Final Report
(Payroll Management System)

Course Code: CS254 **Course Title:** DBMS LAB

Semester: B. Tech 4^{th} Sem **Section:** S2

Academic Year: 2020-21 **Course Instructor:** Dr. Annappa B and Mr. Sharath Yaji

Team Members:

1. Sanjkeet Jena, 191CS246, 7735778182, iamkeet.191cs246@nitk.edu.in

2. Akash Meena, 191CS206, 8824257029, akash.191cs206@nitk.edu.in

3. Pranav DV, 191CS234, 7760785980, pranavdv.191cs234@nitk.edu.in

1 Abstract

Brief Description:

Payroll Management System is an important area of any business. It allows for a company to manage employee's expenses, salary, gross salary, deduction, tax and other things for a given time period. The two essential components of this system are management and accounting. Payroll is a major concern for any company since it is mandatory to pay every employee according to the governments rules and regulations otherwise there would be serious financial and legal consequences. This system facilitates a company to handle all the legal processes and an employee's expenditure in a systematic manner.

This payroll system is designed to perform all tasks of employee payment and filing of the required taxes. These tasks include keeping track of leaves, calculating wages, taxes, deduction and bonuses, as well as printing and delivering cheques to employees. This system does not require too much input from the user (the administrator) since once the details like base salary, department, grade etc. have been entered, the system automatically calculates all other information.

The purpose of the system is to implement the following:

- Manage employee information efficiently
- Define the salary, deductions, leaves etc.
- Conveniently generate payslips
- Generate and manage salary structure of employees
- Generate reports related to employees

The only users authorized to access this system are administrators and this is done using a valid username and password. Admins can add new employees, departments and pay grades. They also control the duration of time for which employees have been employed under specific pay grades. Apart from this, they can also view past data of any employee and generate their monthly salaries via an automated process.

Some of the advantages of this system are:

1. It saves time and speeds up virtually every aspect through a range of automated services
2. It is secure as only admins are authorized to access and modify the information
3. It is user friendly, the admins need not know the technicalities of the payroll since it is automatically calculated and no former knowledge is required due to the intuitive user interface
4. There is very little scope for any errors in processing since it is fully automated

Overall, this software has been designed specifically to cater to a company's employee management, is self contained and works efficiently. It uses a simple database so the requirements are feasible and provides an intuitive graphical user interface for easy accessibility. Finally, automating this process saves the admin or manager a lot of time and resources.

Key Features:

1. Login facility
2. Add new employee
3. Add new department

4. Add new pay grades
5. Manage employee grade and department
6. Manage employee salary details
7. Generate payroll

Software Specifications:

- Frontend: HTML, CSS, JavaScript(React Framework)
- Backend: Node.js, MySQL

Full Github Code Link :

- Full Project Link

2 Introduction

Front end description:

The front end consists of six pages for user interactions, data verification, insertion and display. The first page is a login page used to verify an administrator's credentials since unauthorized personnel cannot perform any operations on the data stored in the back end. Once an admin has successfully logged in, they land on the menu page.

The menu page allows the admin to navigate across various features available. These include the following :

- Add new employee
- Add new department
- Add new pay grade
- Specify pay grade and department of a particular employee
- Generate the payroll for a particular employee

In each web pages we first create a simple form using HTML. Then after we do styling using CSS and simple BOOTSTRAP.And we made input field according to the database and also we check the data validation after each input is filled.
Every page has options to go back to the navigation page or logout of the application.

The first three features listed above are used to insert new information to the database. Specifying the pay grade and department of an employee depends on existing data, i.e. the employee, department and pay grade must already exist. The last feature also depends on existing data but the information added to the database is simply an indication that a payroll has been generated (new information is not added).

Back end description:

The back end database is implemented using MySQL. The details of a table's attributes and the relations can be seen in the ER diagram.

- The "admin" entity is used to store information on the administrators who have access to the database. This includes their username and password.
- The employee entity is used only to store an employee's personal information.
- The grade entity is used to store information on the components of the salary associated with a pay grade.
- The department entity is used to keep track of the department comprising the company
- The emp-grade is a relation that maps an employee to a pay grade and department.
- The transaction entity contains details on payrolls generated for every employee and their corresponding department and pay grade when the payroll was generated.

Before any data or interaction can be done, we establish an HTML server as well as a connection to the database stored locally on our system.

The various operations on the data are done using Node.js. Whenever information entered on a page is submitted, the form responsible for collecting that information sends a post request. This request is then parsed to get the corresponding input fields. Using these fields, a SQL query is created and made to the database through the previously established connection.

For example, consider the flow of control involved in adding a new department.

1. The administrator navigates to the "Add new department" page, types a department name and clicks submit.
2. This results in a post request corresponding to the URL of this page. The information in the input field i.e. dept-name is parsed from the body of the request.
3. This is then used to generate an SQL query as "INSERT INTO dept(dept-name) values ('"+dept-name+"');" and is sent to the server.

4. The server tries to execute the query.

5. If there are no errors, an appropriate message along with the inserted row is displayed. Else an alert is raised that an error has occurred.

A similar procedure is followed for all the other pages. The only page that does not perform an insertion to the database is the login page. Here the query is a select statement and only fetched the password of the username that has been entered. If there is a match, the user is authorized, else an error is displayed.

Integration of Back end and Front end:

The integration of back end and front end was done using embedded JavaScript also called as EJS.

Login page used EJS to tell the user if the username and password was correct. After checking with the back end appropriate alert was generated.

All the pages where we inserted the data also used EJS to notify the user if the inserted data was correct of if there was any mistake.

EJS was extensively used in generating the payroll report. The data was accepted from the user and using SQL queries the payroll was generated which was then sent to the user using EJS

