***Analysis***

**Description of Problem**

I intend to develop a system that logs each visit to the toilet for my school. The system will require pupils to fill out a HTML form with the required information such as their name, email, year and current class. Teachers will also have access to multiple pages where they can view and delete logs, pupils and teachers. Some functions such as viewing all pupils and teachers will only be accessed by admin accounts who will have the ability to also create new teachers.

The end-users of this system will be pupils and teachers, pupils will only be able to login in and submit a leave to the toilet whereas teachers will be able to view all the submit forms and see stats based on the form.

My project meets the advanced higher computing requirements because it will have a suitable HTML form with validated inputs, a database to store the form submissions with MySQL and will extract the required info from the database for teachers to view using SQL queries.

**Scope**

The scope of my project will include:

1. A completed design with pseudocode, data dictionary, query design and wireframes showing the intended interface of the HTML form and the page for teachers to review the submitted forms.
2. A functioning HTML form with validated inputs that can be submitted to a database.
3. A functioning page for teachers to view the submitted leaves in a table.
4. A functioning login page for pupils and teachers.
5. The results of final testing.
6. An evaluation report.

**Constraints**

There are a number of constraints that will apply to this system and development.

1. I will use HTML, CSS, PHP and MySQL to create this system.
2. The system will run on a locally hosted web page with database also being run locally.
3. The submission from the form will be stored in a locally hosted MySQL database.
4. There will be no costs involved in the development of the system as all the software I will be using is free and legally licensed.
5. I will ensure that my project is completed by the deadline of (Due Date) as it will need to be delivered to the SQA for marking.

**Boundaries**

My working system will contain:

1. A system that allows users to sign up with their email and password, forename, surname, house and year group:
   1. The user cannot create an account with an email which has already been used
   2. The password must be a minimum of 8 characters
   3. The password must include capital letters
   4. The password must include lowercase letters
   5. The password must include at least 1 number
   6. The password must contain at least 1 special character
2. A system that allows users to login with their email and password.
3. A form to submit details about their leave to the toilet:
   1. All inputs must be validated
   2. All inputs must be sent to the locally hosted database using MySQL
4. A form to submit details of reported damages.
5. A page for teachers to view data extracted from the database using SQL queries in the form of tables and graphs.
   1. The page must allow teachers to filter the results of the database.

**Requirements Specification**

For this system to be properly functioning it much meet several criteria:

**1. Login System Requirements**

The login page will be the first page users see and will include the following functions:

1. The user will see 2 input boxes to enter the following details:
   1. A valid email
   2. A valid password
2. When the user presses the ‘login’ button the input validation will check that:
   1. Neither of the fields are blank
   2. The email exists within the database
   3. The password matches the one linked to that email
3. If the validation is successful the system will:
   1. Store the email in a session variable
   2. The user will be sent to the designated page – The form or the teacher dashboard
4. If the validation is not successful the system will:
   1. Ask the user to try again

**2. Register System Requirements**

On the pupil login page, the user will see an option to ‘create an account’ after the user clicks it they will be sent to the register page:

1. Send the user to a page to enter the following details:
   1. A unique email
   2. A valid password
   3. Their forename
   4. Their surname
   5. Their house group
   6. Their year group
2. When the user presses the ‘sign up’ button the input validation will check that:
   1. The email does not match an email already in the database
   2. The password matches the following requirements:
      1. The password must be a minimum of 8 characters
      2. The password must include capital letters
      3. The password must include lowercase letters
      4. The password must include at least 1 number
3. If the validation is successful the system will:
   1. Send the user to the login page
4. If the validation is not successful the system will:
   1. Ask the user to try again

This will be the same functions for the page to register a teacher account however, this page will only be accessible for admin account users as the admins will create accounts for teachers. There will only be minor changes in the functions as follows:

1. Send the user to a page to enter the following details:
   1. A unique email
   2. A valid password
   3. Their forename
   4. Their surname
   5. If they are an admin
   6. Their subject

**3. Leave Form Requirements**

1. After the pupil logs in they will be sent to a page to enter their:
   1. Subject
   2. Teacher

This will be done using a select menu with PHP and SQL to dynamically update the options.

