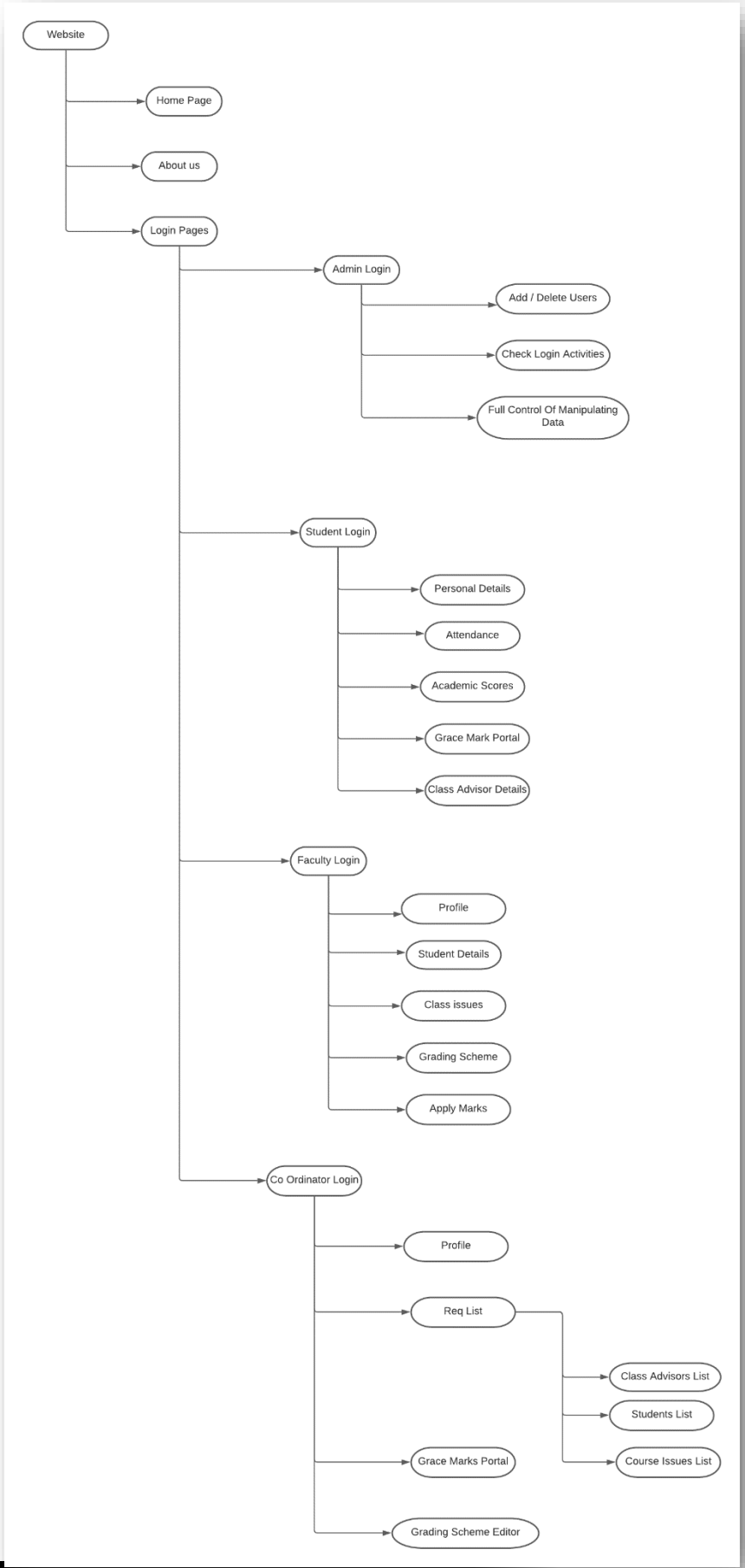
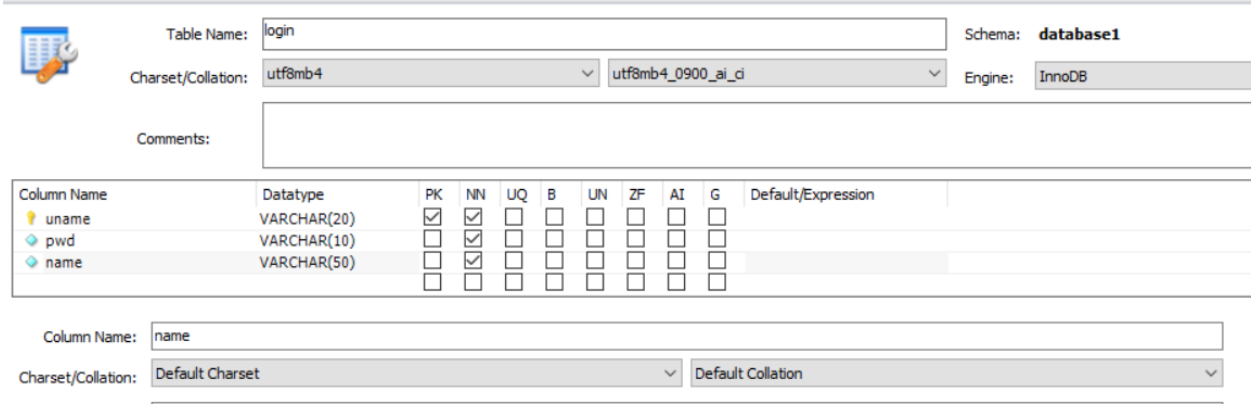
