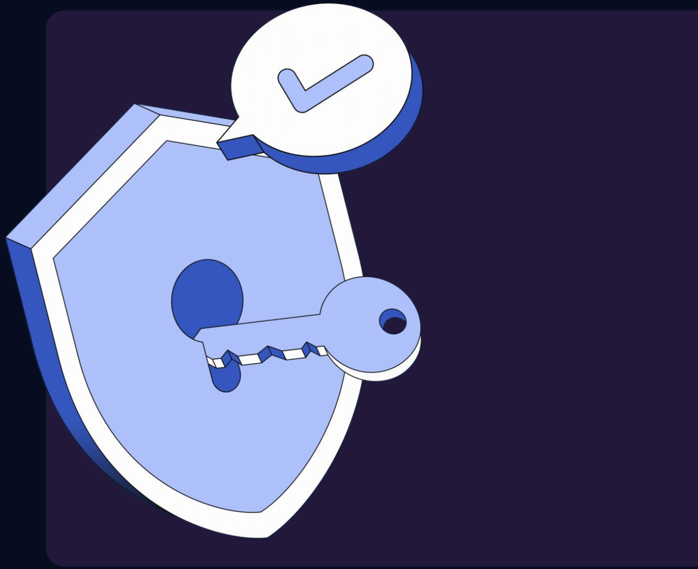
Secure Coding:

Code With Security In Mind

Session 1

Presented by: Hassan Al Achek



Who am i?

Hassan Al Achek

- Red Teamer
- CyberSecurity Researcher
- Offensive Cyber Security Engineer, Passionate about the inner workings of computers ;)
- Focused on web security, network security, reverse engineering, and malware analysis
- Attracted by state-sponsored threat actors 😉
- Moderator of Lebanon's largest cybersecurity community
- Instructor and penetration tester at Semicolon Academy
- Former Red Teamer at Covéa
- Ingénieur Civil des Mines (ICM) - École des Mines de Saint-Étienne
- Electrical and Telecommunications Engineer - Lebanese University

Agenda

Agenda

01 What is Vulnerability?

What is Vulnerability?

❖ Vulnerabilities are weaknesses or gaps that threats can exploit to cause harm.

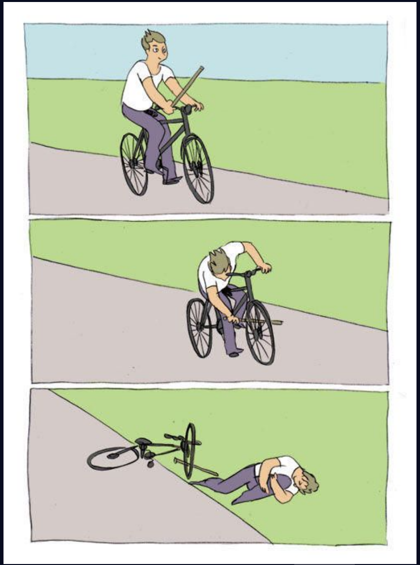
❖ These vulnerabilities might involve a particular operating system or application, the physical location of your office building, or other similar factors.

How Vulnerability Occurs?

- ❖ A vulnerability occurs when an application mishandles user-crafted input, causing the application to behave in an unexpected manner.

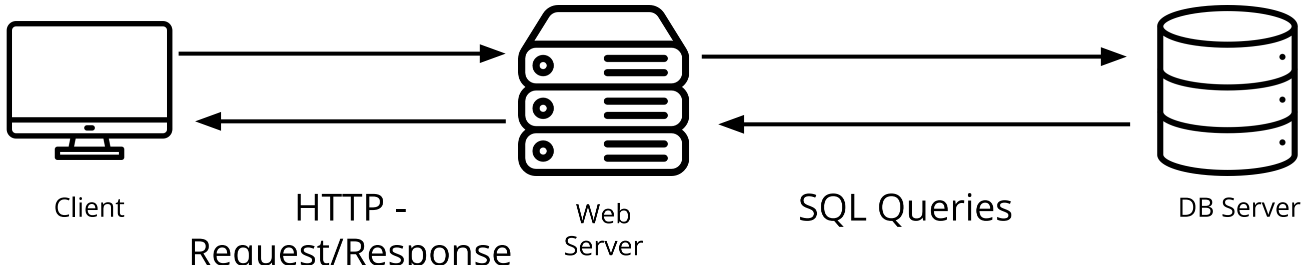
- ❖ Imagine you're riding a bicycle, and someone unexpectedly jams a stick into the spokes of your wheel.

  - ➢ **The vulnerability** here is that the wheel is not designed to handle a foreign object being forced into it.
  - ➢ **The exploit** is the stick, deliberately used to take advantage of this weakness.
  - ➢ **The impact** is that you lose control of the bicycle and fall, potentially injuring yourself, such as breaking an arm or a leg.



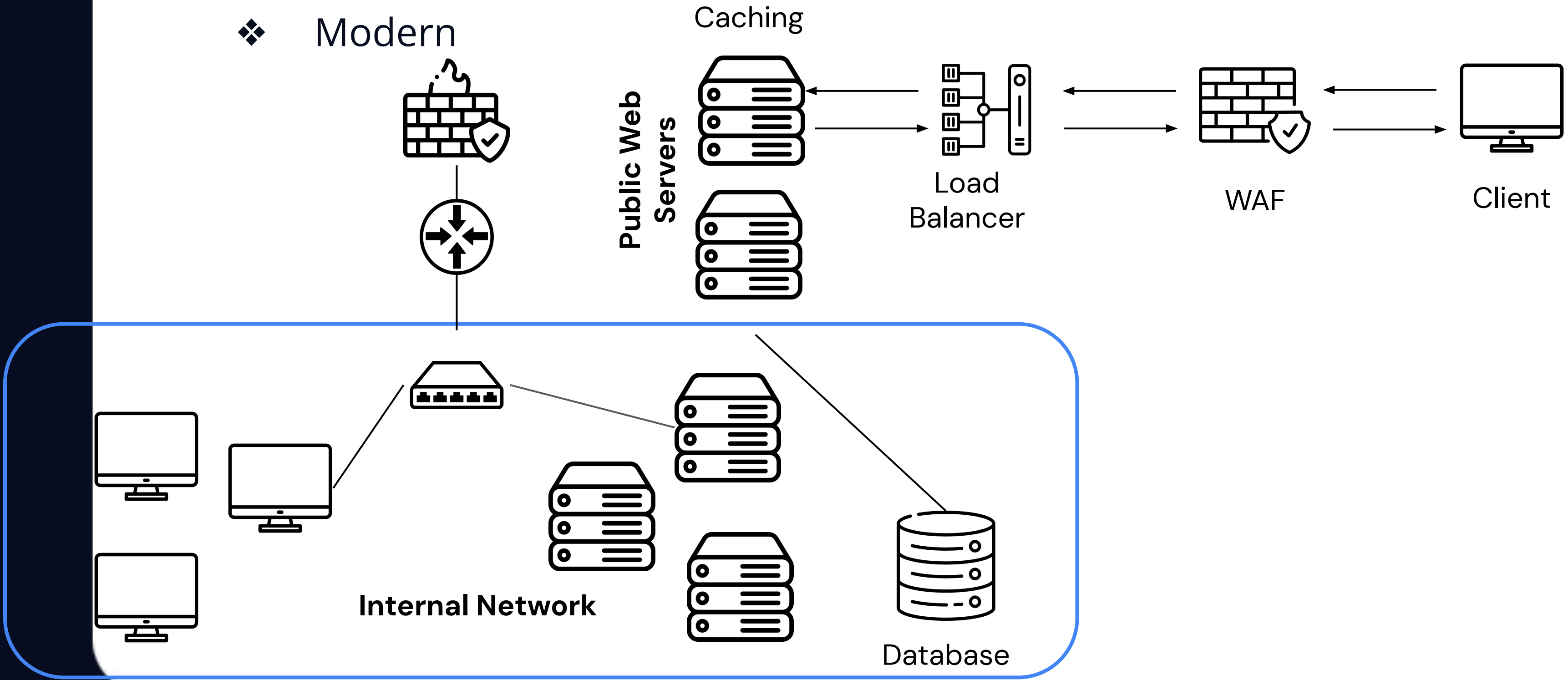
Web Application Overview

- Classic



Web Application Overview

- Modern



Simple Concept |From The Source to The Sink

Simple Concept | From The Source to The Sink

Sink

A function to process user data and display it somewhere.

A function to log user actions.

A function to execute a database query based on user input.

A function to execute a command based on user input.

A backend procedure call.

A client side function.

A function to retrieve a remote or local file.

…

Source

User Input Form

HTTP Headers

HTTP Query Parameters

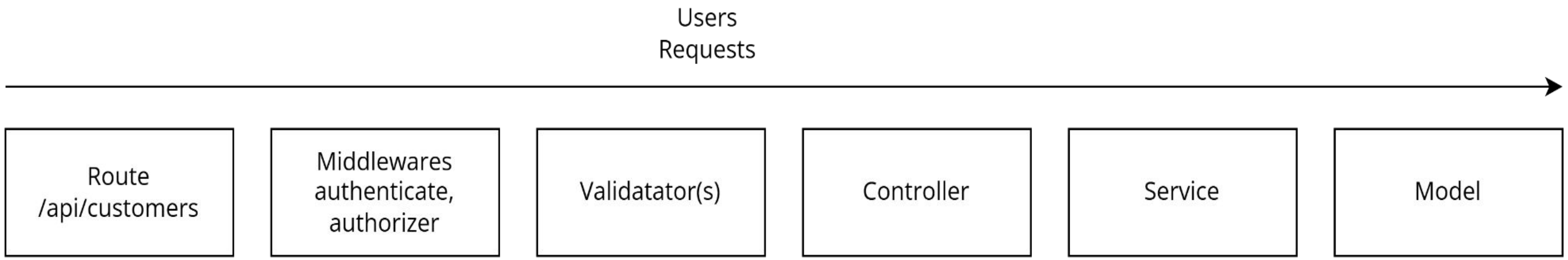
HTTP Body Parameters

Endpoint Data Parsing

Client-Side Storage

…

Simple Concept |From The Source to The Sink

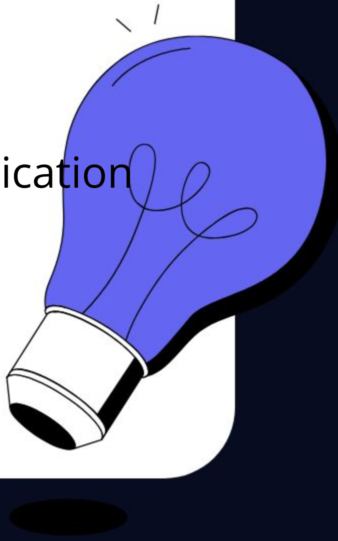


02 OWASP

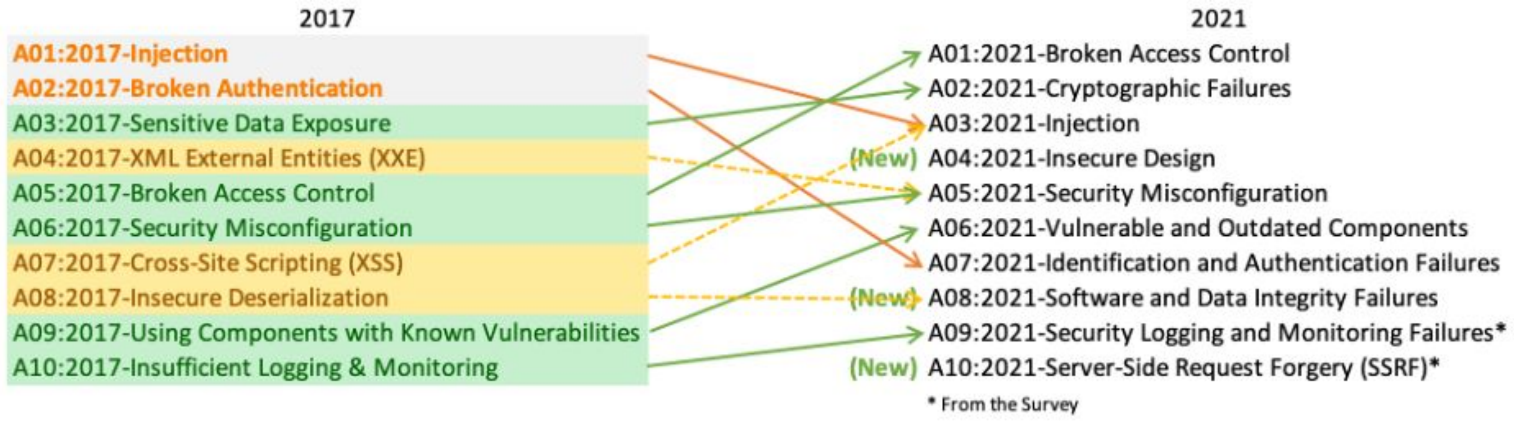
OWASP

- ❖ The Open Worldwide Application Security Project (OWASP) is a nonprofit organization focused on enhancing software security. It follows an **open community** model, encouraging anyone to join, engage in discussions, and contribute to OWASP projects and initiatives.