3 ER Diagram

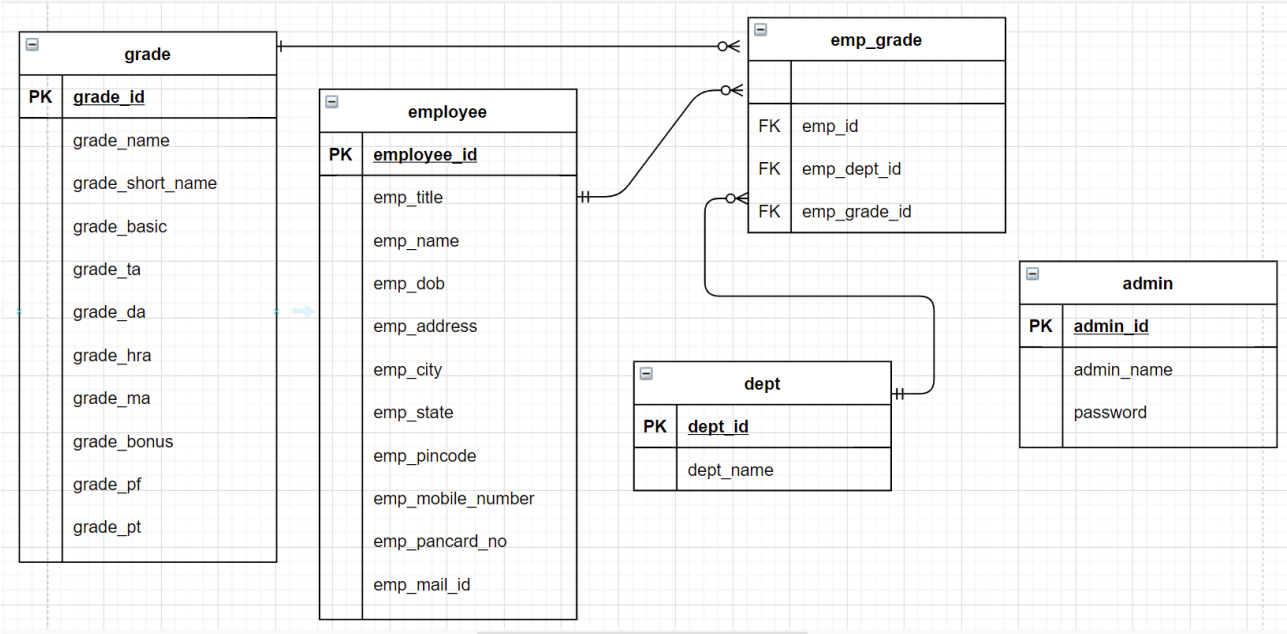

Figure 1: Entity Relation Diagram

From the ER diagram, the attributes corresponding to each entity can be seen. The foreign and primary key constraints applied on the tables have been highlighted as well. (PK = Primary key, FK = Foreign key).

4 Source Code

Front end:

The following is a snippet from the "Add New Employee" page that is used to input data from the user. The form element sends the data through a post request when the submit button is clicked. This triggers a SQL query in the backend code as depicted in the next page.

```
<form name="form3" action="login3" method="post">
<div class="container">
<center>  <h2> <b>Employee Details</b></h2> </center>
<hr>
<div class="row">
<div class="col">
    <label> <b>Employee Title</b> </label>
    <input type="text" name="employeetitle" placeholder= "
       EmployeeTitle" size="15" required />
</div>
<div class="col">
  <label> <b>Employee Name</b> </label>
  <input type="text" name="employeename" placeholder="Employeename"
     size="15" required />
</div>
</div>
<div class="row">
<div class="col">
   <b>Current Address</b>
   <input type="text" name="address" placeholder="Enter your Address
      " style="width: 500px; height: 150px;" / required/></input>
</div>
<div class="col">
  <label> <b>City</b></label>
  <input type="text" name="city" placeholder="City" size="15"
     required />
</div>
```

```
</div>
<div class="row">
    <div class="col">
    <label> <b>State</b> </label>
    <input type="text" name="state" placeholder="State" size="15"
       required />
</div>
<div class="col">
  <label> <b>Pincode</b> </label>
  <input type="text" name="pincode" placeholder="Pincode" maxlength=
     "6" pattern="\d{6}" title="Please enter exactly 7 digits" /
     required >
</div>
</div>
<div class="row">
<div class="col">
  <label>
  <b>Phone</b>
  </label>
  <input type="text" name="phone" placeholder="Enter your Contact
     number"  maxlength="10" pattern="\d{10}" title="Please enter
     exactly 10 digits" / required>
</div>
<div class="col">
    <label for="email"><b>Email</b></label>
    <input type="text" placeholder="ex:username@gmail.com" name="
       email" required>
</div>
</div>
<div class="row">
  <div class="col">
  <label ><b>Date Of Birth</b></label>
```

```
  <input type="date" name="DOB" placeholder="DD/MM/YYYY" size="10"/
     required>
</div>
<div class="col">
  <label ><b>Pan Card no.</b></label>
  <input type="text" name="pancardno" placeholder="Pan Card No."
     maxlength="10" pattern="\d{10}" title="Please enter exactly 10
     digits" / required>
</div>
</div>
<div class="row">
<div class="col">
    <button type="submit" class="registerbtn">Submit</button>
</div>
<div class="col">
    <button type="clear" class="cancelbtn">Clear</button>
</div>
</div>
</form>
```

Back End:

```
app.get('/login3', function(req, res) {
  res.render('login3', {error : 'None'});
});
app.post('/login3', (req,res) => {
  var title,name,city,state,pincode,phone,email,dob,pancardno,addr;
  title = req.body.employeetitle;
  name = req.body.employeename;
  city = req.body.city;
  state = req.body.state;
  pincode = req.body.pincode;
  email = req.body.email;
  dob = req.body.DOB;
  pancardno = req.body.pancardno;
  address = req.body.address;
  phone = req.body.phone;
  var sql = "INSERT INTO employee(emp_title,emp_name,emp_dob,
     emp_address,emp_city,emp_state,emp_pincode,emp_mobile_number,
     emp_pancard_number,emp_mail_id) values ('"+title+"','"+name
     +"','"+dob+"','"+address+"','"+city+"','"+state+"',"+pincode
     +",'"+phone+"','"+pancardno+"','"+email+"');"
  console.log(sql);
  con.query(sql, function (err, result, fields) {
    if (err){
      console.log(err);
      res.render('login3', {error : 'Error'});
    }
    else{
      console.log(fields);
      res.render('login3', {error : 'Done'});
      // console.log("inserted row");
    }
  }); });
```