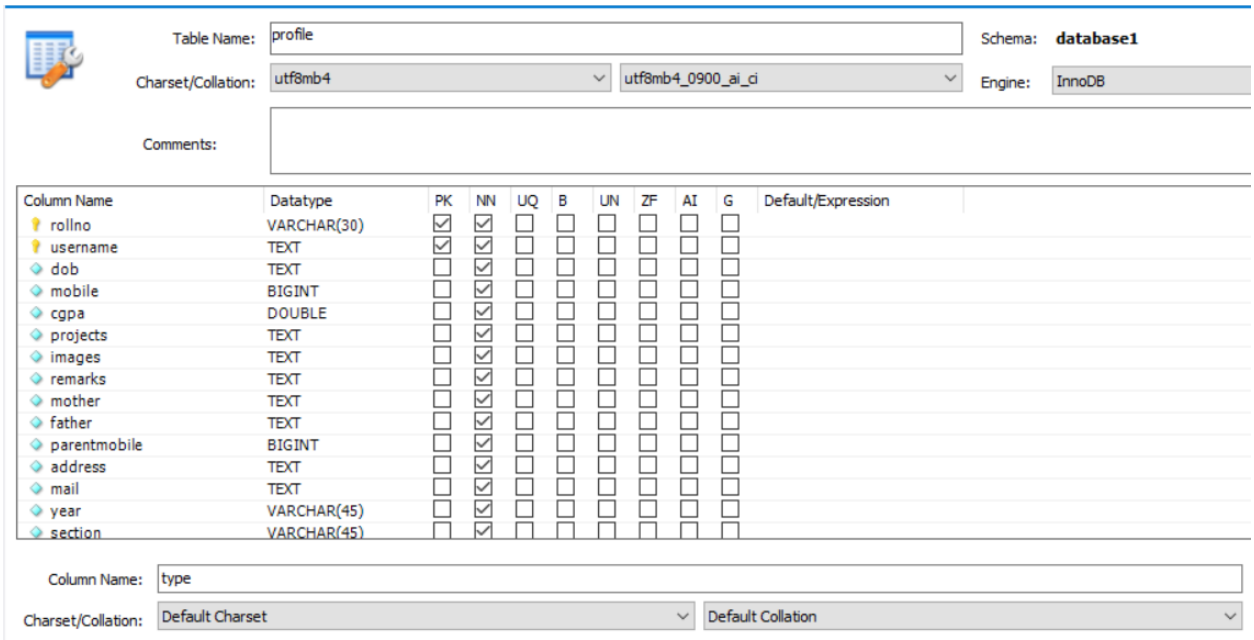