- ❖ The OWASP Top 10 offers a ranked list of the most critical web application security risks, along with guidance on how to address them.



OWASP



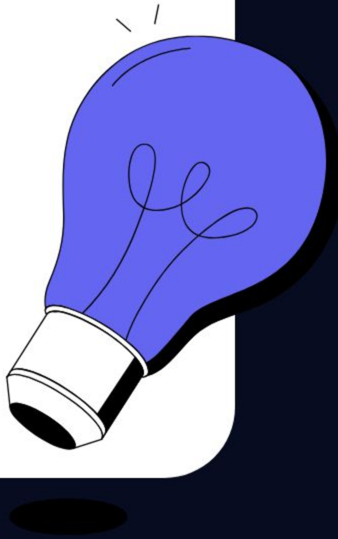
2017	2021
A01:2017-Injection	A01:2021-Broken Access Control
A02:2017-Broken Authentication	A02:2021-Cryptographic Failures
A03:2017-Sensitive Data Exposure	A03:2021-Injection
A04:2017-XML External Entities (XXE)	(New) A04:2021-Insecure Design
A05:2017-Broken Access Control	A05:2021-Security Misconfiguration
A06:2017-Security Misconfiguration	A06:2021-Vulnerable and Outdated Components
A07:2017-Cross-Site Scripting (XSS)	A07:2021-Identification and Authentication Failures
A08:2017-Insecure Deserialization	(New) A08:2021-Software and Data Integrity Failures
A09:2017-Using Components with Known Vulnerabilities	A09:2021-Security Logging and Monitoring Failures*
A10:2017-Insufficient Logging & Monitoring	(New) A10:2021-Server-Side Request Forgery (SSRF)*

* From the Survey

03 Business Logic Vulnerabilities

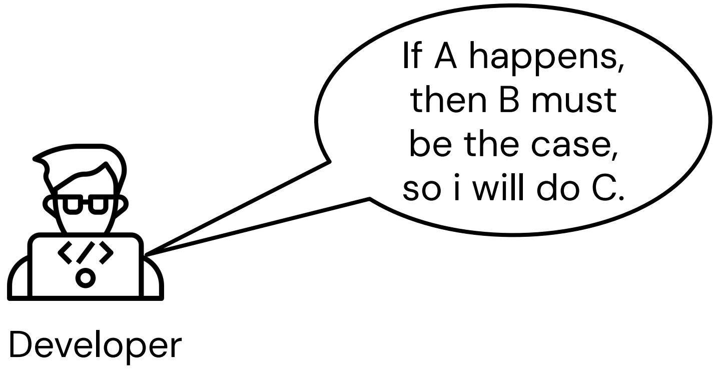
Business Logic Vulnerabilities

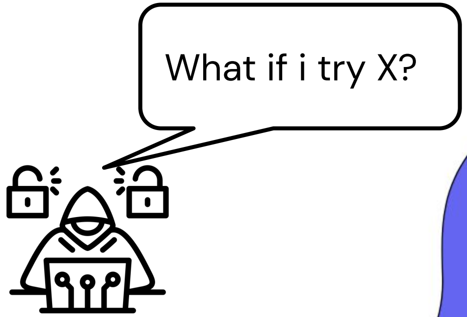
- ❖ The logic implemented within the application is defective in some way.
- ❖ The assumption that the designer or developer made is the first cause of this defect.

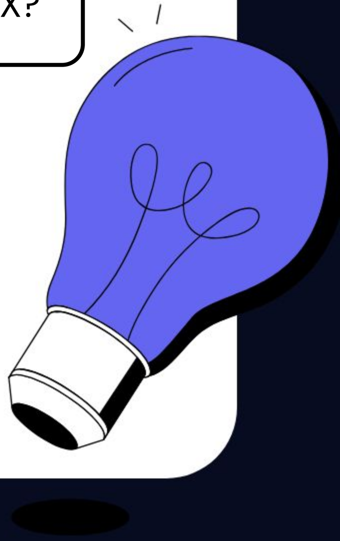


Business Logic Vulnerabilities | Continued

- The developer may have reasoned something like:







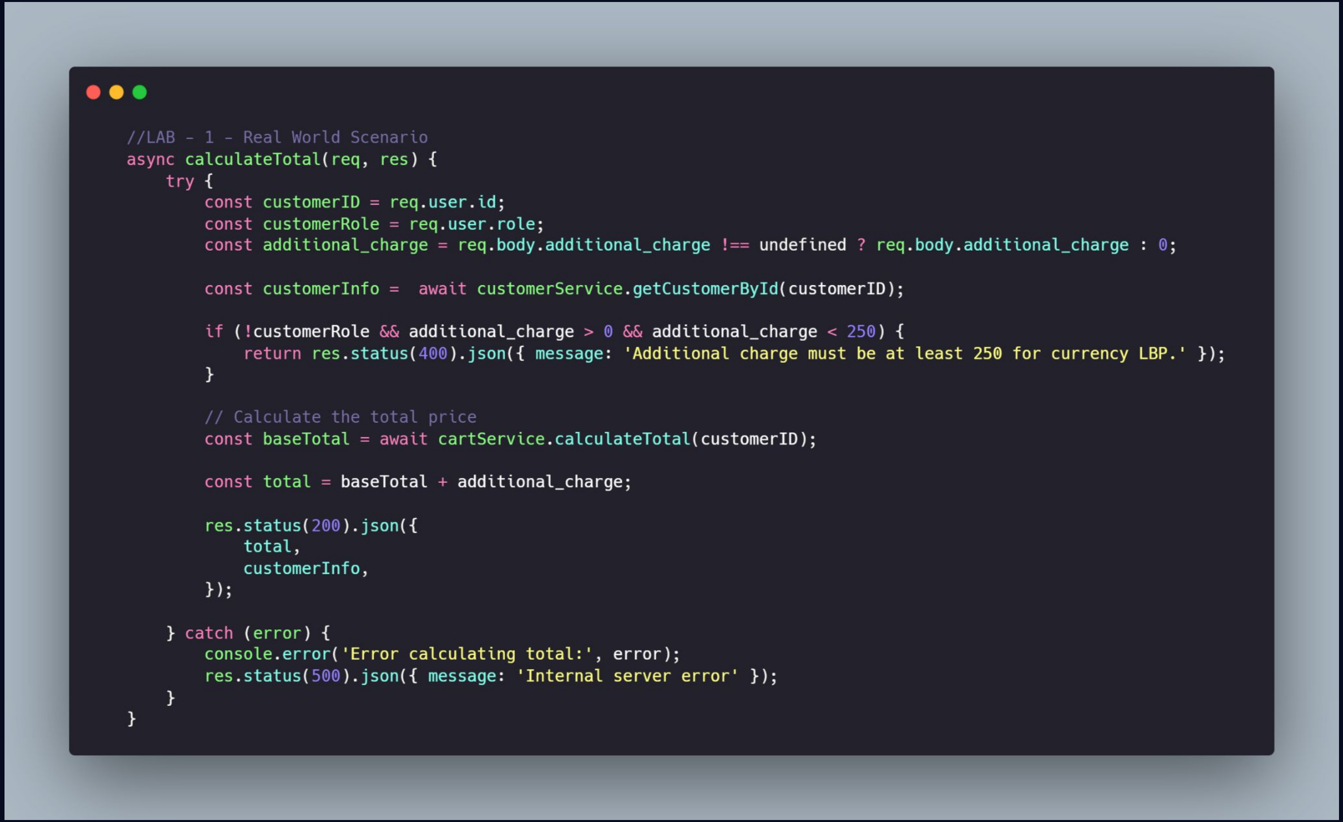
- The developer failed to consider a scenario that violates the assumption.

Business Logic Vulnerabilities |Continued

- The logic implemented within the application is defective in some way.

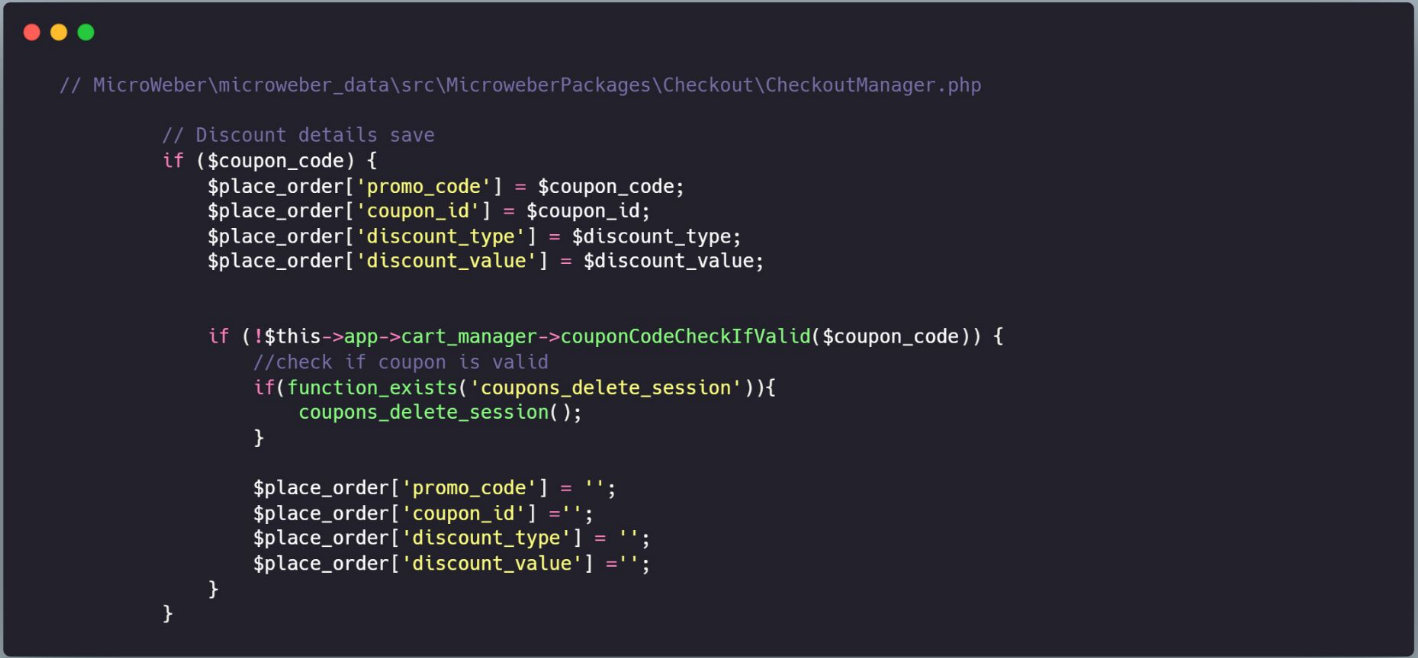
- The assumption that the designer or developer made is the first cause of this defect.

