Client Server Systems

CRN: 50249

Assessment 1

George Wilkinson

@00677611

CHC119

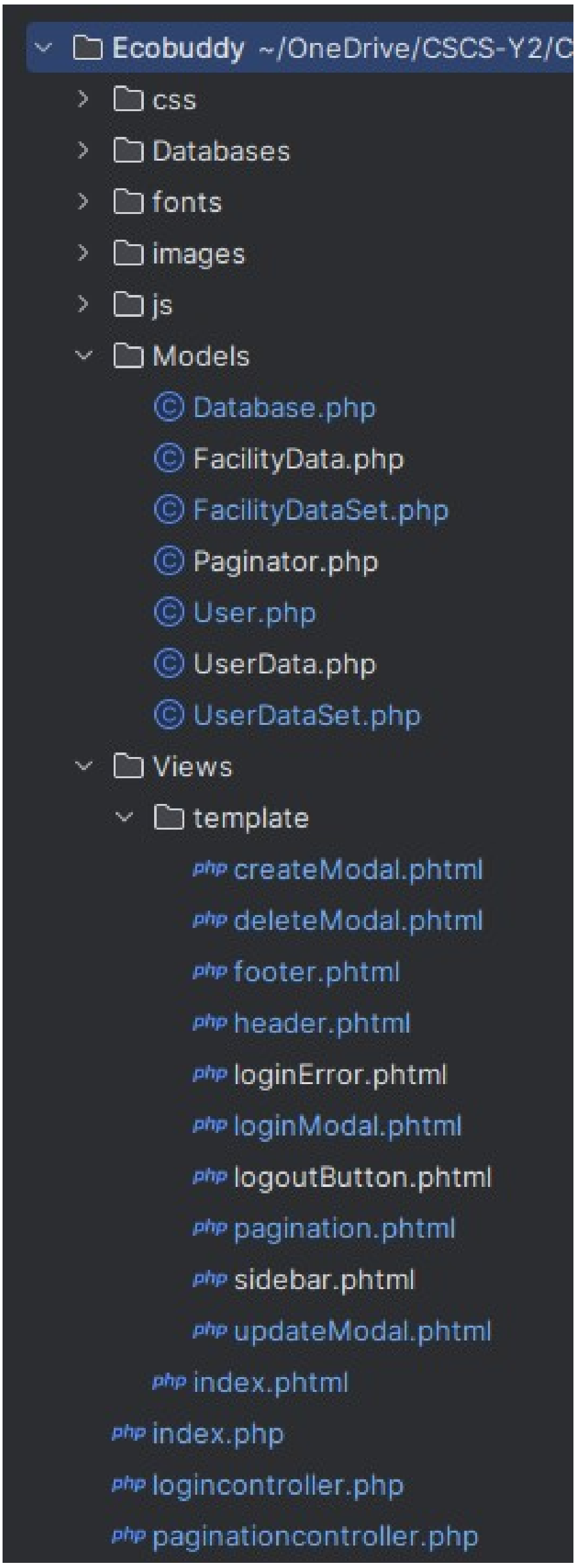
URL: http://localhost:8000

Lee Account: Lee FloatationSinkingDevice3

Admin Account: Admin ConcreteJungleSandwich2

Benny Account: Benny ForestElephantAutomatic7

Directory Layout



Mark range %	100-80%	79-60%	59-40%	39-20%	19-0%
Assessment Criteria	**"A system ready to go live"** 	**"A usable system with useful features"** 	**"A minimum viable system"** 	**"Basic data/items listings"** 	**"Non-functioning system"** 
Well designed OO code using MVC Design Pattern, strictly created using phpStorm, XAMPP and SQLite on Microsoft Windows. Document as required and marking scheme (20 Marks)	Considerable amount of OO code (including inheritance) demonstrating correct use of MVC, classes/properties and methods used with appropriate names, using PDO to access the database. Submitted code presented organised, neatly and readable and excellently commented. Demonstrable performance enhancements. Meets technical server requirements as detailed above. All documentation as required well presented and readable. Fully highlighted marking scheme included.	Reasonable amount of OO code demonstrating correct use of MVC, classes/properties and methods, using PDO to access the database. Implementation of performance enhancements. Submitted code presented organised, neatly and readable and well commented. Good documentation presented. Marking scheme highlighted. Meets technical server requirements as detailed above.	Small amount of OO code demonstrating correct general use of MVC, classes/properties and methods, using PDO to access the database. Consideration of performance. Code submitted and organised with some code comments. Marking scheme highlighted. Meets technical server requirements as detailed above, some issues.	Small amount of code , not OO, demonstrating limited understanding of MVC, classes/properties and methods, or database access. Basic code comments. Code submitted. Does not meet technical server requirements. **Significant amounts of copied code.** Poorly presented document. **Extensive use of AI tools.**	Very little code demonstrating no real understanding of MVC, classes/properties and methods, or database access. Little or no code comments. Does not meet technical server requirements. **Significant amounts of copied code.** Poorly presented document. **Extensive use of AI tools.**
Database and system	Created using SQLite via phpStorm and running on Microsoft Windows under phpStorm with XAMPP based PHP CLI only.			**Not** built on the platform specified in the brief above. Significant run time issues.	
User feature (20 Marks)	Manager users can login successfully using encryption for their passwords and manage all aspects of the **ecoFacility** records. Anti-spam feature used. Some form of session used to maintain state. Validation on input fields. Malicious code filtering on items. Driven by excellent, logical OO design. Excellently commented code. Test accounts working.	Manager users can login successfully using encryption for their passwords and create new **ecoFacility** records. Some form of session used to maintain state. Validation on input fields. Driven by good, logical OO design. Well commented code. Test accounts working.	Users can login successfully. Some form of session used to maintain state. Basic OO implementation. Some code comments. Test accounts working.	User login not fully implemented/working correctly. Test accounts **not** fully working. Basic code comments. **Significant amounts of copied code.** Poorly presented document. **Extensive use of AI tools.**	Login not implemented, or not working. No test accounts or not working correctly. **Significant amounts of copied code.** Poorly presented document. **Extensive use of AI tools.**