Table Specifications:

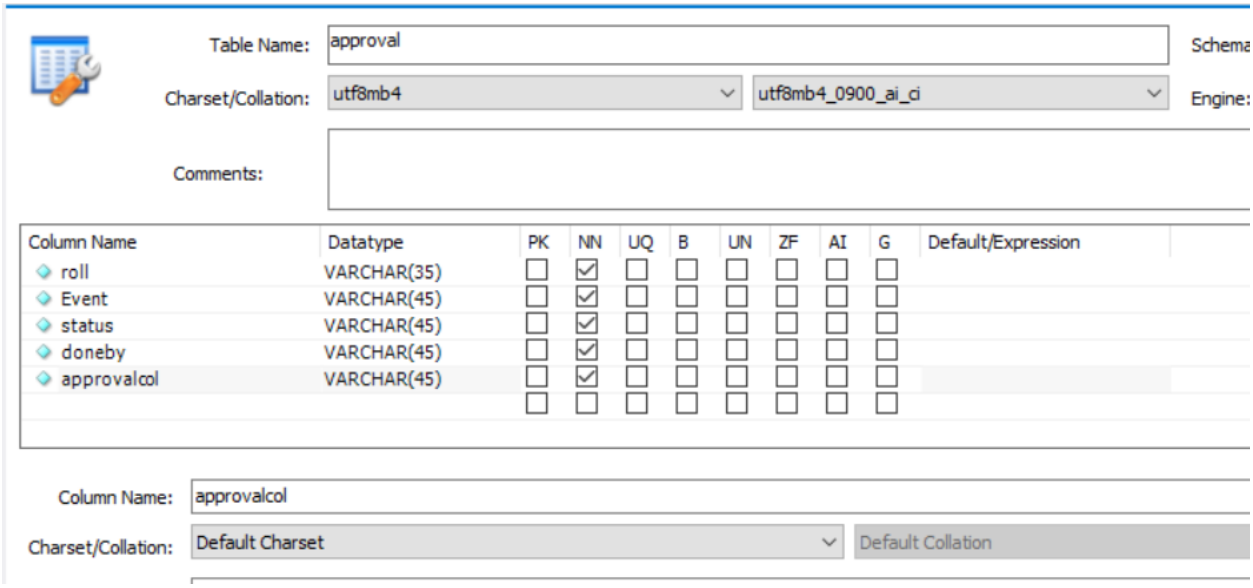
1)Users



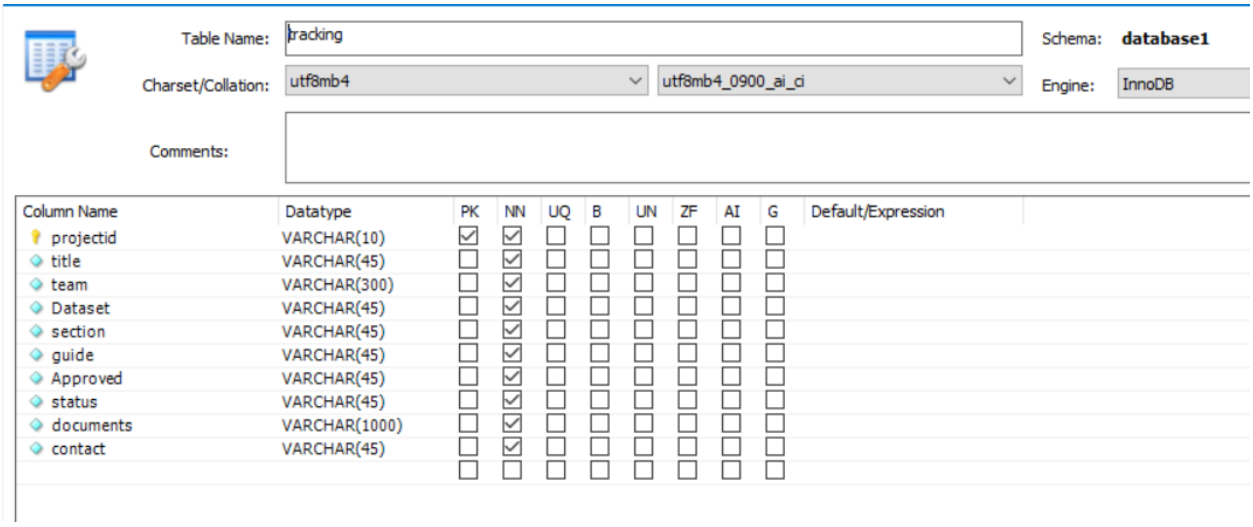
2)Profile



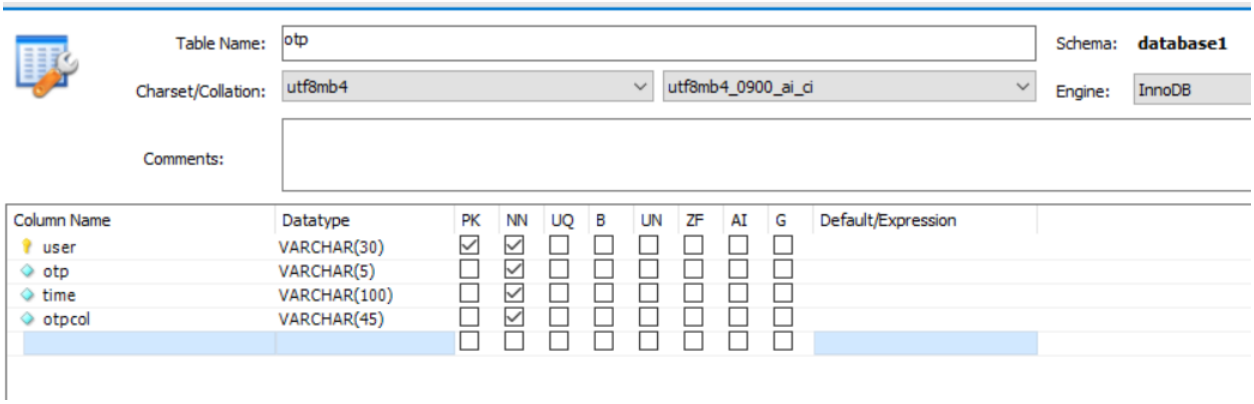
3) Approval



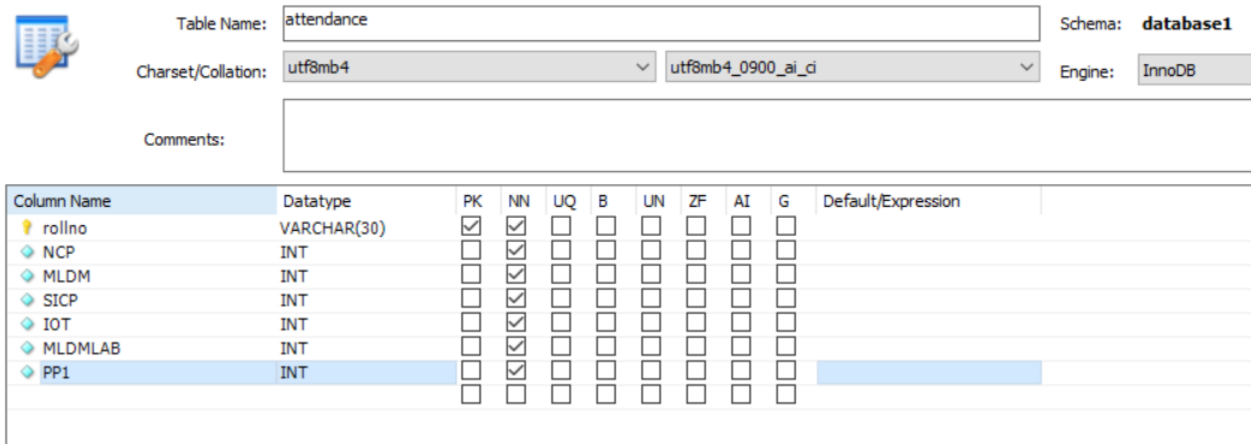
4)Project Tracking



5)Otp



6)Attendance



Functionalities using servlets

1) Login Action

a) Path :/login
b) Method: Get
c) Input: username, password

2) Enroll Student

a)Path:/enroll

b)Method:Get

c)Input:Students Data

3) Marks Re enter

a)Path:/editmarks

b)Method:Get

c)Input:Marks Record

4) Approval

a)Path:/approval
b)Method:Get
c)Input:Co Ordinator Approval

5) Class Issues

a)Path:/issue
b)Method:Get
c)Input:Students Feedback

6) Admin Access

a)Path:/admin
b)Method:Get
c)Input:Login Credentials

Database connectivity

Java:

```
    System.out.println("\n\n***** MySQL JDBC Connection Testing *****");
           Connection conn = null;
           try
           {
           Class.forName ("com.MySQL.jdbc.Driver").newInstance ();
               String userName = "root";
               String password = "Awez@0987";
               String url = "jdbc:MySQL://localhost/database1";
               conn = DriverManager.getConnection (url, userName, password);
               System.out.println ("\nDatabase Connection Established...");
           }
