**Quality Assurance**

Format for Each Test

**Problem:**

**Potential Issue(s):**

**Full List of Case(s):**

**Solution(s):**

1. **Problem:** Account pages need to be secured. (System Test)

**Potential Issue(s):** An account page could be reached through a direct change of a url.

**Full List of Case(s):**

* A user is signed in as an employee and attempts to alter the url to access a homepage of another user.
* A user is signed in as an administrator and attempts to alter the url to access a homepage of another user.
* A user is signed in as an owner and attempts to alter the url to access a homepage of another user.
* A user is not signed in at all and attempts to alter the url to access a homepage of another user.

**Solution(s):**

**If:**

* A user is signed in as an employee and attempts to alter the url to access a homepage of another user.
* A user is signed in as an administrator and attempts to alter the url to access a homepage of another user.
* A user is signed in as an owner and attempts to alter the url to access a homepage of another user.

**Then:** User is redirected to an error page.

**If:**

* A user is not signed in at all and attempts to alter the url to access a homepage of another user.

**Then:** User is redirected back to the sign-in page.

1. **Problem:** The sign-in process needs to be secured. (Not related to password strength) (System Test)

**Potential Issue(s):**

- Brute Force attacks

- MySQL Injection attacks

- (I’m sure there is more but at the moment I can’t think of any)

**Full List of Case(s):**

* Someone is attempting to use a script/program to enact a brute force attack on the sign-in page.
* Someone is attempting a MySQL injection by structuring either their password or username to simulate a query to the database.

**Solution(s):**

**If:**

* Someone is attempting to use a script/program to enact a brute force attack on the sign-in page.

**Then:**

* The sign-in page should enforce a rate-limit system, as well as always giving a standard message on incorrect login information; e.g. “Your username/password is incorrect”

**If:**

* Someone is attempting a MySQL injection by structuring either their password or username to simulate a query to the database.

**Then:**

* The only allowed queries to the database during the login/registration process should be a set of prepared statements and parameterized queries that the database server parses separately from any other parameters.

1. **Problem:** User information needs to be reasonably unique to each specific user. (System Test)

**Potential Issue(s):** A user could specify information that already exists in the database for another user.

**Full List of Case(s):**

* A user decides on a username that is already in the database.

**Solution(s):**

**If:**

* A user decides on a username that is already in the database.

**Then:**

* An error message is displayed to the user and the user is asked to create a different username.

1. **Problem:** Keeping the permissions between the three ranks separated. (System Test)

**Potential Issue(s):** How would we keep a User from being granted admin privileges or owner privileges?

**Full List of Case(s):**

* A user is granted administrator privileges or owner privileges.

**Solution(s):**

**If:**

* A user is granted administrator privileges or owner privileges.

**Then:**

* There should not be an option for an administrator to grant privileges to a regular user account, this should be handled through database interactions by the owner.

1. **Problem:** Keeping information in the database secured. (System Test)

**Potential Issue(s):** Does MySQL have security built in to prevent this aside from user authentication, or do we have to put in some prevention methods ourselves?

**Full List of Case(s):**

* Someone attempts to access the information in the database as a regular User account (not admin or owner).

**Solution(s):**

**If:**

* Someone attempts to access the information in the database as a regular User account (not admin or owner).

**Then:**

1. **Problem:** Need to make sure that confirmation emails are sent successfully. (System Test)

**Potential Issue(s):** A confirmation email could be sent, but not received due to any of the possible reasons; this results in a halt of the workflow process.

**Full List of Case(s):**

* A timesheet submission confirmation email has not been received.
* A invoice confirmation email has not been received by the vendor.
* A paystub confirmation email has not been received.

**Solution(s):**

**If:**

* A timesheet submission confirmation email has not been received.
* A paystub confirmation email has not been received.

**Then:**

* An alert should be sent to both the sender and the receiver; either by email or by a notification on the website, so that a solution can be reached.

**If:**

* A invoice confirmation email has not been received by the vendor.

**Then:**

* An alert/notification will be sent to the admin who initiated the invoice confirmation as well as the owner of the website, so that the issue can be resolved as quickly as possible.

1. **Problem:** Each Link/Button needs to lead to the correct page. (User Test)

**Potential Issue(s):** An easy oversight to make would be to have an incorrect path for a link or button.

**Full List of Case(s):**

* A button/link does not lead to the correct page.

**Solution(s):**

**If:**

* A button/link does not lead to the correct page.

**Then:**

* This error needs to be reported by a user/tester.

1. **Problem:** Values entered into a field by the user need to be the correct data-type so that the database can store them.

**Potential Issue(s):** A user could input a random value that does not match a connected attribute in the database. (System Test)

**Full List of Case(s):**

* The user enters in a data-type not accepted by the database attribute.

**Solution(s):**

**If:**

* The user enters in a data-type not accepted by the database attribute.

**Then:**

* The resulting database error triggers a refresh of the current page with the field in question highlighted so as to alert the user to their error.

1. **Problem:** Need to make sure that all of the parts/tools of the project are correctly communicating with each other.

**Potential Issue(s):** Something could go wrong so that one of the servers could not contact another server, rendering the project broken.

**Full List of Case(s):**

* The database server can no longer connect to the web server.

**Solution(s):**

**If:**

* The database server can no longer connect to the web server.

**Then:**

* An error messages needs to be displayed to admins and the owner detailing what is going wrong, and a general maintenance error displayed to users.

1. **Problem:**

**Potential Issue(s):**

**Full List of Case(s):**

**Solution(s):**

**If:**

**Then:**

1. **Problem:**

**Potential Issue(s):**

**Full List of Case(s):**

**Solution(s):**

**If:**

**Then:**

1. **Problem:**

**Potential Issue(s):**

**Full List of Case(s):**

**Solution(s):**

**If:**

**Then:**

1. **Problem:**

**Potential Issue(s):**

**Full List of Case(s):**

**Solution(s):**

**If:**

**Then:**

1. **Problem:**

**Potential Issue(s):**

**Full List of Case(s):**

**Solution(s):**

**If:**

**Then:**

1. **Problem:**

**Potential Issue(s):**

**Full List of Case(s):**

**Solution(s):**

**If:**

**Then:**

1. **Problem:**

**Potential Issue(s):**

**Full List of Case(s):**

**Solution(s):**

**If:**

**Then:**

1. **Problem:**

**Potential Issue(s):**

**Full List of Case(s):**

**Solution(s):**

**If:**

**Then:**

1. **Problem:**

**Potential Issue(s):**

**Full List of Case(s):**

**Solution(s):**

**If:**

**Then:**

1. **Problem:**

**Potential Issue(s):**

**Full List of Case(s):**

**Solution(s):**

**If:**

**Then:**

1. **Problem:**

**Potential Issue(s):**

**Full List of Case(s):**

**Solution(s):**

**If:**

**Then:**

1. **X**
2. **X**
3. **X**
4. **X**
5. **X**
6. **X**
7. **X**
8. **X**
9. **X**