Scenario #1 - Real World Scenario



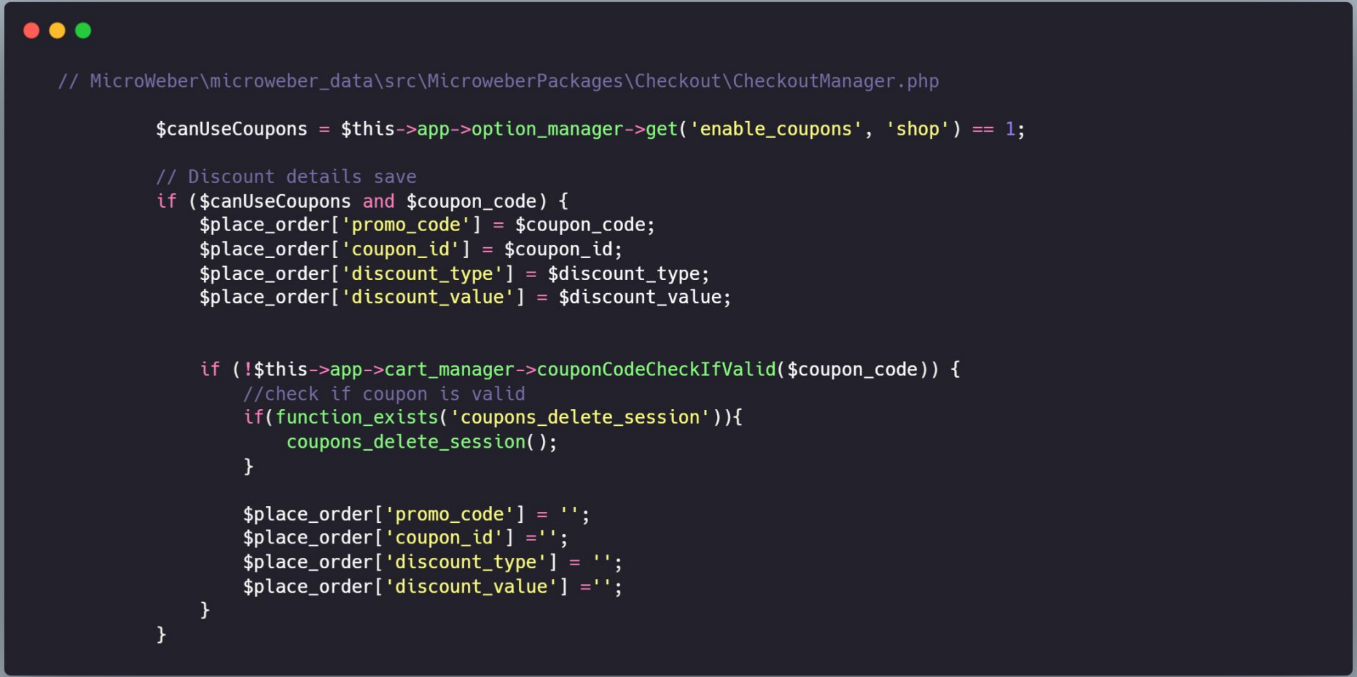
```
//LAB - 1 - Real World Scenario
async calculateTotal(req, res) {
    try {
        const customerID = req.user.id;
        const customerRole = req.user.role;
        const additional_charge = req.body.additional_charge !== undefined ? req.body.additional_charge : 0;

        const customerInfo =  await customerService.getCustomerById(customerID);

        if (!customerRole && additional_charge > 0 && additional_charge < 250) {
            return res.status(400).json({ message: 'Additional charge must be at least 250 for currency LBP.' });
        }

        // Calculate the total price
        const baseTotal = await cartService.calculateTotal(customerID);

        const total = baseTotal + additional_charge;

        res.status(200).json({
            total,
            customerInfo,
        });

    } catch (error) {
        console.error('Error calculating total:', error);
        res.status(500).json({ message: 'Internal server error' });
    }
}
```

Scenario #2 - Microweber | CVE-2023-6832



```php
// MicroWeber\microweber_data\src\MicroweberPackages\Checkout\CheckoutManager.php

        // Discount details save
        if ($coupon_code) {
            $place_order['promo_code'] = $coupon_code;
            $place_order['coupon_id'] = $coupon_id;
            $place_order['discount_type'] = $discount_type;
            $place_order['discount_value'] = $discount_value;


            if (!$this->app->cart_manager->couponCodeCheckIfValid($coupon_code)) {
                //check if coupon is valid
                if(function_exists('coupons_delete_session')){
                    coupons_delete_session();
                }

                $place_order['promo_code'] = '';
                $place_order['coupon_id'] ='';
                $place_order['discount_type'] = '';
                $place_order['discount_value'] ='';
            }
        }
```

Scenario #2 - Microweber | CVE-2023-6832

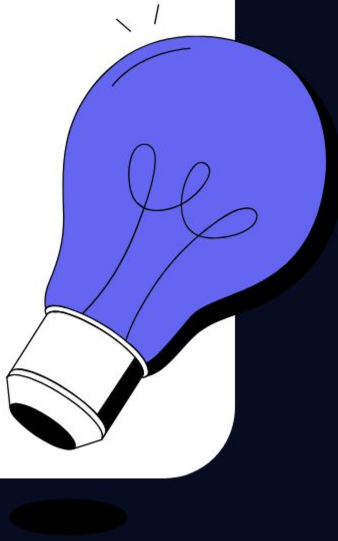


```php
// MicroWeber\microweber_data\src\MicroweberPackages\Checkout\CheckoutManager.php

         $canUseCoupons = $this->app->option_manager->get('enable_coupons', 'shop') == 1;

         // Discount details save
         if ($canUseCoupons and $coupon_code) {
             $place_order['promo_code'] = $coupon_code;
             $place_order['coupon_id'] = $coupon_id;
             $place_order['discount_type'] = $discount_type;
             $place_order['discount_value'] = $discount_value;


             if (!$this->app->cart_manager->couponCodeCheckIfValid($coupon_code)) {
                 //check if coupon is valid
                 if(function_exists('coupons_delete_session')){
                     coupons_delete_session();
                 }

                 $place_order['promo_code'] = '';
                 $place_order['coupon_id'] ='';
                 $place_order['discount_type'] = '';
                 $place_order['discount_value'] ='';
             }
         }
```

04 Insecure Direct Object Reference (IDOR)

Insecure Direct Object Reference (IDOR)

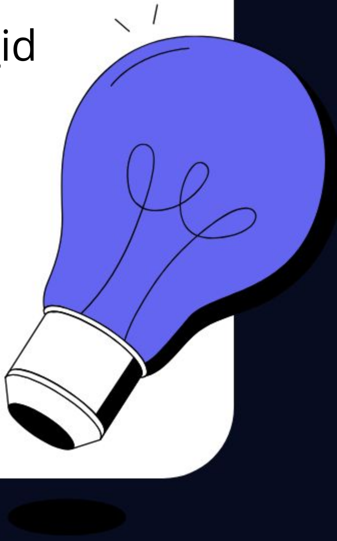
- ❖ IDOR are a type of access control vulnerability that arises when an application uses user-supplied input to access objects directly.
- ❖ An example of IDOR vulnerability with direct access to database objects
- ❖ https://workshop.com/api/user_profile?id=132355



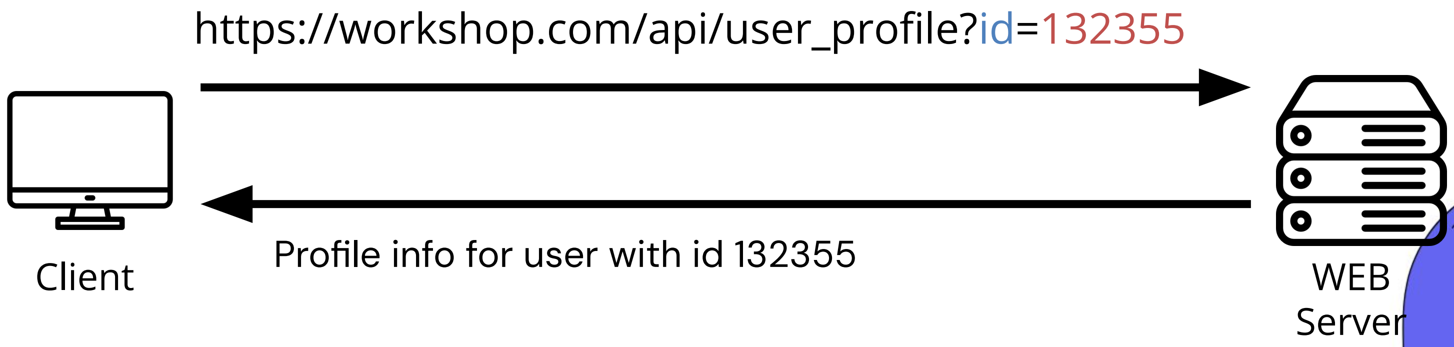
Insecure Direct Object Reference (IDOR) |Continued

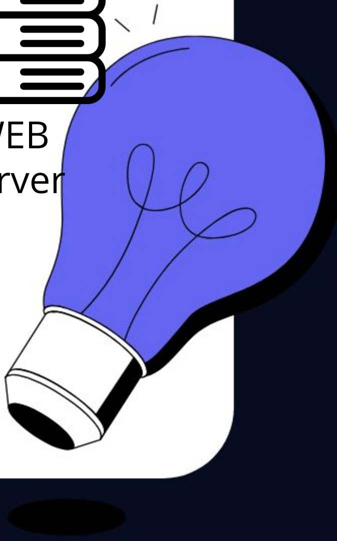
- The profile id is used directly to perform queries on the back-end database.

- In absence of access controls, an attacker can modify the profile_id value to view the records of other customers.