5 Results

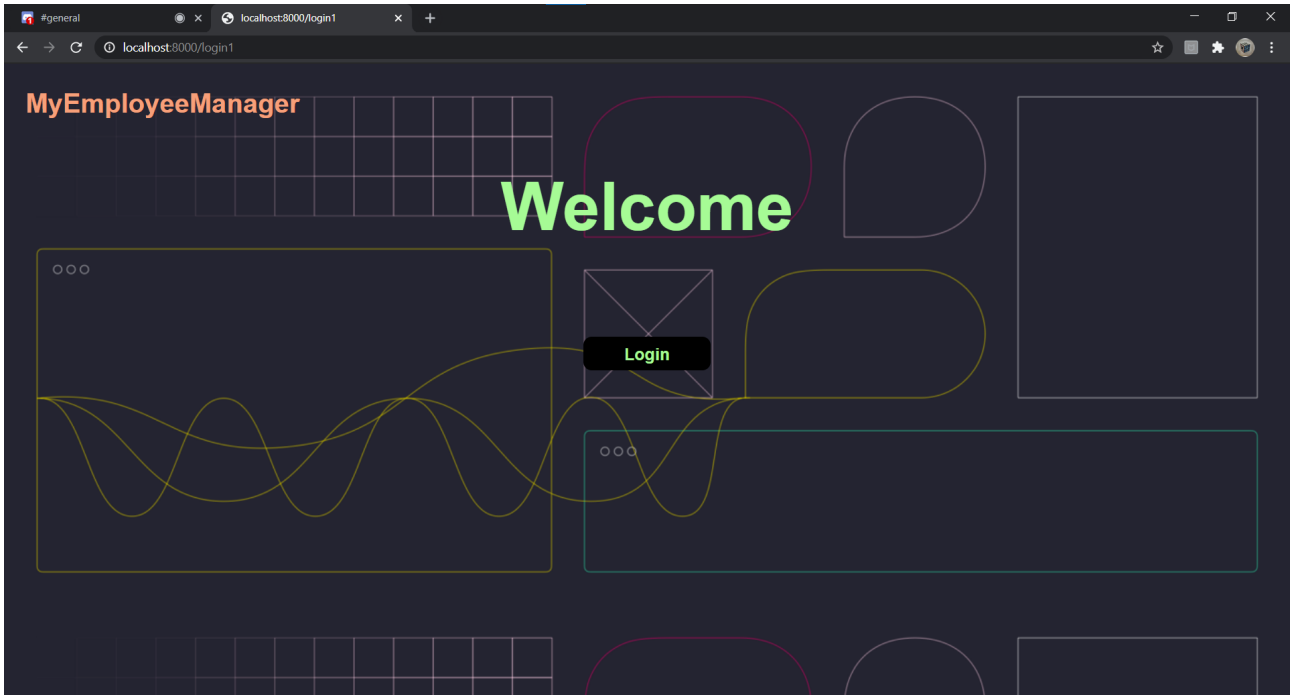

Figure 2: Login Page

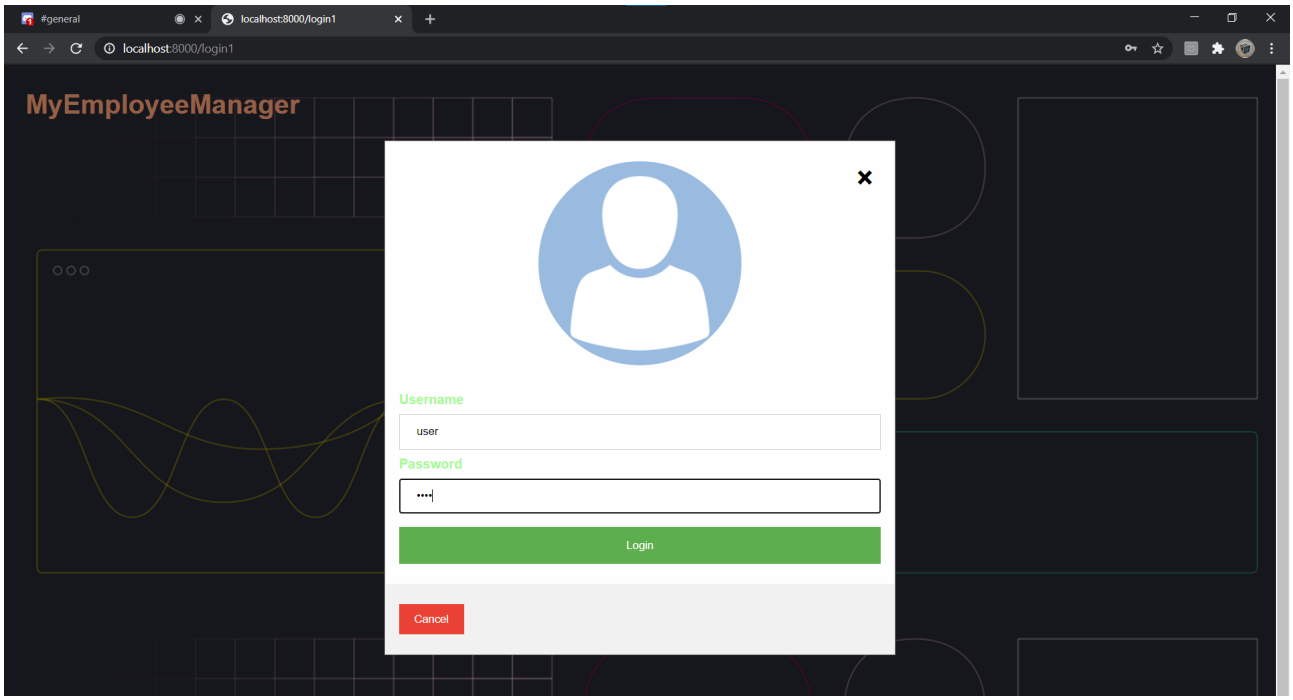

Figure 3: Login Credentials

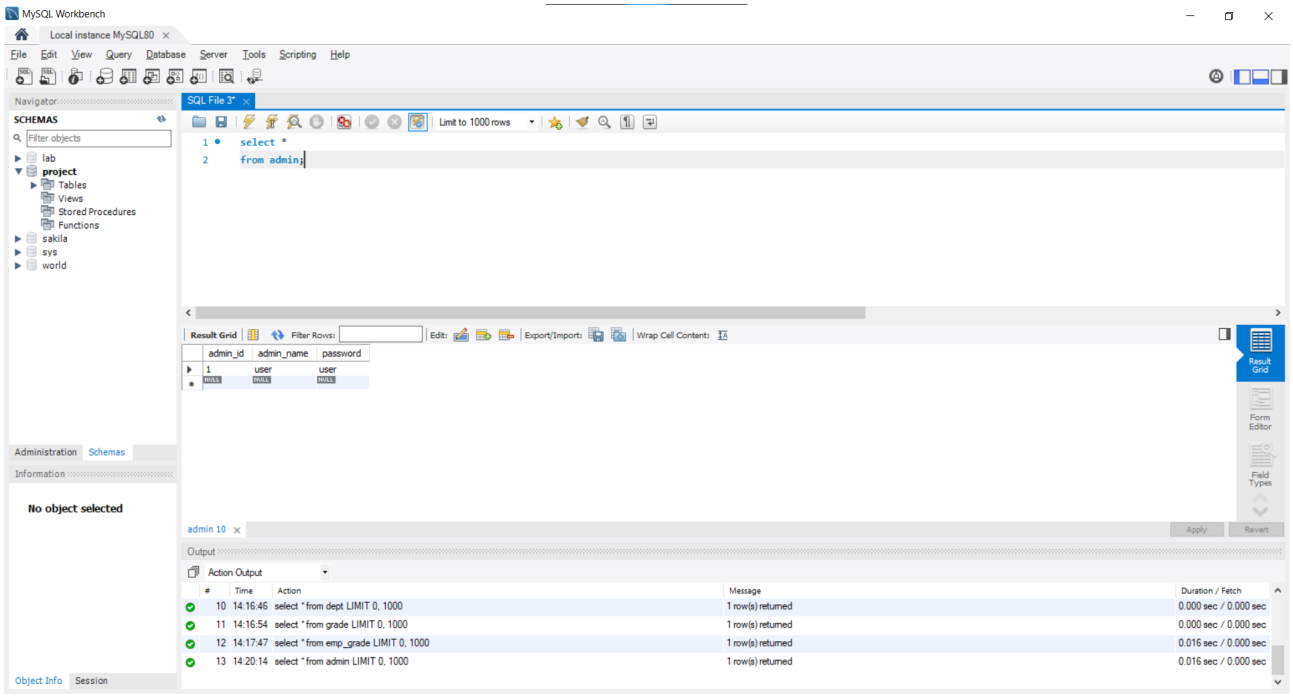

Figure 4: Admin Table