1. When the user presses the ‘Submit’ button the system must create a new submission in the ‘Log’ Table of the database with:
   1. A unique LogID
   2. The pupils email
   3. The pupils forename
   4. The pupils surname
   5. The pupils current class
   6. The pupils current teacher
   7. The date and time that the pupil left – This will be retrieved using the NOW() function in SQL

**4. Damage Report Requirements**

1. On the leave form page the user will see an option at the button to submit a damage report.
2. This form will include 2 functions to:
   1. Choose which toilet the damage was in
   2. Details of the damage
3. When the user presses the ‘Submit’ button the system must create a new submission in the ‘Damages’ Table of the database with:
   1. A unique DamageID
   2. The pupils email
   3. The pupils forename
   4. The pupils surname
   5. The date and time that the report was submitted – This will be retrieved using the NOW() function in SQL

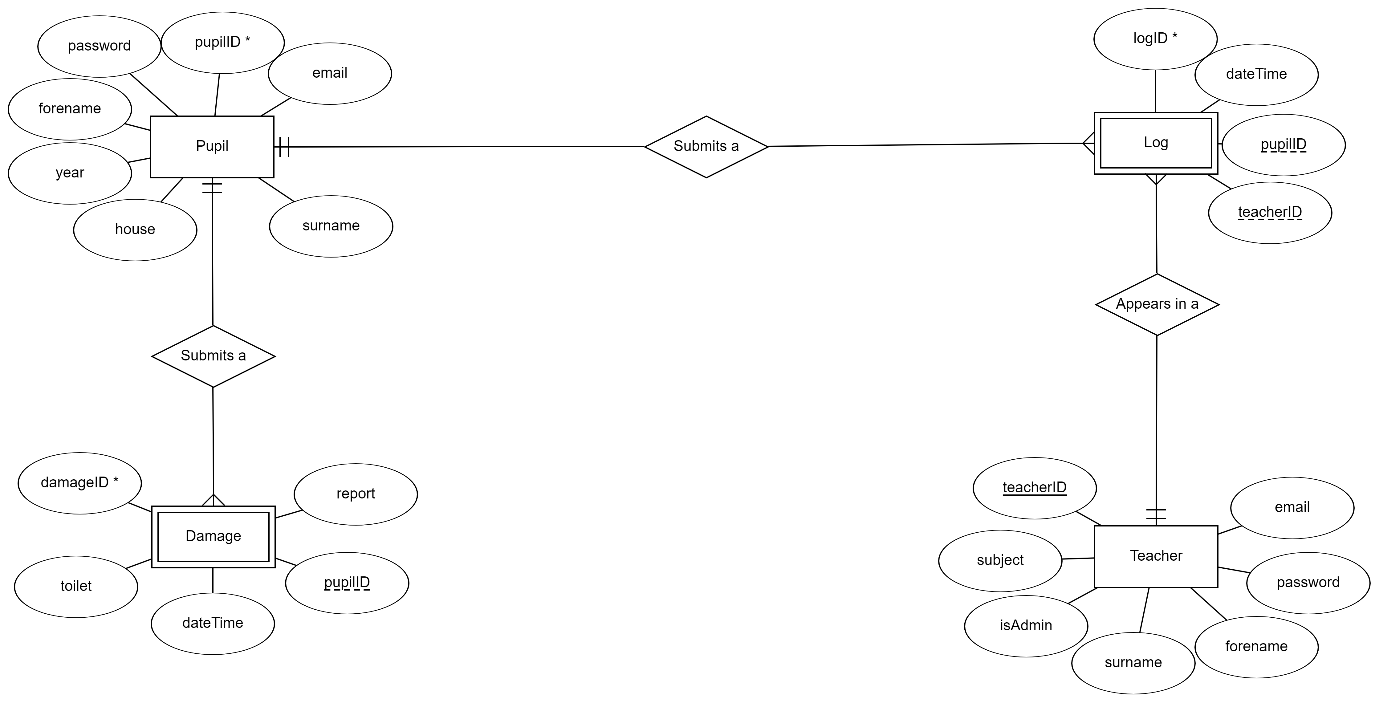
**5. Log Entries Requirements**

1. The system must allow teachers to view all the logs submitted to the data base.

***Design***

**Database Design**

**Entity Relationship Diagram**

****

**Data Dictionary**

**Log Table**

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Key | Extra |
| logID | INT | PK | Auto\_increment |
| dateTime | dateTime |  |  |
| subject | TEXT |  |  |
| pupilID | INT | FK |  |
| teacherID | INT | FK |  |