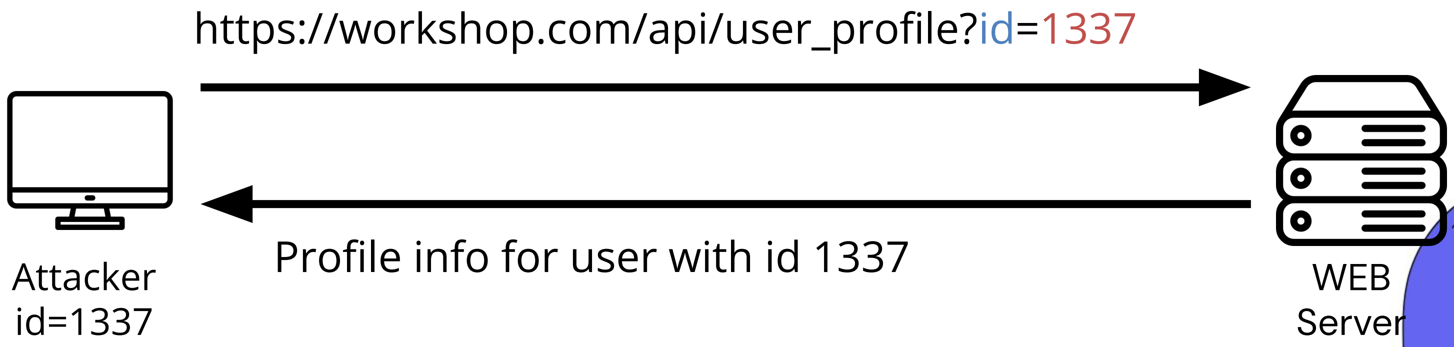


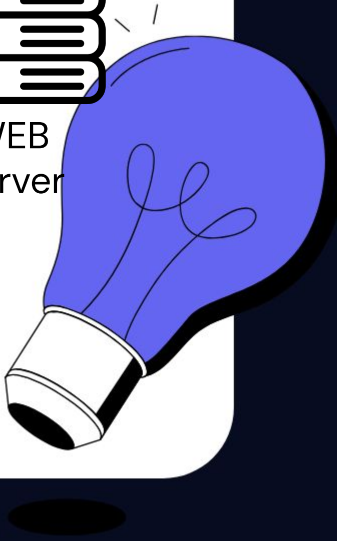
Insecure Direct Object Reference (IDOR) |Continued


https://workshop.com/api/user_profile?id=132355
Client
Profile info for user with id 132355
WEB Server

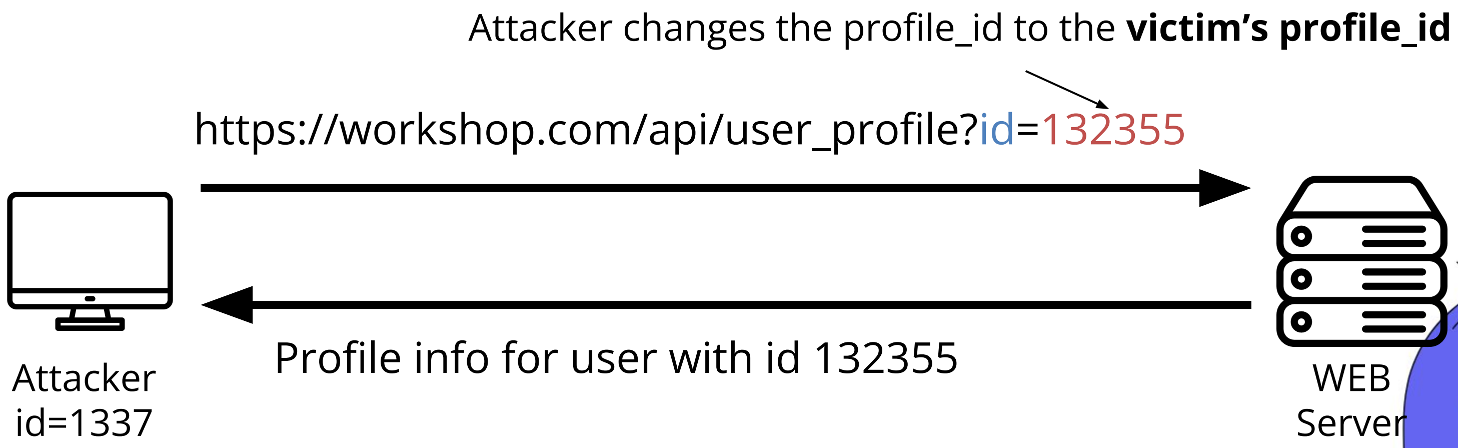


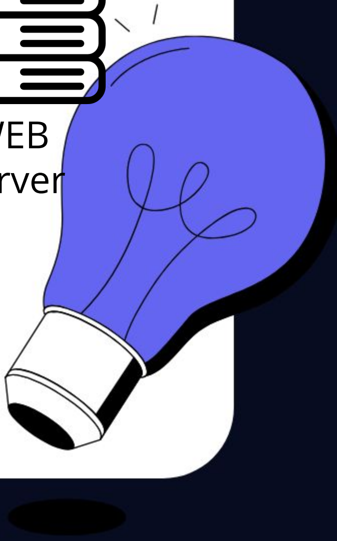
Insecure Direct Object Reference (IDOR) |Continued


https://workshop.com/api/user_profile?id=1337
Profile info for user with id 1337
Attacker
id=1337
WEB
Server

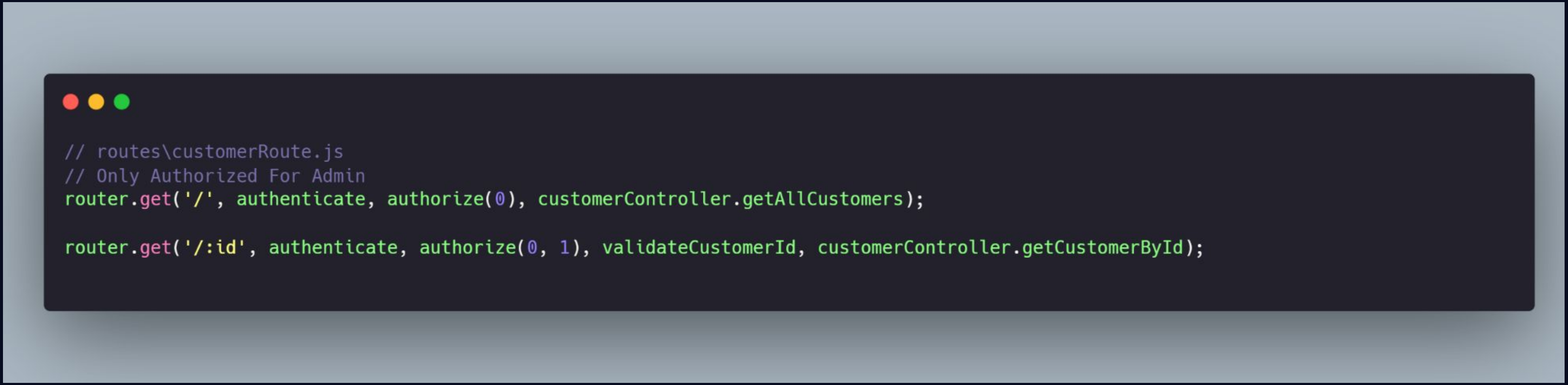


Insecure Direct Object Reference (IDOR) |Continued


Attacker changes the profile_id to the **victim's profile_id**
https://workshop.com/api/user_profile?id=132355
Profile info for user with id 132355
Attacker
id=1337
WEB
Server

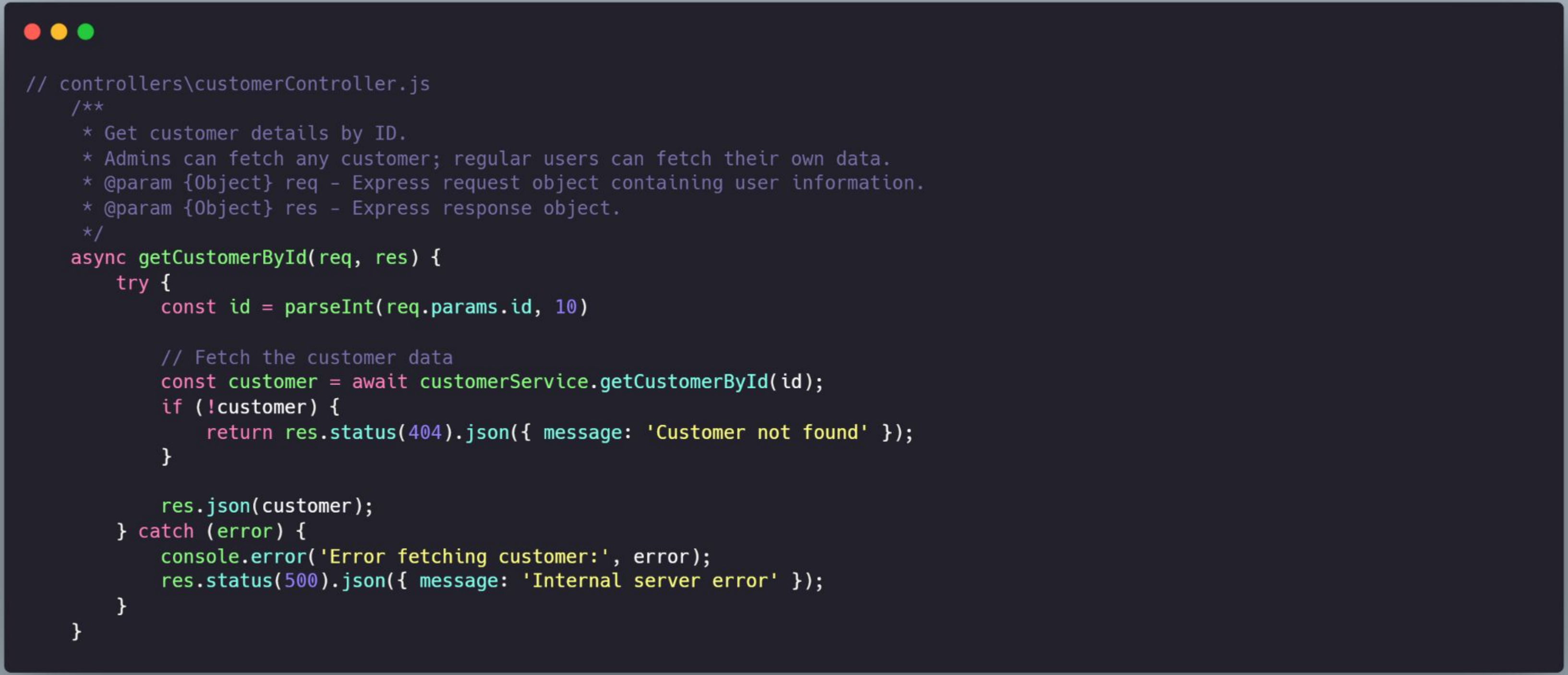


4) Scenario #1 - Real World Scenario



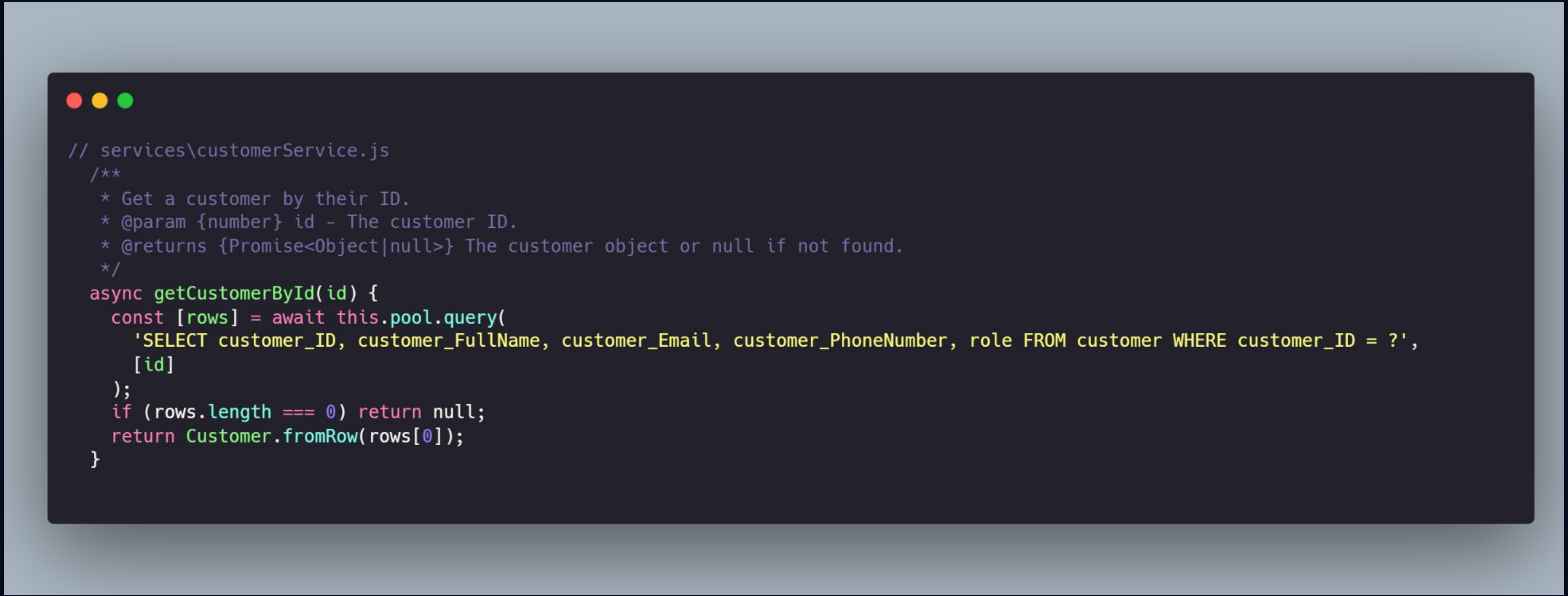
```
// routes\customerRoute.js
// Only Authorized For Admin
router.get('/', authenticate, authorize(0), customerController.getAllCustomers);

router.get('/:id', authenticate, authorize(0, 1), validateCustomerId, customerController.getCustomerById);
```