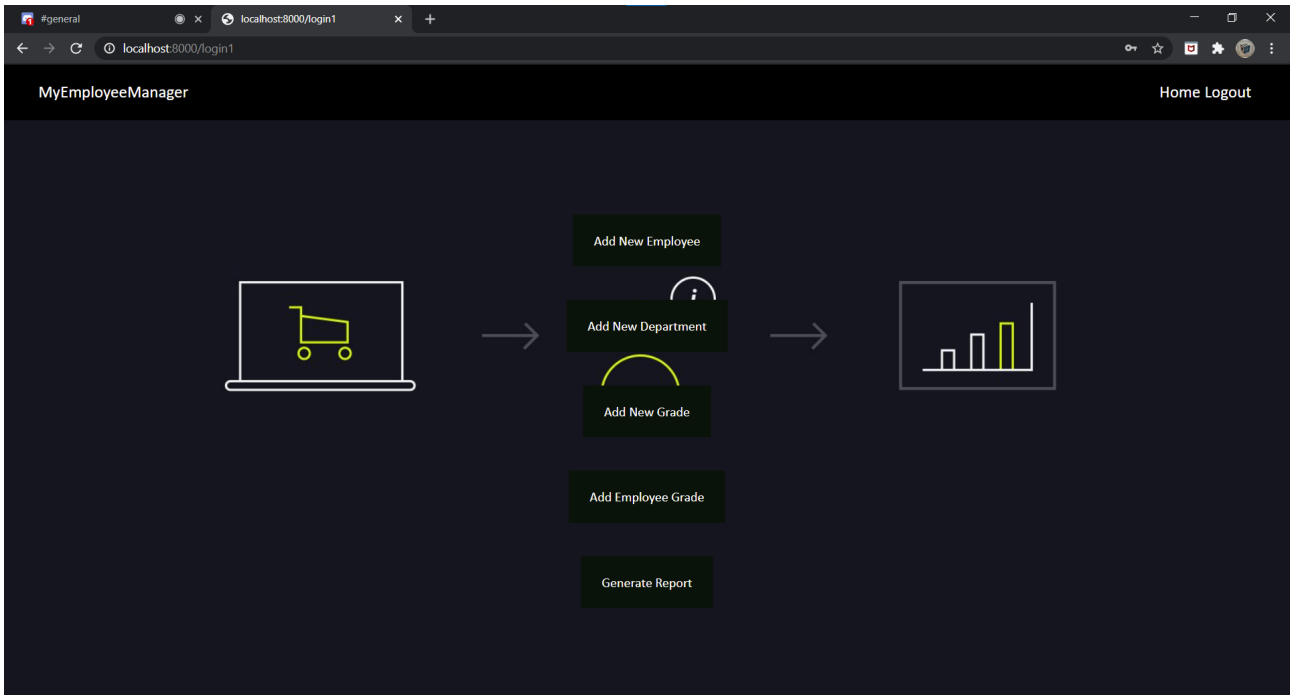

Figure 5: Navigation Page

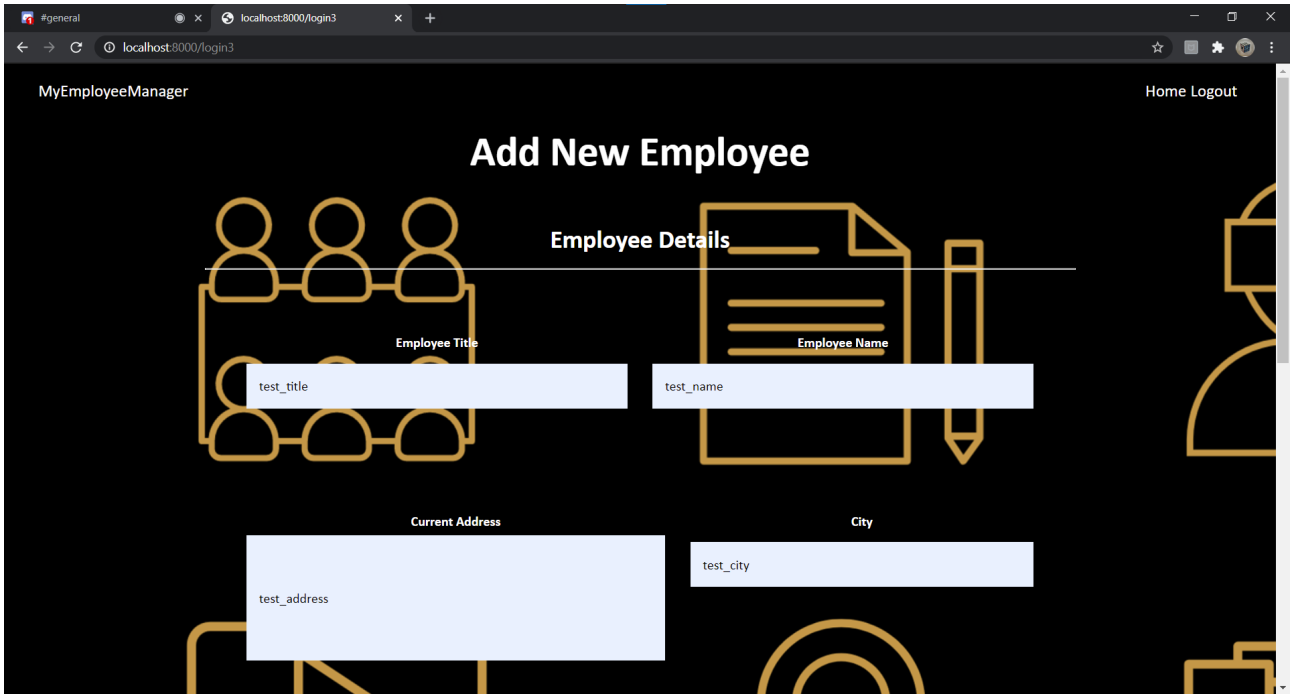

Figure 6: Add New Employee Page

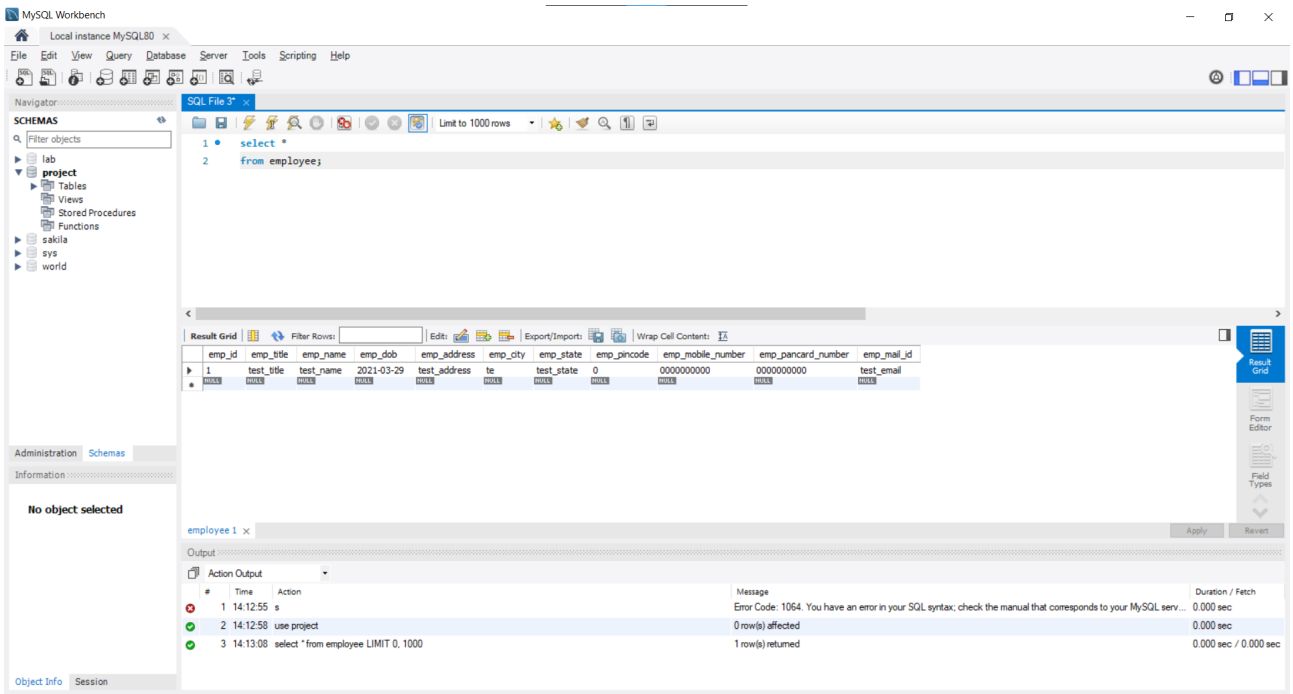

