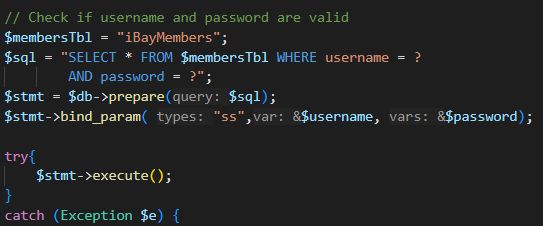
# Backend – Tom

### Each file has

* Error messages for:
  + Wrong get/post
  + Called from the correct location
  + User not being logged in
  + Sql query issues
  + Empty inputs that are essential
* SQL injection protection when $\_POST is used for sql querys
* Use commonly repeated code, connect.php

### Examples of these

* All files where user information is used have SQL injection checking, this is done using prepare.

code- 

* All pages that should check that the user is logged in before allowing any code to run

code- A computer screen with colorful text

AI-generated content may be incorrect.

* Error checking if the result of an SQL query is null/blank.

code- A computer screen shot of a black background

AI-generated content may be incorrect.

* Errors from user input pass a popup back.

code- A screen shot of a computer screen

AI-generated content may be incorrect.

* Check location that the file is called from to ensure it is being used as designed (kradle)

Code- A screen shot of a computer

AI-generated content may be incorrect.

* Php files restricted to post & get methods separately dependent on the files function.

Code- A computer screen with white text

AI-generated content may be incorrect.

* ../../connect.php used code- 

## Connect.php

Sets up Access to the db, used across multiple files.

## Login

Nothing fancy to note

starts session adding userId & passes onto buyer’s page

incorrect username or password checker is implemented

## Register

Nothing fancy to note

Adds user to db.

## Sellers Create Item

Takes inputs from the form in the form of $\_POST variable, finds a new item id by iterating through adding 1 each time and finding the smallest possible unique value. Then using prepare() adds the item to the db.

## Sellers View items

Loops through each item with the userID equal to that found in the $\_SESSION variable. Then loops through each items image encoding the stored binary data using 

Base64\_encode. This allows us to use json\_encode($items) later. One by one we add all of the images to $images for a given item and then add a new entry to items containing the image.

## Sellers Update item

Updates the item in the db as expected using the sql injection protection techniques stated previously

## Sellers Delete item

Nothing to note. Uses listed techniques

## Buyers View Items

Verry similar to seller’s view items. However instead of sorting items by userId we sort them by a given category. This allows us to load the page one category at a time. With the use of AJAX…..not my bit…