          catch (Exception ex)
           {
               System.err.println ("Cannot connect to database server");
               ex.printStackTrace();
           }
```

Javascript:

```
var mysql = require('mysql');
var con = mysql.createConnection({
  host: "localhost",
  user: "root",
  password: "Awez@0987"
});

con.connect(function(err) {
  if (err) throw err;
  console.log("Connected!");
});
```

Insertion:

```
String query="insert into approval (roll , Event , status) values (?,?,?,?,?,?)";
   PreparedStatement st=con.prepareStatement(query);
   st.setString(1,rollno);
   st.setString(2,event);
   st.setString(3,status);
```

Updation:

```
String query="UPDATE database1.marks SET  "+ subcode + "= ? WHERE (rollno = ?) ";
   PreparedStatement st=con.prepareStatement(query);

   st.setString(1,mark);
   st.setString(2,rollno);
   int check=st.executeUpdate();
   return check;
```

Validation:

```
   String username=  req.getParameter("username");
   String password= req.getParameter("password");

   LoginDao dao = new LoginDao();

   try {
       if(dao.connectdb(username, password)) {
           HttpSession session = req.getSession();
           res.sendRedirect("http://localhost:8000/Student");
       }else {
           res.sendRedirect("http://localhost:8000/?username=+&password=+");