**Pupil Table**

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Key | Extra |
| pupilID | INT | PK | Auto\_increment |
| email | TEXT |  |  |
| password | TEXT |  |  |
| forename | TEXT |  |  |
| surname | TEXT |  |  |
| house | TEXT |  |  |
| year | TEXT |  |  |

**Teacher Table**

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Key | Extra |
| teacherID | INT | PK | Auto\_increment |
| email | TEXT |  |  |
| password | TEXT |  |  |
| forename | TEXT |  |  |
| surname | TEXT |  |  |
| isAdmin | BOOL |  |  |
| subject | TEXT |  |  |

**Damage Table**

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Key | Extra |
| damageID | INT | PK | Auto\_increment |
| report | TEXT |  |  |
| dateTime | dateTime |  |  |
| toilet | TEXT |  |  |
| pupilID | INT | FK |  |

**Query Design**

**Register Pupil**

|  |  |
| --- | --- |
| **SELECT** | email |
| **FROM** | Pupil |
| **WHERE** | email = POST(email) |

|  |  |
| --- | --- |
| **INSERT** | email, password, forename, surname, house, year |
| **TABLE** | Pupil |
| **VALUES** | POST(email), POST(password), POST(forename), POST(surname), POST(house), POST(year) |

**Pupil Login**

|  |  |
| --- | --- |
| **SELECT** | \* |
| **FROM** | Pupil |
| **WHERE** | email = POST(email) AND password = POST(password) |

**Leave Form**

|  |  |
| --- | --- |
| **SELECT DISTINCT** | subject |
| **FROM** | Teacher |
| **ORDER BY** | subject ASC |

|  |  |
| --- | --- |
| **SELECT** | \* |
| **FROM** | Teacher |
| **ORDER BY** | Subject ASC |

|  |  |
| --- | --- |
| **INSERT** | dateTime, subject, pupilID, teacherID |
| **TABLE** | Log |
| **VALUES** | NOW(), POST(subject), $pupilID, POST(teacher) |

**Damage Form**

|  |  |
| --- | --- |
| **INSERT** | dateTime, toilet, report, pupilID |
| **TABLE** | Damages |
| **VALUES** | NOW(), POST(toilet), POST(report), $pupilID |

**Register Teacher**

|  |  |
| --- | --- |
| **SELECT DISTINCT** | subject |
| **FROM** | Teacher |
| **ORDER BY** | Subject ASC |

|  |  |
| --- | --- |
| **INSERT** | email, password, forename, surname, isAdmin, subject |
| **TABLE** | Teacher |
| **VALUES** | POST(email), POST(password), POST(forename), POST(surname), POST(isAdmin), POST(subject) |

**Teacher Login**

|  |  |
| --- | --- |
| **SELECT** | \* |
| **FROM** | Teacher |
| **WHERE** | email = POST(email) AND password = POST(password) |

**Dashboard**

|  |  |
| --- | --- |
| **SELECT** | logID, dateTime, log.subject, pupil.forename, pupil.surname, teacher.forename AS “TForename”, teacher.surname AS “TSurname”, |
| **FROM** | Log, Teacher, Pupil |
| **WHERE** | pupil.pupilID = log.pupilID AND teacher.teacherID = log.teacherID |

**All Damages**

|  |  |
| --- | --- |
| **SELECT** | damageID, report, damages.pupilID, dateTime, toilet, pupil.forename, pupil.surname |
| **FROM** | Damages |
| **WHERE** | pupil.pupilID = damages.pupilID |