Figure 7: Employee Table

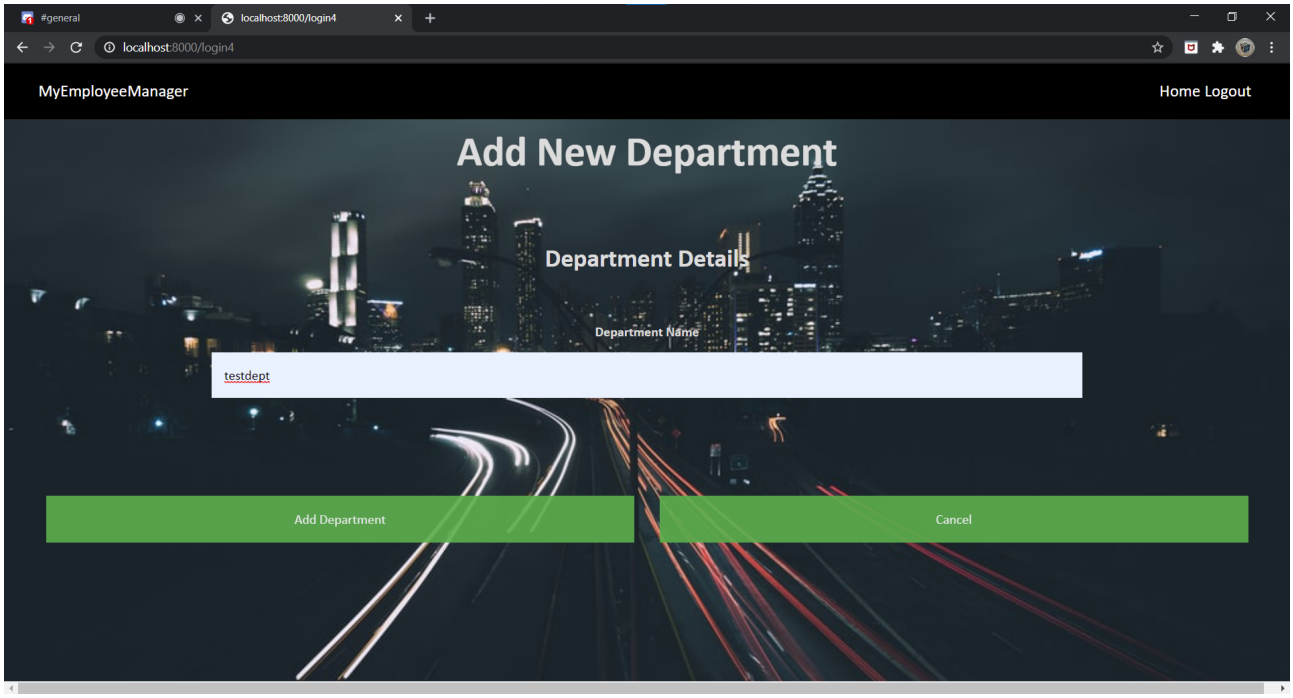

Figure 8: Add New Department Page

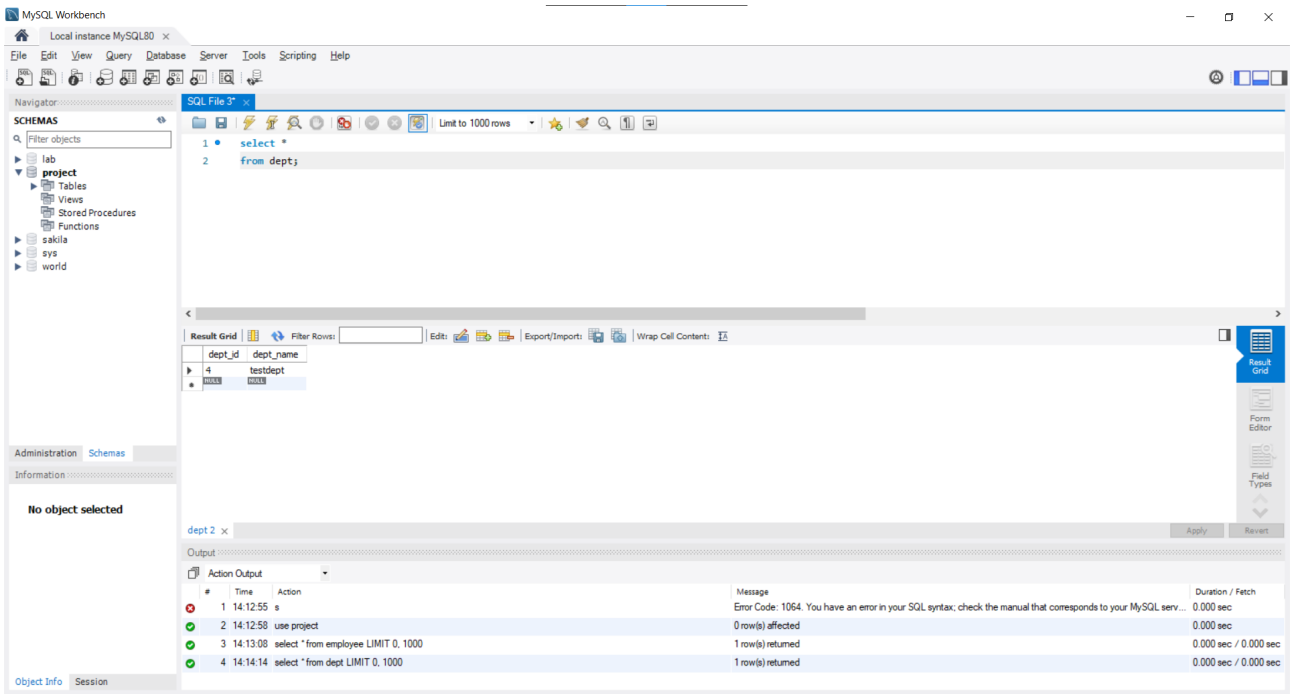

Figure 9: Dept Table

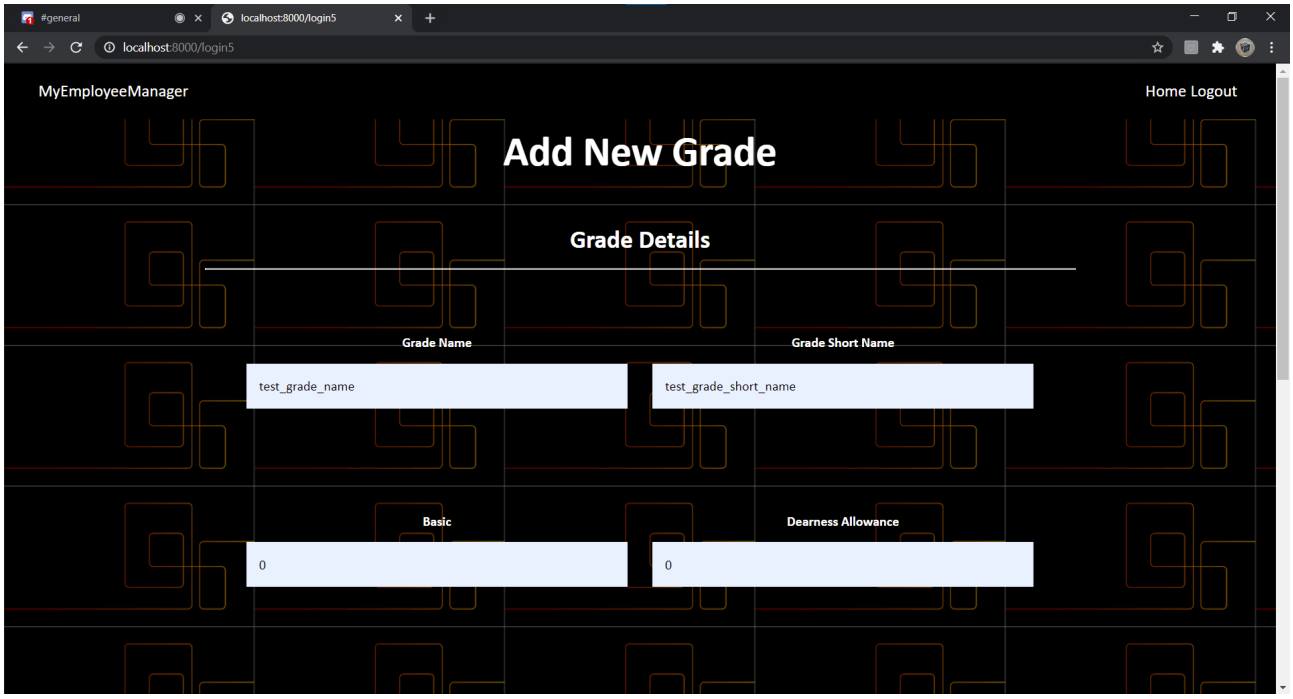