4) Scenario #1 - Real World Scenario



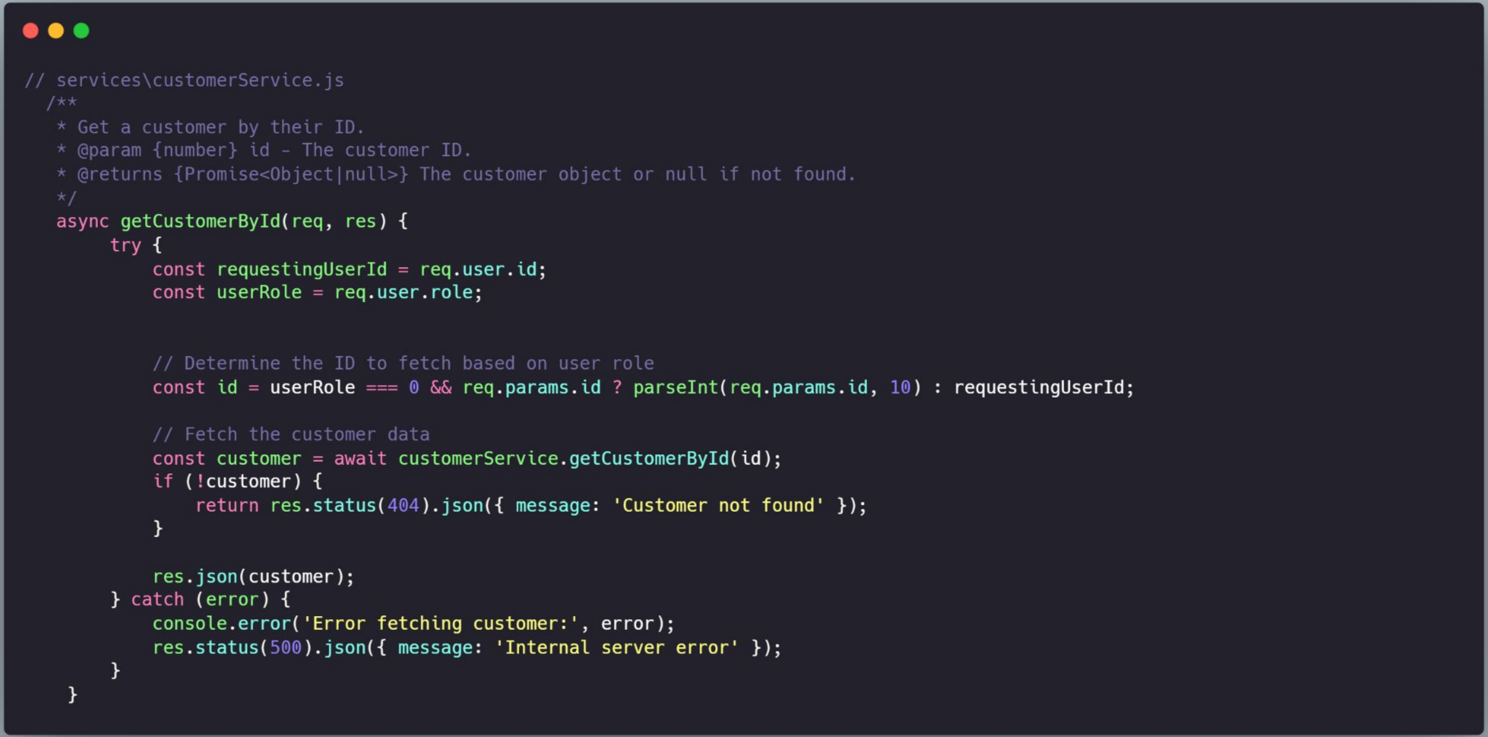
```javascript
// controllers\customerController.js
    /**
     * Get customer details by ID.
     * Admins can fetch any customer; regular users can fetch their own data.
     * @param {Object} req - Express request object containing user information.
     * @param {Object} res - Express response object.
     */
    async getCustomerById(req, res) {
        try {
            const id = parseInt(req.params.id, 10)

            // Fetch the customer data
            const customer = await customerService.getCustomerById(id);
            if (!customer) {
                return res.status(404).json({ message: 'Customer not found' });
            }

            res.json(customer);
        } catch (error) {
            console.error('Error fetching customer:', error);
            res.status(500).json({ message: 'Internal server error' });
        }
    }
```

4) Scenario #1 - Real World Scenario



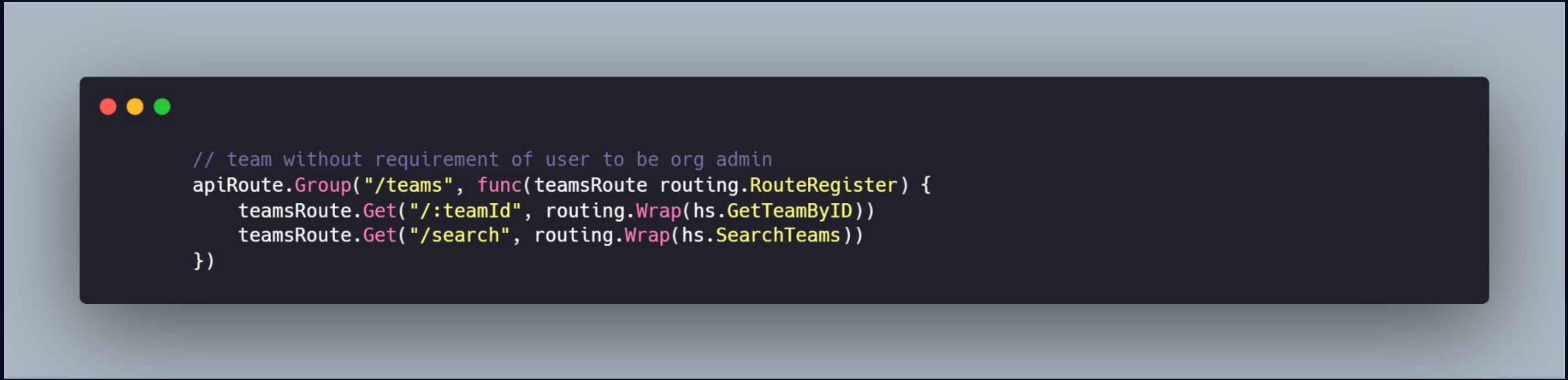
```
// services\customerService.js
  /**
   * Get a customer by their ID.
   * @param {number} id - The customer ID.
   * @returns {Promise<Object|null>} The customer object or null if not found.
   */
  async getCustomerById(id) {
    const [rows] = await this.pool.query(
      'SELECT customer_ID, customer_FullName, customer_Email, customer_PhoneNumber, role FROM customer WHERE customer_ID = ?',
      [id]
    );
    if (rows.length === 0) return null;
    return Customer.fromRow(rows[0]);
  }
```

4) Scenario #1 - Real World Scenario | Patch



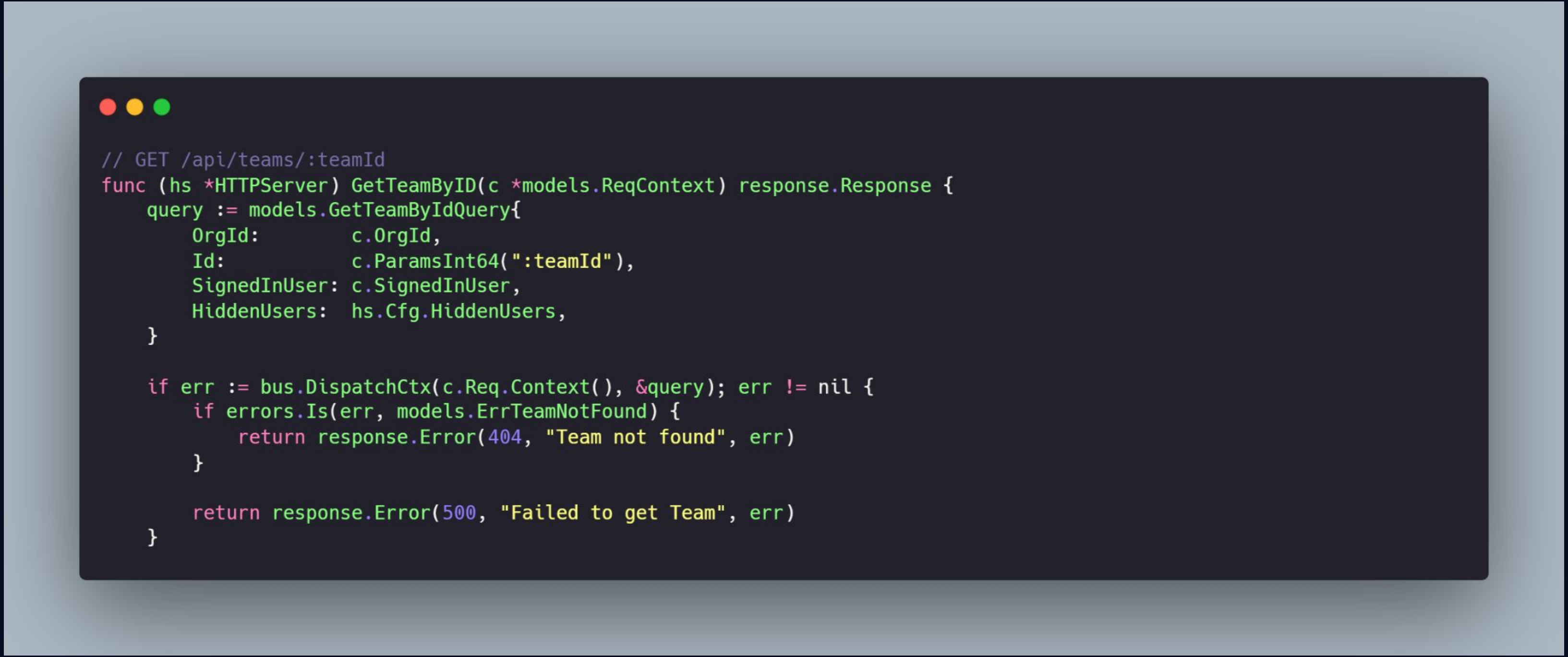
```javascript
// services\customerService.js
  /**
   * Get a customer by their ID.
   * @param {number} id - The customer ID.
   * @returns {Promise<Object|null>} The customer object or null if not found.
   */
   async getCustomerById(req, res) {
        try {
            const requestingUserId = req.user.id;
            const userRole = req.user.role;


            // Determine the ID to fetch based on user role
            const id = userRole === 0 && req.params.id ? parseInt(req.params.id, 10) : requestingUserId;

            // Fetch the customer data
            const customer = await customerService.getCustomerById(id);
            if (!customer) {
                return res.status(404).json({ message: 'Customer not found' });
            }

            res.json(customer);
        } catch (error) {
            console.error('Error fetching customer:', error);
            res.status(500).json({ message: 'Internal server error' });
        }
    }
```