Browsing user can display information from records stored in the database (20 Marks)	Browsing users can retrieve and display Some code **ecoFacility** details. Sophisticated, responsive layout using CSS/Bootstrap. Includes efficient paging. Demonstration of system with large number of realistic records/users. Driven by excellent, logical OO design. Excellently commented code.	Browsing users can retrieve and display Some code **ecoFacility** details, including images. Good responsive layout using CSS/Bootstrap. Attempt at basic paging. Demonstration of system with large number of records/users. Driven by good, logical OO design. Well commented code.	Browsing users can retrieve some item details. Basic cellular listing layout using tables or divs with some formatting evident. Basic OO implementation. Some code comments.	Browsing users can retrieve some item details. Simple table or list of text data, issues with displaying data. Basic code comments. **Significant amounts of copied code.** Poorly presented document. **Extensive use of AI tools.**	Users cannot retrieve or display details from the database. Little or no code comments. **Significant amounts of copied code.** Poorly presented document. **Extensive use of AI tools.**
Ability for browsing user to search for **ecoFacility** (20 Marks)	Comprehensive, interactive faceted search to narrow down output to less than 10 results using all record parameters. Complex SQL statements. Driven by excellent, logical OO design. Excellently commented code.	Free text search facility with 3 parameters **combined** into the search. Good OO implementation. Well commented code.	Free text search facility working but basic. Some basic filters implemented, but may have issue. Basic OO implementation. Some code comments.	Basic text search facility may have issues. Basic code comments. **Significant amounts of copied code.** Poorly presented document. **Extensive use of AI tools.**	Some search/filters partly implemented or no implementation. Little or no code comments. **Significant amounts of copied code.** Poorly presented document. **Extensive use of AI tools.**
Ability to create/edit records (Manager). (20 Marks)	Manager users can add, view and edit/add **ecoFacility** records. Driven by excellent, logical OO design. Excellently commented code including single lines as appropriate and function/class summaries.	Manager users can view and add new **ecoFacility** records. Driven by good, OO design. Well and consistently commented code.	Attempt at Manager features for viewing **ecoFacility** data with some success. Not sophisticated/some issues present. A Basic OO implementation. Some code comments.	Attempt at Manager features for viewing **ecoFacility** data with some success. Basic code comments. **Significant amounts of copied code.** Poorly presented document. **Extensive use of AI tools.**	Little or no implementation. Little or no code comments. **Significant amounts of copied code.** Poorly presented document. **Extensive use of AI tools.**