Figure 10: Add New Grade Page

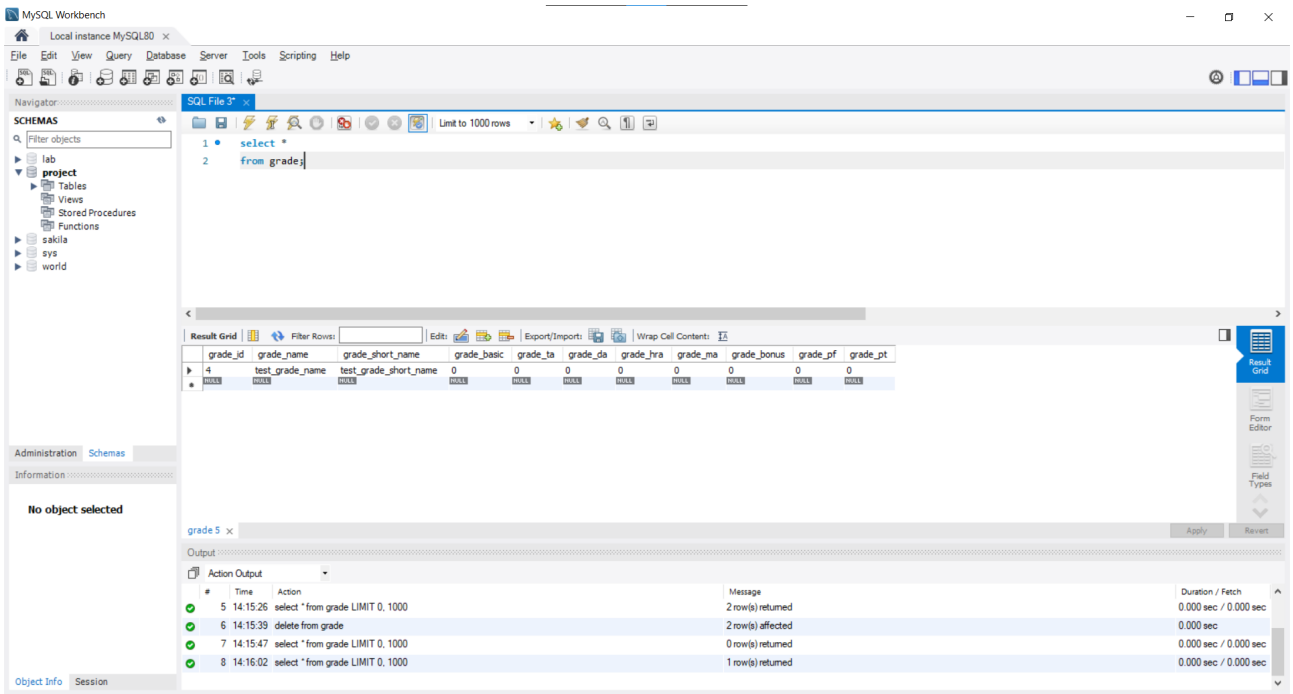

Figure 11: Grade Table

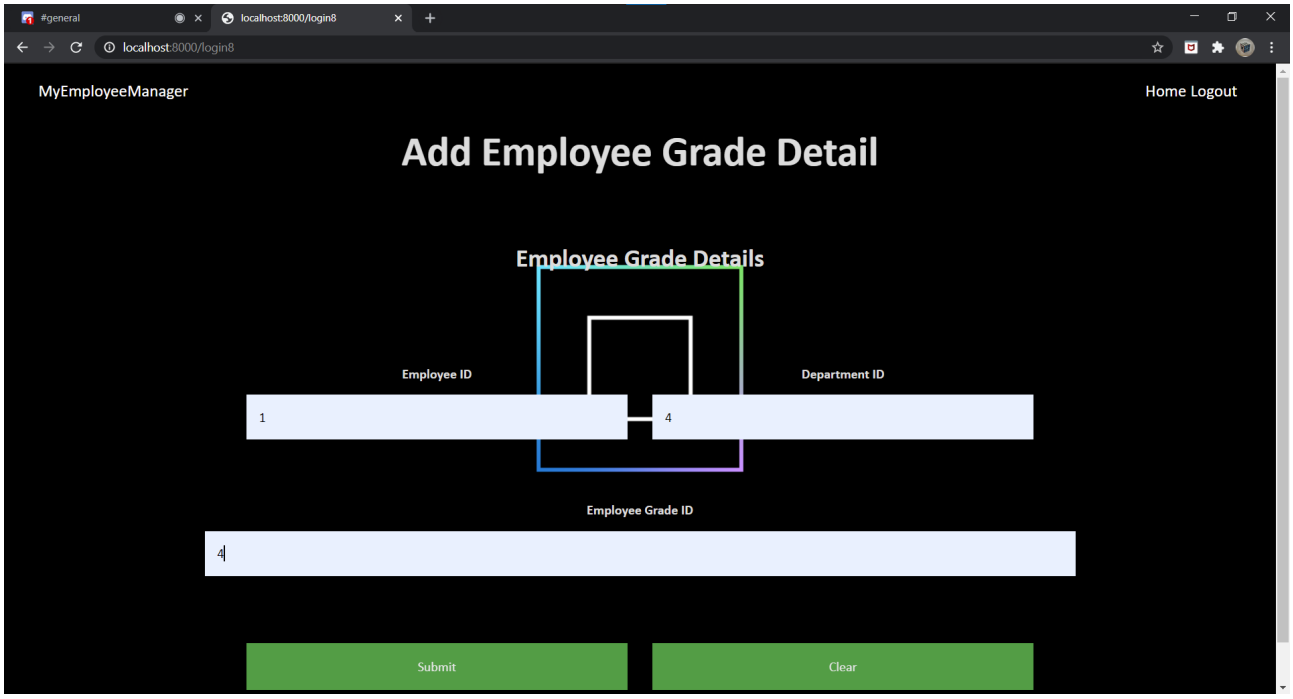

Figure 12: Add Employee Grade Page

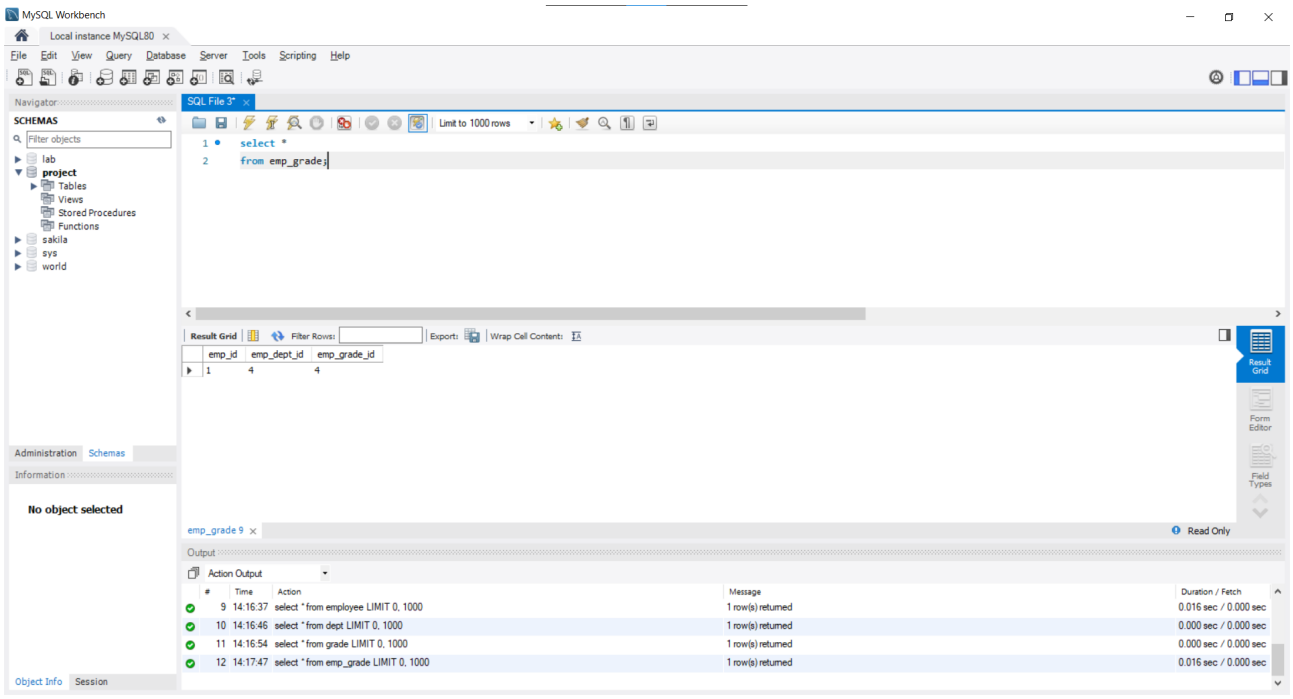