       }
```

Validation

```
if(username != undefined && password != undefined) {
fs.truncate('currentlogin.txt', 0, function() {
console.log("File Content Deleted");
});

fs.readFile('block.txt', 'utf-8', (err, data) => {  if (err) throw err;

if(username==data){
res.send("Your account is blocked ..pls login after 24hrs");
}else{

var test = 0;
var pd=password;
fs.readFile('logintype.txt', 'utf-8', (err, data) => {
var sqlquery='SELECT * from ';
sqlquery +=data;

mycon.query(sqlquery, function (error,login, fields) {
if (error) throw error;
var length = login.length;
var m=null;
for(var i = 0; i < length; i++){
if (login[i].uname==username){
m=i;
}
}
if(m==null){
res.sendfile("oops.html");
}else{
if(login[m].pwd==password){
test=1
fs.appendFile('currentlogin.txt', username , (err) => {
if (err) throw err;
const sql5=`INSERT INTO database1.currentlogin (user) VALUES ('${username}')`;
mycon.query(sql5, function (err1, result) {
if (err1) throw err1;
console.log(result);
});
})
console.log(test);
if(test==1){       res.redirect("/loginwelcome");  }
```

Evaluation sheet:

Roll No	Technology	Max Marks	Marks Awarded	Total (30)
18117	Servlet JDBC	10 10		
18136	Servlet JDBC	10 10		
18137	Servlet JDBC	10 10		
18138	Servlet JDBC	10 10		
18148	Servlet JDBC	10 10		
	Project Documentation	10		