Model Files

Database.php

```php
<?php
class Database {
    /**
     * @var Database
     */
    protected static $_dbInstance = null;
    /**
     * @var PDO
     */
    protected $_dbHandle;

    public function getDbConnection(): PDO
    {
        return $this->_dbHandle;
    }
    public static function getInstance(): ?Database
    {
        if(self::$_dbInstance == null) {
            self::$_dbInstance = new self();
        }
        return self::$_dbInstance;
    }

    private function __construct() {
        try {
            $this->_dbHandle = new PDO("sqlite:Databases/ecobuddynew.sqlite");
            $this->_dbHandle->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_WARNING);
            $this->_dbHandle->setAttribute(PDO::ATTR_EMULATE_PREPARES, false);
        }
        catch (PDOException $e) {
            echo $e->getMessage();
        }
    }
    public function __destruct() {
        $this->_dbHandle = null; // destroys the PDO handle when no longer needed
    }
}
```

FacilityData.php

```php
<?php
class FacilityData {
    protected $_id, $_title, $_category, $_status, $_description, $_houseNumber, $_streetName, $_county, $_town, $_postcode, $_lng, $_lat, $_contributor;

    public function __construct($dbRow) {
        $this->_id = $dbRow['id'];
        $this->_title = $dbRow['title'];
        $this->_category = $dbRow['category'];
        $this->_status = $dbRow['status'];
        $this->_description = $dbRow['description'];
        $this->_houseNumber = $dbRow['houseNumber'];
        $this->_streetName = $dbRow['streetName'];
        $this->_county = $dbRow['county'];
        $this->_town = $dbRow['town'];
        $this->_postcode = $dbRow['postcode'];
        $this->_lng = $dbRow['lng'];
        $this->_lat = $dbRow['lat'];
        $this->_contributor = $dbRow['contributor'];
    }

    public function getId() {
        return $this->_id;
    }
    public function getTitle() {
        return $this->_title;
    }
    public function getCategory() {
        return $this->_category;
    }

    public function getStatus() {
        return $this->_status;
    }
    public function getDescription() {
```

```
        return $this->_description;
    }
    public function getHouseNumber() {
        return $this->_houseNumber;
    }
    public function getStreetName() {
        return $this->_streetName;
    }
    public function getCounty() {
        return $this->_county;
    }
    public function getTown() {
        return $this->_town;
    }
    public function getPostcode() {
        return $this->_postcode;
    }
    public function getLng() {
        return $this->_lng;
    }
    public function getLat() {
        return $this->_lat;
    }
    public function getContributor() {
        return $this->_contributor;
    }
}
```