Figure 13: Employee-grade Table

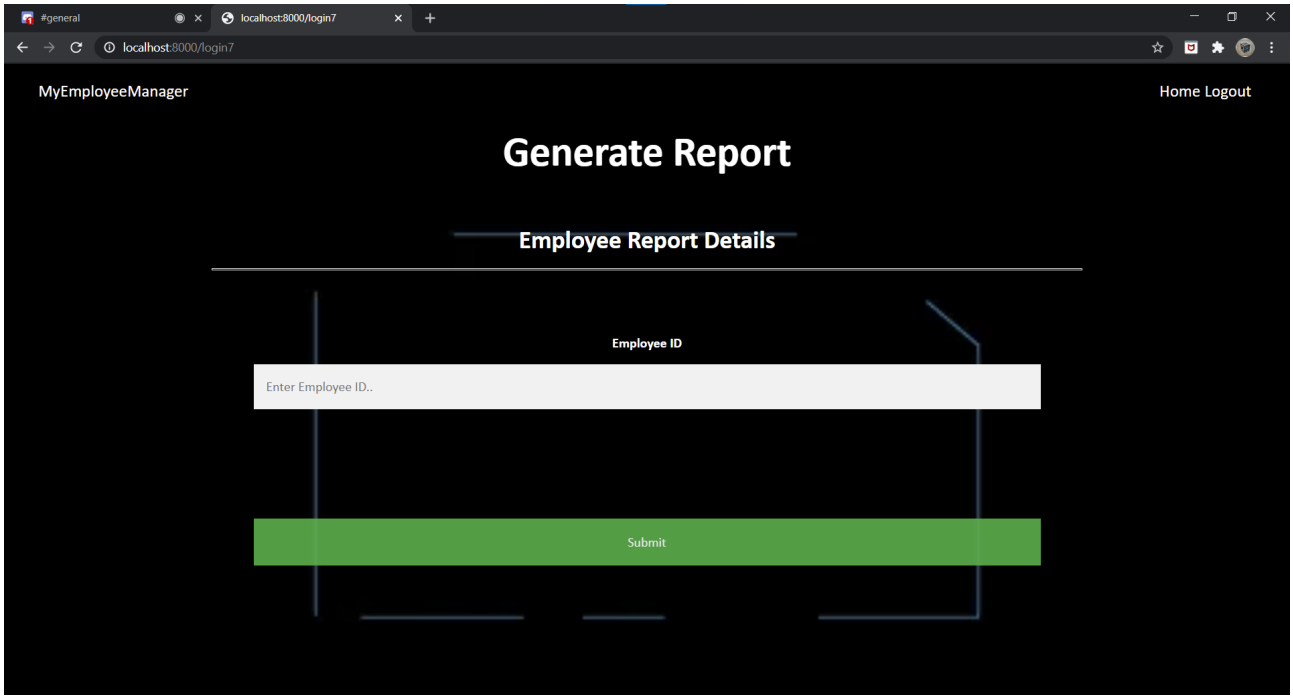

Figure 14: Generate Report Page

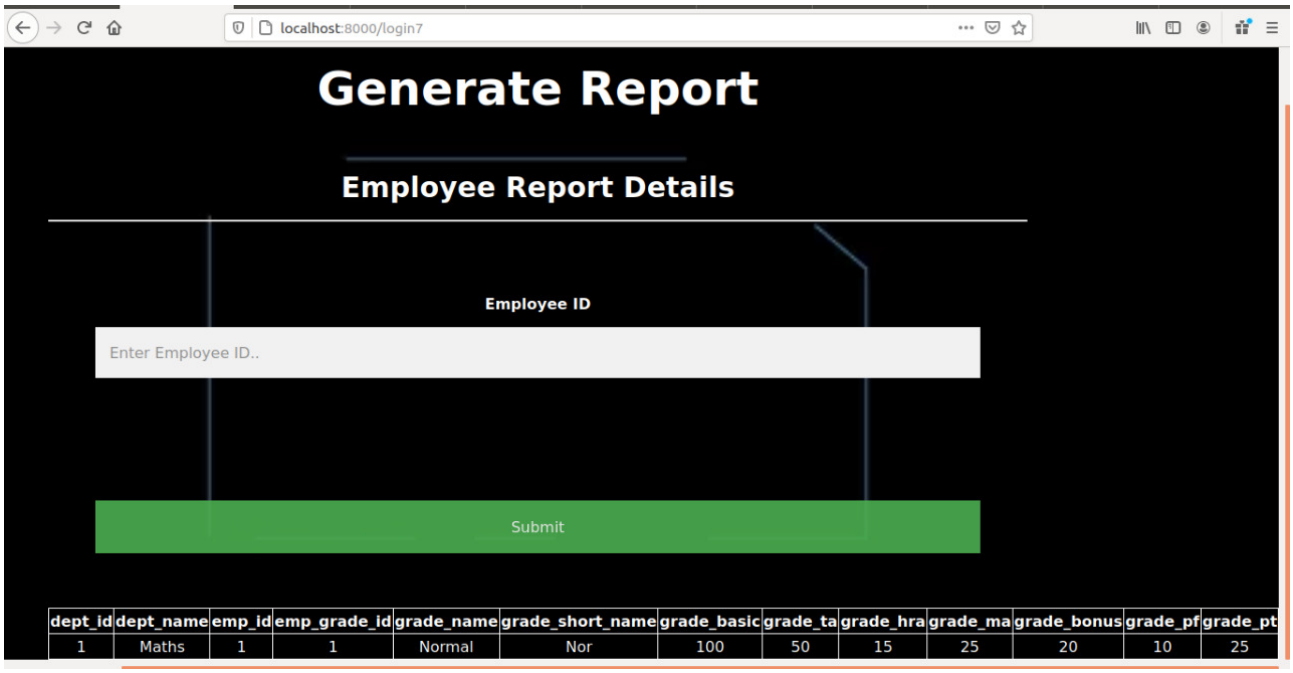

Figure 15: Generate Report After Insertion Page

6 References:

1. https://expressjs.com/

2. https://ejs.co/

3. https://www.w3schools.com/

4. https://www.slideshare.net/ShubhamModi5/payroll-management-system-srs

****** END ******