NAME: Christofer B. Baldano
COURSE YEAR, & SECTION: BSIS 2A
SUBJECT: Web System
PROFESOR: Reymark A. Llagas

SCENARIO 1: Using $_POST instead of $_GET
Problem: Undefined index if no POST request.
Solution: CHANGE THE $_POST INTO $_GET
Explanation: Magkakaroon ng error ang script dahil ang id na parameter ay nagmula sa URL. Dahil dito, mas better gamitin ang $_GET, mali ang pangalan ng column; dapat yung "student_id" upang umayon sa database.

SCENARIO 2: Missing quotes in SQL when using POST
Problem: Unknown column 'Ana' SQL error..
Solution: Variable should be quoted
Explanation: Ang mga string na halaga ay dapat nakapaloob sa mga quotes, kung hindi ay iisipin ng SQL na ito ay pangalan ng column, kaya't magkakaroon ng "unknown column" error.

SCENARIO 3: SQL injection vulnerability
Problem: Users can input 1 OR 1=1 and retrieve all records
Solution: Use prepared statements
Explanation: I think mas better gumamit ng prepared statements dahil minimake-sure nito na ang input ay correct bago patakbuhin ang query, kaya't naiiwasan ang SQL injection.

SCENARIO 4: Forgetting to validate empty post field
Problem: If form is empty → blank rows inserted or SQL errors.
Solution: Validate input before inserting
Explanation: Kung may walang laman o walang data na isinubmit, maaaring mag result ito ng mga error o magulong data sa database, kaya't mahalaga na ito ay validated muna.

SCENARIO 5: Wrong key name in POST
Problem: Undefined index: emial
Solution: Check and correct the variable name
Explanation: Napaka-strikto ng PHP sa pagbaybay at casing ng mga keys; kung mali ang POST key, hindi makukuha ng PHP ang halaga at ipapakita nito ang "undefined index" error.

SCENARIO 6: Unsafe direct use of GET in DELETE `
Problem: User can delete everything using ?id=0 OR 1=1.
Solution: ID should be integer
Explanation: Kung gagamitin mo lamang ang raw GET values, possible i abuse ito ng mga user para i remove o baguhin ang mga records. Mas ligtas na i-limit ito sa mga integers lamang.

SCENARIO 7: Query fails but script continues

Problem: Missing quotes around email → SQL error, but code still prints "Updated!"

Solution: Add ng quotes sa email

Explanation: Always check the query if nag succeed ba para maayos yung report at para maiwasan ang error

SCENARIO 8: Missing mysqli_fetch_assoc loop

Problem: The first record is the only one that prints.

Solution: Loop through the results

Explanation: Kung walang loop, isang row lamang ang kinukuha kahit marami sa database. kaya dapat gamitin ang loop para mag run lahat ng records

SCENARIO 9: Using GET but link send POST

Problem: Undefined index because link does not send POST.

Solution: Change to GET

Explanation: Dahil nagsesend ang link ng isang ID sa through URL, dapat mong gamitin ang $_GET upang ang pag-uugali ng script ay mag match at ang mga error ay maiiwasan.

SCENARIO 10: Wrong variable used in SQL

Problem: Undefined variable $aeg

Solution: Just correct the variable name

Explanation: Madali lang sanang i fix if may typographical error na isa pa sa variable o key, kaya need i-retrieve lahat ng na-store. Always namang i-double check ang spelling at casing.

SCENARIO 11: Mismatched method (expets POST but form sends GET)

Problem: Undefined index: email

Solution: The methods should match

Explanation: Di nag conenct ng maayos ang method so nag result yun ng error, example sa left is POST ng form at sa right is GET ng script.

SCENARIO 12: Numeric GET used inside quotes

Problem: ID should not be numeric

Solution: Remove the quotes

Explanation: Kung ang ID ay number, kaya di na kailangan ng quotes, pero if sa string values ay dapat na may quotes para ma-interpret ng SQL ng tama.

SCENARIO 13: Missing WHERE clause in UPDATE

Problem: Updates ALL rows.

Solution: Add WHERE clause

Explanation: Kapag mayo ki WHERE clause sa UPDATE, affected lahat ng rows ng table, kaya dapat always i-double check.

SCENARIO 14: Using POST array incorrectly

Problem: Undefined index / missing quotes.

Solution: Ensure that correct and proper syntax is used.

Explanation: Typo, kaipuhan ki quotes ng array and nakaattach yung brackets ng maayos

SCENARIO 15: Get parameter used inside SQL without sanitation

Problem: User can do ?page=1000000000 and crash MySQL

Solution: Validate and limit the page numbers

Explanation: If super laki po ng page number, pwedeng bumagal o mag crash ng database, kaya need i validate and i limit tabi before mag query