**All Pupils**

|  |  |
| --- | --- |
| **SELECT** | \* |
| **FROM** | Pupil |

**All Teachers**

|  |  |
| --- | --- |
| **SELECT** | \* |
| **FROM** | Teacher |

**Delete Damage**

|  |  |
| --- | --- |
| **DELETE FROM** | Damages |
| **WHERE** | damageID = $damageID |

**Delete Pupil**

|  |  |
| --- | --- |
| **DELETE FROM** | Pupil |
| **WHERE** | pupilID = $pupilID |

**Delete Teacher**

|  |  |
| --- | --- |
| **DELETE FROM** | Teacher |
| **WHERE** | teacherID = $teacherID |

**Delete Log**

|  |  |
| --- | --- |
| **DELETE FROM** | Log |
| **WHERE** | logID = $logID |

**Pseudo Code**

**Connection Query – Used All Pages**

Define servername;

Define username;

Define password;

Define database;

Connect to the database;

If connection fails { display error message };

Start the session;

**Register Pupil**

**Checking if email in use:**

if Query Result returns 1 {

display Error Message;

} else {

SQL Query to insert pupil information;

}

**Pupil Login**

**Authenticating user login:**

If submit is clicked {

$email = User inputted email

$password = User inputted email

SQL Query to check if email and password are in the database and get all pupil data

If Query Result returns 1 {

Start a session variable for email with the email assigned to the login data;

Start a session variable for forename with the forename assigned to the login data;

Start a session variable for surname with the surname assigned to the login data;

Start a session variable for pupilID with the pupilD assigned to the login data;

Redirect user to the form;

}

}

**Leave Form**

**Dynamic subject option list:**

SQL Query to retrieve all subjects from teachers table;

Loop for each subject {

Set option value to current subject value;

Display current subject value as the option name;

}

**Dynamic teacher option list:**

SQL Query to retrieve all data from teachers table;

Loop for each teacher {

Set option value to current teacherID value;

Display current teacher name as the option name;

}

**Teacher Login**

**Authenticating user login:**

If submit is clicked {

$email = User inputted email;

$password = User inputted email;

SQL Query to check if email and password are in the database and get all teacher data;

If Query Result returns 1 {

Start a session variable for email with the value from the email php variable;

Start a session variable for forename with the forename assigned to the login data;

Start a session variable for surname with the surname assigned to the login data;

Start a session variable for teacherID with the teacherlD assigned to the login data;

Start a session variable for isAdmin with the isAdmin value assigned to the login data;

Redirect user to the form;

}

}

**Dashboard**

**Admin nav bar – Used on all pages teachers view:**

If isAdmin session variable is equal to 1 {

Display link to all teachers page;

Display link to all pupils page;

}

**Dynamic Table:**

Loop for all entries in log table {

Display current logID value

Display current dateTime value

Display current subject value

Display current forename value and surname value

Display current teacher forename value and teacher surname value

}

**Table Admin Controls – Used on all pages teachers view:**

If isAdmin session variable is equal to 1 {

Display button to delete selected entry;

}

**All Damages**

**Dynamic Table:**

Loop for all entries in damages table {

Display current damageID value

Display current dateTime value

Display current forename value and surname value

Display current report value

Display current toilet value

}

**All Pupils**

**Dynamic Table:**

Loop for all entries in pupils table {

Display current pupilID value

Display current email value

Display current password value

Display current forename value and surname value

Display current house value

Display current year value

}

**All Pupils**

**Dynamic Table:**

Loop for all entries in teachers table {

Display current pupilID value

Display current email value

Display current password value

Display current forename value and surname value

If isAdmin value returns true {display “Yes”} else {display “No”}

Display current subject value

}

**Delete Damage**

Get damageID passed through the URL

SQL Query to delete entry where damageID = $damageID

Redirect user back to previous page

**Delete Pupil**

Get damageID passed through the URL

SQL Query to delete entry where damageID = $damageID

Redirect user back to previous page

**Delete Teacher**

Get damageID passed through the URL

SQL Query to delete entry where damageID = $damageID

Redirect user back to previous page

**Delete Log**

Get damageID passed through the URL

SQL Query to delete entry where damageID = $damageID

Redirect user back to previous page