4) Scenario #2 - Grafana Teams API IDOR | CVE-2022-21713



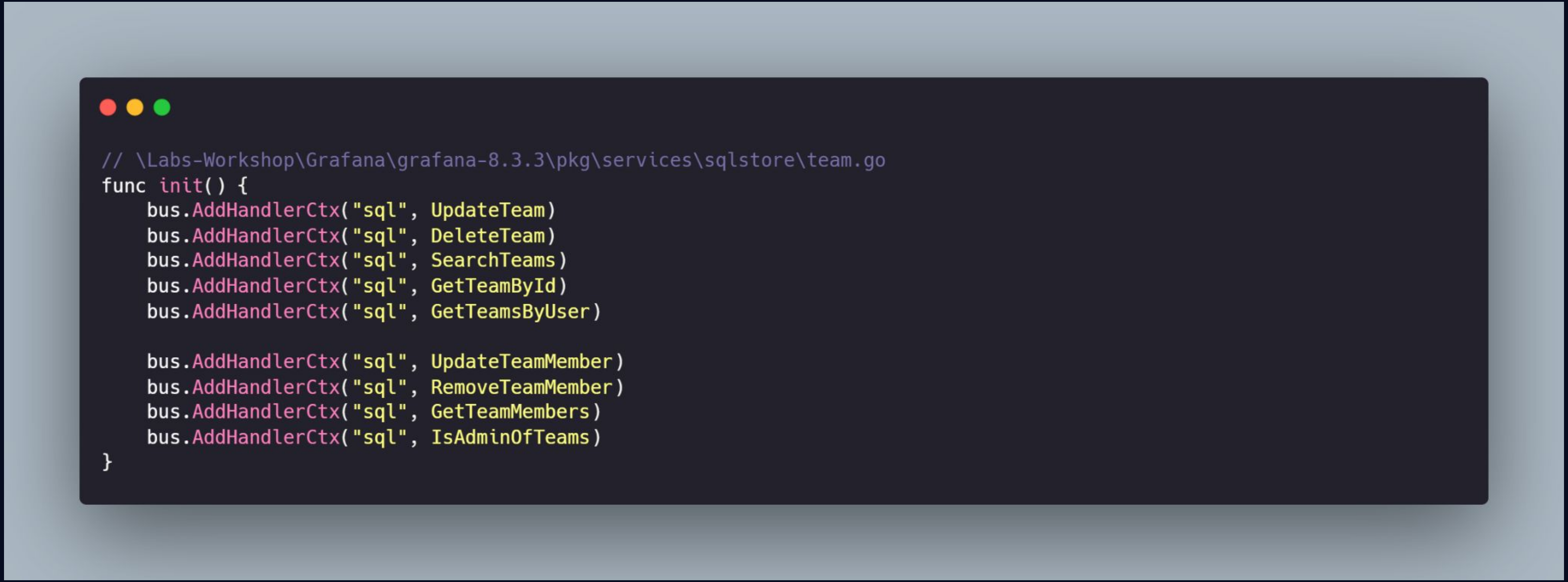
```
// team without requirement of user to be org admin
apiRoute.Group("/teams", func(teamsRoute routing.RouteRegister) {
    teamsRoute.Get("/:teamId", routing.Wrap(hs.GetTeamByID))
    teamsRoute.Get("/search", routing.Wrap(hs.SearchTeams))
})
```

4) Scenario #2 - Grafana Teams API IDOR | CVE-2022-21713



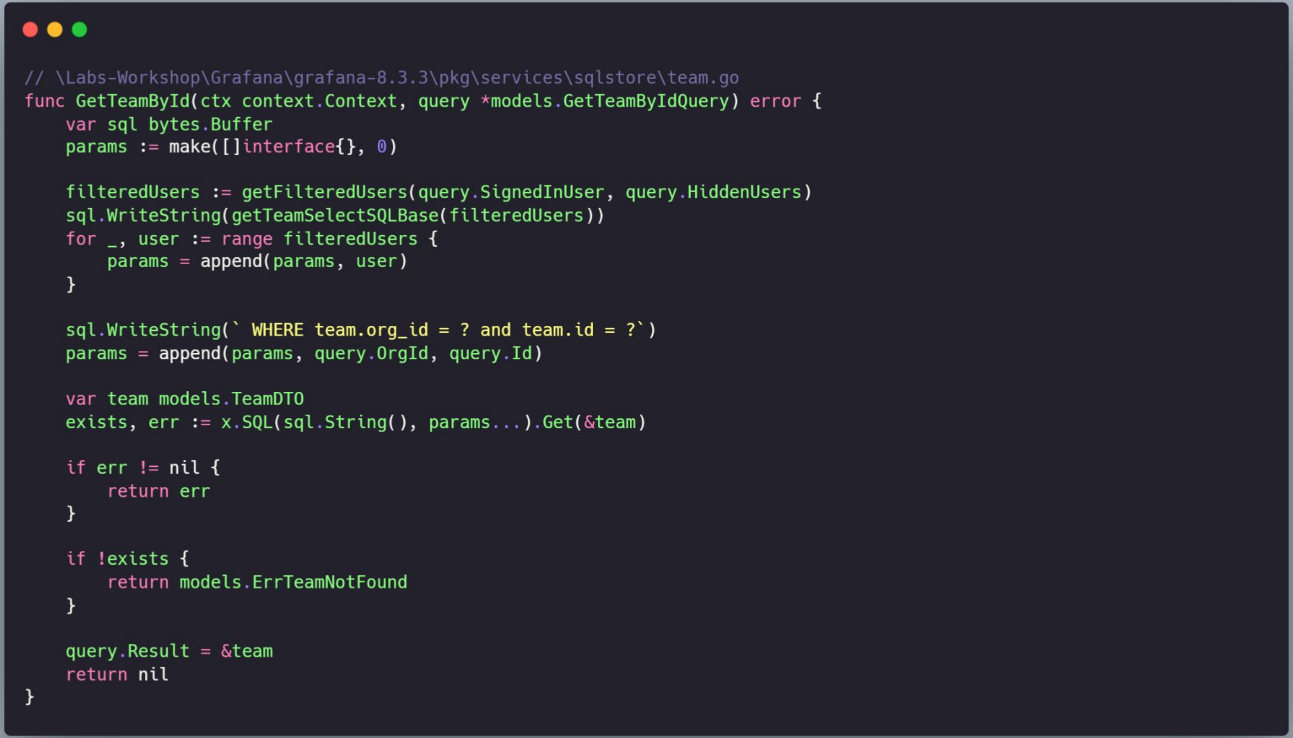
```go
// GET /api/teams/:teamId
func (hs *HTTPServer) GetTeamByID(c *models.ReqContext) response.Response {
    query := models.GetTeamByIdQuery{
        OrgId:        c.OrgId,
        Id:           c.ParamsInt64(":teamId"),
        SignedInUser: c.SignedInUser,
        HiddenUsers:  hs.Cfg.HiddenUsers,
    }

    if err := bus.DispatchCtx(c.Req.Context(), &query); err != nil {
        if errors.Is(err, models.ErrTeamNotFound) {
            return response.Error(404, "Team not found", err)
        }

        return response.Error(500, "Failed to get Team", err)
    }
```

4) Scenario #2 - Grafana Teams API IDOR | CVE-2022-21713



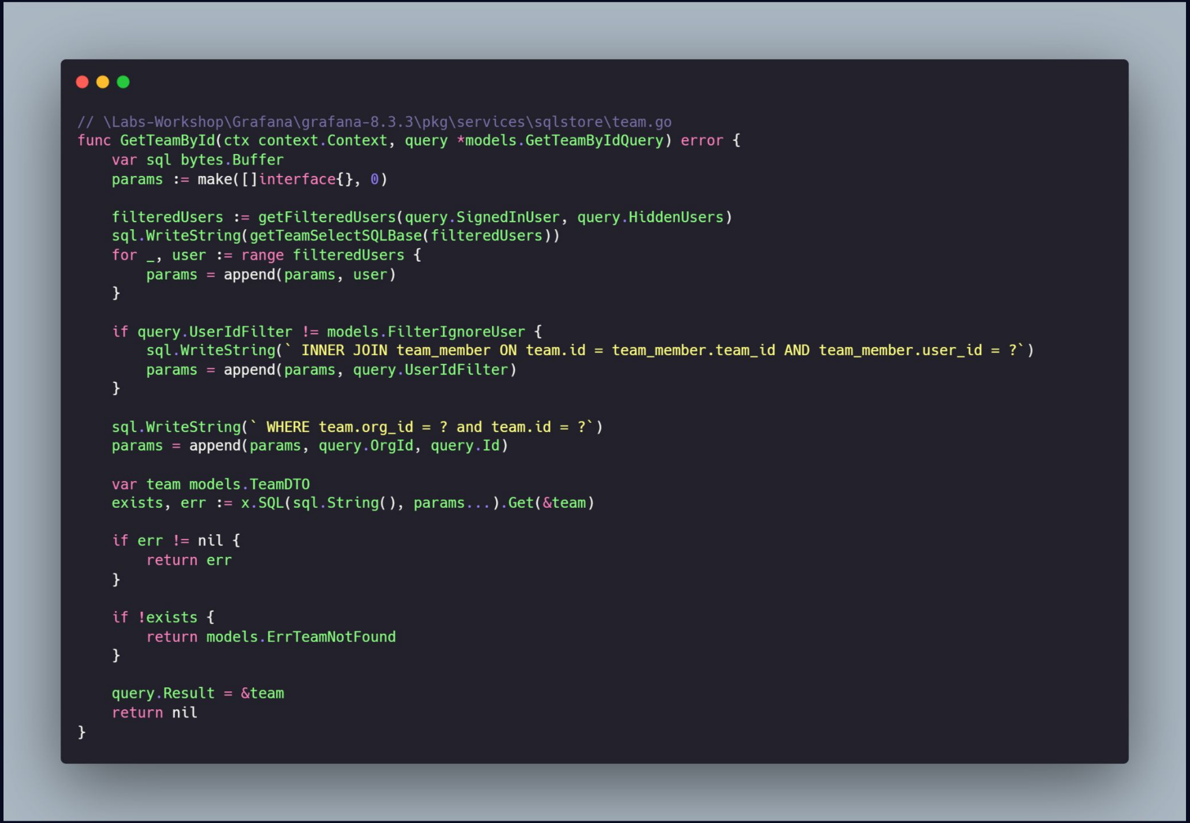
```
// \Labs-Workshop\Grafana\grafana-8.3.3\pkg\services\sqlstore\team.go
func init() {
    bus.AddHandlerCtx("sql", UpdateTeam)
    bus.AddHandlerCtx("sql", DeleteTeam)
    bus.AddHandlerCtx("sql", SearchTeams)
    bus.AddHandlerCtx("sql", GetTeamById)
    bus.AddHandlerCtx("sql", GetTeamsByUser)

    bus.AddHandlerCtx("sql", UpdateTeamMember)
    bus.AddHandlerCtx("sql", RemoveTeamMember)
    bus.AddHandlerCtx("sql", GetTeamMembers)
    bus.AddHandlerCtx("sql", IsAdminOfTeams)
}
```

4) Scenario #2 - Grafana Teams API IDOR | CVE-2022-21713



```go
// \Labs-Workshop\Grafana\grafana-8.3.3\pkg\services\sqlstore\team.go
func GetTeamById(ctx context.Context, query *models.GetTeamByIdQuery) error {
    var sql bytes.Buffer
    params := make([]interface{}, 0)

    filteredUsers := getFilteredUsers(query.SignedInUser, query.HiddenUsers)
    sql.WriteString(getTeamSelectSQLBase(filteredUsers))
    for _, user := range filteredUsers {
        params = append(params, user)
    }

    sql.WriteString(` WHERE team.org_id = ? and team.id = ?`)
    params = append(params, query.OrgId, query.Id)

    var team models.TeamDTO
    exists, err := x.SQL(sql.String(), params...).Get(&team)

    if err != nil {
        return err
    }

    if !exists {
        return models.ErrTeamNotFound
    }

    query.Result = &team
    return nil
}
```

4) Scenario #2 - Grafana Teams API IDOR | CVE-2022-21713



```go
// \Labs-Workshop\Grafana\grafana-8.3.3\pkg\services\sqlstore\team.go
func GetTeamById(ctx context.Context, query *models.GetTeamByIdQuery) error {
    var sql bytes.Buffer
    params := make([]interface{}, 0)

    filteredUsers := getFilteredUsers(query.SignedInUser, query.HiddenUsers)
    sql.WriteString(getTeamSelectSQLBase(filteredUsers))
    for _, user := range filteredUsers {
        params = append(params, user)
    }

    if query.UserIdFilter != models.FilterIgnoreUser {
        sql.WriteString(` INNER JOIN team_member ON team.id = team_member.team_id AND team_member.user_id = ?`)
        params = append(params, query.UserIdFilter)
    }

    sql.WriteString(` WHERE team.org_id = ? and team.id = ?`)
    params = append(params, query.OrgId, query.Id)

    var team models.TeamDTO
    exists, err := x.SQL(sql.String(), params...).Get(&team)

    if err != nil {
        return err
    }

    if !exists {
        return models.ErrTeamNotFound
    }

    query.Result = &team
    return nil
}
```

05 Broken Access Control

Broken Access Control

- Within the application’s core security mechanisms, access controls are logically built on **authentication** and **session management**.

- **Authentication:** verify a user’s identity.

- **Session management:** confirm that a particular sequence of requests that the web application receives originated from the same user.

Broken Access Control |Continued

- ❖ The reason that the application needs to do these things is because it needs a way to decide whether it should **permit** a given request to perform its attempted action or access the resources it is requesting.

- ❖ **Access controls** (or **authorization**) are responsible for making these key decisions.

- ❖ Access control vulnerabilities are conceptually simple: **The application lets you do something you shouldn't be able to**.

Broken Access Control |Continued

Access controls can be divided into three broad categories:

- ❖ **Vertical:** allow different types of users to access different parts of the application’s functionality. **Example:** division between normal users and administrators.

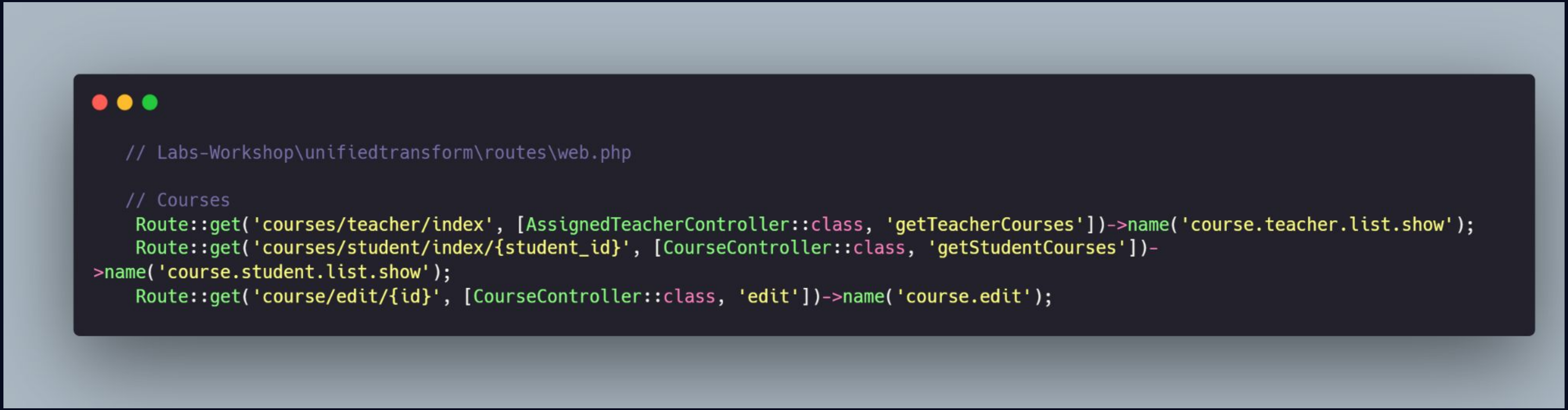
- ❖ **Horizontal:** allow users to access a certain subset of a wider range of resources of the same type. **Example:** a web mail application may allow you to read your email but no one else’s.

Scenario #1 - Real World Scenario

Maybe I am crazy. Let’s introduce it directly.

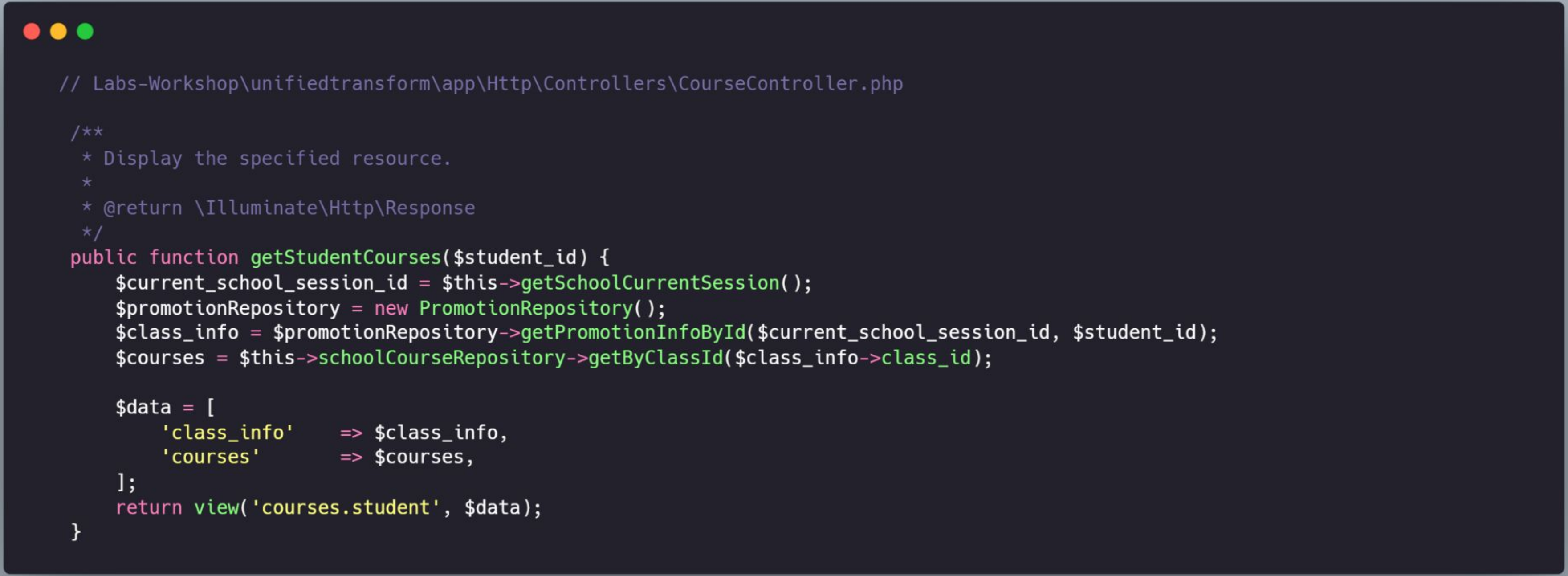
;)

Scenario #2 - unifiedtransform | CVE-2024-2292



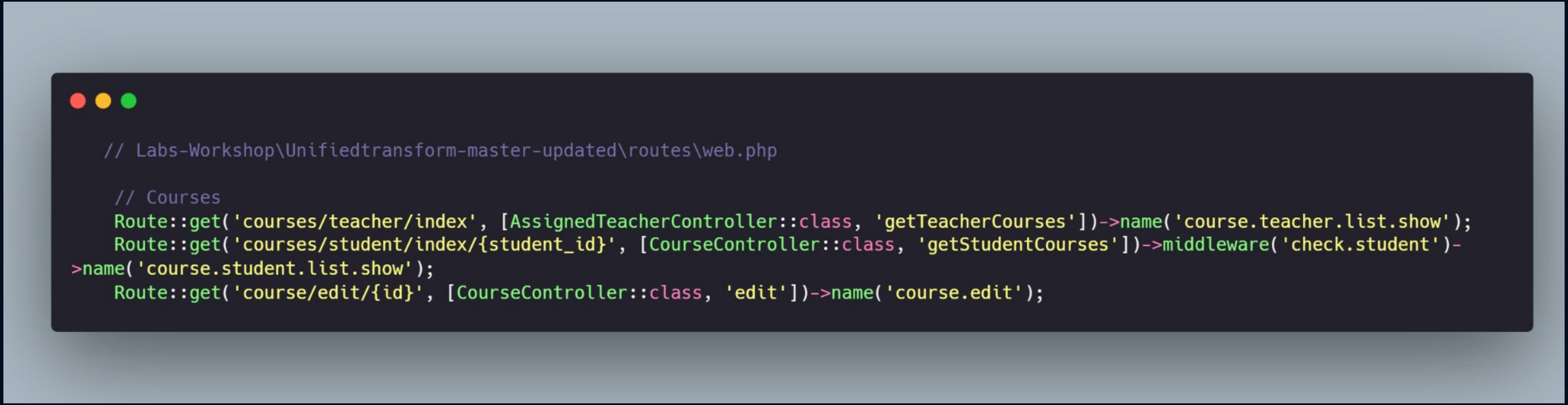
```
// Labs-Workshop\unifiedtransform\routes\web.php

// Courses
 Route::get('courses/teacher/index', [AssignedTeacherController::class, 'getTeacherCourses'])->name('course.teacher.list.show');
 Route::get('courses/student/index/{student_id}', [CourseController::class, 'getStudentCourses'])-
>name('course.student.list.show');
 Route::get('course/edit/{id}', [CourseController::class, 'edit'])->name('course.edit');
```

Scenario #2 - unifiedtransform | CVE-2024-2292



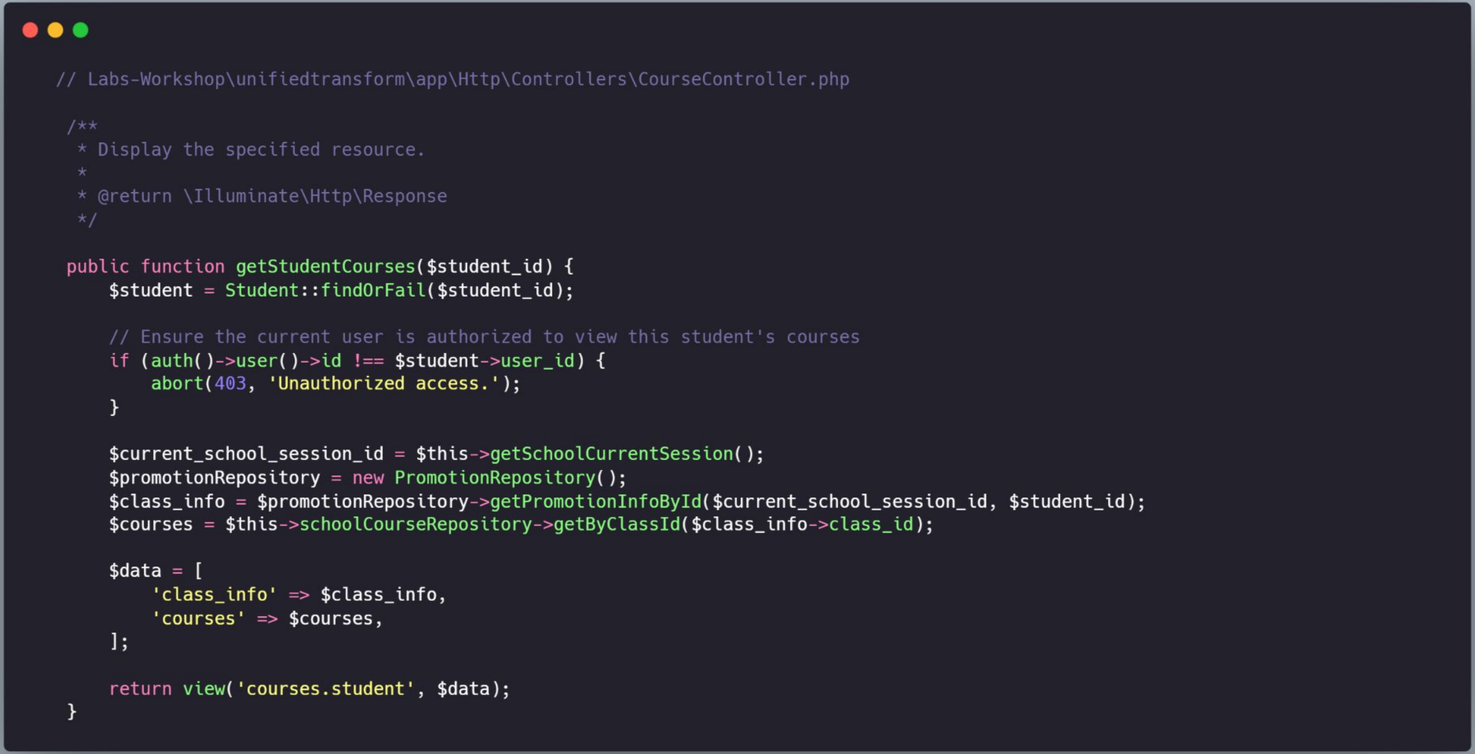
```php
// Labs-Workshop\unifiedtransform\app\Http\Controllers\CourseController.php

 /**
  * Display the specified resource.
  *
  * @return \Illuminate\Http\Response
  */
 public function getStudentCourses($student_id) {
     $current_school_session_id = $this->getSchoolCurrentSession();
     $promotionRepository = new PromotionRepository();
     $class_info = $promotionRepository->getPromotionInfoById($current_school_session_id, $student_id);
     $courses = $this->schoolCourseRepository->getByClassId($class_info->class_id);

     $data = [
         'class_info'    => $class_info,
         'courses'       => $courses,
     ];
     return view('courses.student', $data);
 }
```

Scenario #2 - unifiedtransform | CVE-2024-2292 |Pacth



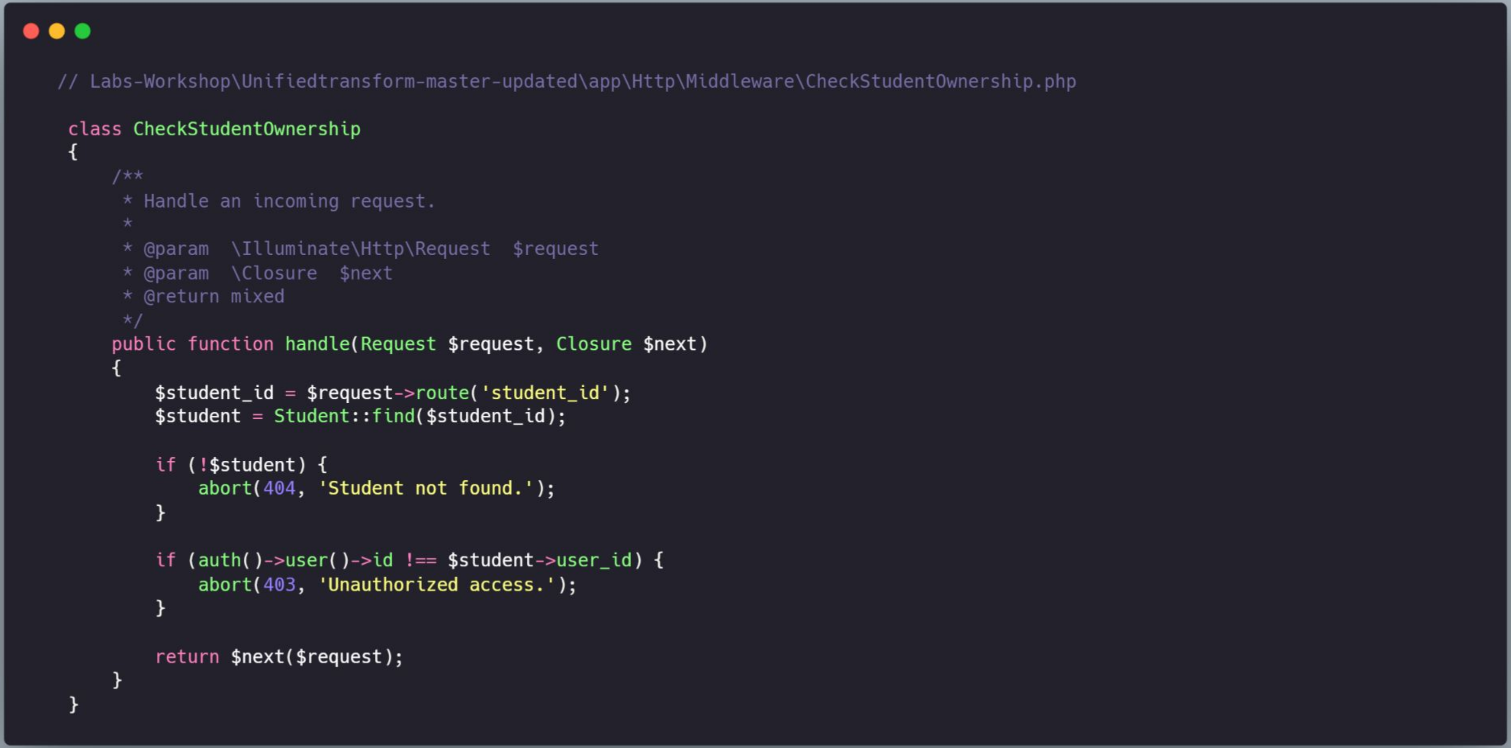
```
   // Labs-Workshop\Unifiedtransform-master-updated\routes\web.php

    // Courses
    Route::get('courses/teacher/index', [AssignedTeacherController::class, 'getTeacherCourses'])->name('course.teacher.list.show');
    Route::get('courses/student/index/{student_id}', [CourseController::class, 'getStudentCourses'])->middleware('check.student')-
>name('course.student.list.show');
    Route::get('course/edit/{id}', [CourseController::class, 'edit'])->name('course.edit');
```

Scenario #2 - unifiedtransform | CVE-2024-2292 |Patch



```php
// Labs-Workshop\unifiedtransform\app\Http\Controllers\CourseController.php

 /**
  * Display the specified resource.
  *
  * @return \Illuminate\Http\Response
  */

 public function getStudentCourses($student_id) {
     $student = Student::findOrFail($student_id);

     // Ensure the current user is authorized to view this student's courses
     if (auth()->user()->id !== $student->user_id) {
         abort(403, 'Unauthorized access.');
     }

     $current_school_session_id = $this->getSchoolCurrentSession();
     $promotionRepository = new PromotionRepository();
     $class_info = $promotionRepository->getPromotionInfoById($current_school_session_id, $student_id);
     $courses = $this->schoolCourseRepository->getByClassId($class_info->class_id);

     $data = [
         'class_info' => $class_info,
         'courses' => $courses,
     ];

     return view('courses.student', $data);
 }
```

Scenario #2 - unifiedtransform | CVE-2024-2292 |Patch



```php
// Labs-Workshop\Unifiedtransform-master-updated\app\Http\Middleware\CheckStudentOwnership.php

class CheckStudentOwnership
{
    /**
     * Handle an incoming request.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  \Closure  $next
     * @return mixed
     */
    public function handle(Request $request, Closure $next)
    {
        $student_id = $request->route('student_id');
        $student = Student::find($student_id);

        if (!$student) {
            abort(404, 'Student not found.');
        }

        if (auth()->user()->id !== $student->user_id) {
            abort(403, 'Unauthorized access.');
        }

        return $next($request);
    }
}
```

06 Information Disclosure

Information Disclosure

“Information disclosure, also referred to as information leakage, occurs when a website inadvertently exposes sensitive information to its users. This can allow potential attackers to access various types of data”, including:

- ❖ Information about other users, such as usernames or financial details
- ❖ Confidential business or commercial data
- ❖ Technical information about the website and its infrastructure

Scenario #1 - Real World Scenario

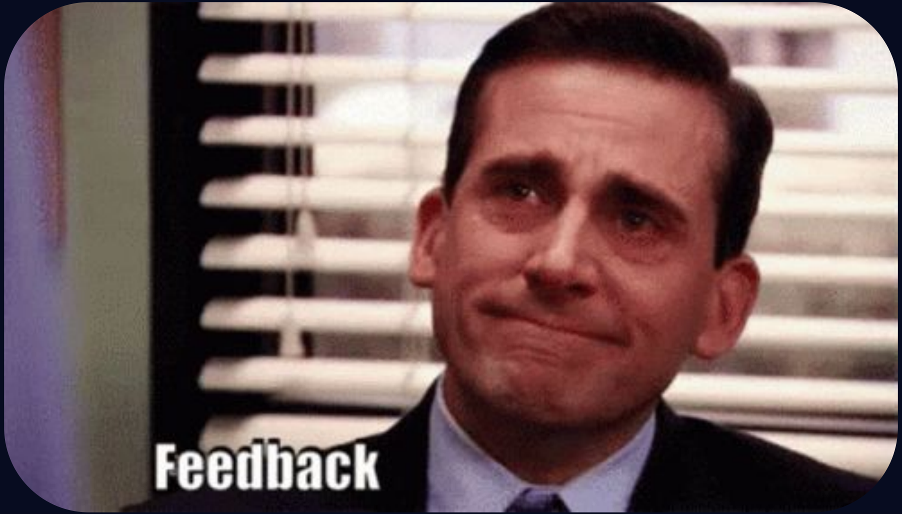
<u>**Maybe I am crazy. Let’s introduce it directly.**</u>

;)

07 What’s Next?

Questions?

Seeking Feedback



REACH ME!

Email: hassanalachek@hotmail.com

LinkedIn: in/hassan-al-achek

GitHub: @gokupwn

Social media: @hassanalachek

WTM Cyber Security Community