FacilityDataSet.php

```
<?php
require_once ('Database.php');
require_once ('FacilityData.php');

class FacilityDataSet
{
    protected $_dbHandle, $_dbInstance;

    public function __construct()
    {
        $this->_dbInstance = Database::getInstance();
        $this->_dbHandle = $this->_dbInstance->getDbConnection();
    }

    /**
     * @param $data
     * @return bool
     * Broken last minute, dont have time to fix.
     * add / update facility to database from array of columns
     */
    public function addFacility($data): bool
    {
        $userQuery = "
            SELECT ecoUser.id FROM ecoUser
            WHERE ecoUser.username = :contributor;
        ";
        $catQuery = "
            SELECT ecoCategories.id FROM ecoCategories
            WHERE ecoCategories.name = :category;
        ";
        $sqlQuery = "
            INSERT OR REPLACE INTO ecoFacilities
                (id,
                 title,
                 category,
                 description,
                 houseNumber,
                 streetName,
                 county,
                 town,
                 postcode,
                 lng,
                 lat,
                 contributor)
            VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, -1, -1, ?)
        ;";
```

```php
    // gets contributor name
    $stmt = $this->_dbHandle->prepare($userQuery);
    $stmt->bindParam(':contributor', $data->contributor, PDO::PARAM_STR);
    $stmt = $this->_dbHandle->prepare($userQuery);
    $stmt->execute();
    $data['contributor'] = (int)$stmt->fetch(PDO::FETCH_ASSOC);

    // gets category ID
    $stmt = $this->_dbHandle->prepare($catQuery);
    $stmt->bindParam(':category', $data->category, PDO::PARAM_STR);
    $stmt = $this->_dbHandle->prepare($catQuery);
    $stmt->execute();
    $data['category'] = (int)$stmt->fetch(PDO::FETCH_ASSOC);

    // run main query and bind updated parameters
    $stmt = $this->_dbHandle->prepare($sqlQuery);
    // Ensures only one value is returned per column name
    $stmt->setFetchMode(\PDO::FETCH_ASSOC);
    if (isset($data['id'])) {
        $stmt->bindParam(1, $data['id']);
    }
    $stmt->bindParam(2, $data['title'], PDO::PARAM_STR);
    $stmt->bindParam(3, $data['category'], PDO::PARAM_INT);
    $stmt->bindParam(4, $data['description'], PDO::PARAM_STR);
    $stmt->bindParam(5, $data['houseNumber'], PDO::PARAM_STR);
    $stmt->bindParam(6, $data['streetName'], PDO::PARAM_STR);
    $stmt->bindParam(7, $data['county'], PDO::PARAM_STR);
    $stmt->bindParam(8, $data['town'], PDO::PARAM_STR);
    $stmt->bindParam(9, $data['postcode'], PDO::PARAM_STR);
    $stmt->bindParam(10, $data['contributor'], PDO::PARAM_INT);
    $stmt->execute();
//      var_dump($stmt);
//      var_dump($this->_dbHandle->errorInfo());
    return !($stmt->rowCount());
  }


  /**
   * @param $id
   * @return bool
   * Deletes Facility Records being passed a facility id.
   */
  public function deleteFacility($id): bool
  {
    $sqlQuery = "DELETE FROM ecoFacilities WHERE ecoFacilities.id = :id;";
    $stmt = $this->_dbHandle->prepare($sqlQuery);
    $stmt->bindValue(':id', (int)$id, \PDO::PARAM_INT);
    $stmt->execute();
    var_dump($stmt);
    echo $stmt->rowCount();
    return !($stmt->rowCount() == 0);
  }


  /**

/**
 * @param $filterArray
 * @param $sortArray
 * @return array
 * Fetch all records depending on filters, and sort by defined column
 */
public function fetchAll($filterArray, $sortArray): array
{
  // Define columns for filtering and sorting
  $filterColumns = [
    0 => 'ecoFacilityStatus.statusComment',
    1 => 'ecoFacilities.title',
    2 => 'ecoCategories.name',
    3 => 'ecoFacilities.description',
    4 => 'ecoFacilities.streetName',
    5 => 'ecoFacilities.county',
    6 => 'ecoFacilities.town',
    7 => 'ecoFacilities.postcode',
    8 => 'ecoUser.username'
  ];

  $sortColumns = [
    0 => 'ecoFacilityStatus.statusComment',
    1 => 'ecoFacilities.title',
```

```
            2 => 'ecoCategories.name',
            3 => 'ecoFacilities.description',
            4 => 'ecoFacilities.streetName',
            5 => 'ecoFacilities.county',
            6 => 'ecoFacilities.town',
            7 => 'ecoFacilities.postcode',
            8 => 'ecoUser.username'
        ];

        // Validate and select the filter column
        $selectedFilterColumn = $filterColumns[$filterArray['category']] ?? 'ecoFacilities.title';
        // Validate and select the sort column
        $selectedSortColumn = $sortColumns[$sortArray['sort']] ?? 'ecoFacilities.title';
        // Validate sort direction
        $direction = strtolower($sortArray['dir']) === 'desc' ? 'DESC' : 'ASC';
        // Base query for filtering and sorting
        $baseQuery = "
        FROM ecoFacilities
        LEFT JOIN ecoCategories ON ecoCategories.id = ecoFacilities.category
        LEFT JOIN ecoUser ON ecoUser.id = ecoFacilities.contributor
        LEFT JOIN ecoFacilityStatus ON ecoFacilityStatus.facilityid = ecoFacilities.id
        WHERE {$selectedFilterColumn} LIKE :term
    ";


        // Query to count total results
        $countQuery = "SELECT COUNT(DISTINCT ecoFacilities.id) AS total {$baseQuery}";

        // Query to fetch filtered and sorted results
        $dataQuery = "
        SELECT DISTINCT ecoFacilities.id,
                        ecoFacilities.title,
                        GROUP_CONCAT(ecoFacilityStatus.statusComment, ', ') AS status,
                        ecoCategories.name AS category,
                        ecoFacilities.description,
                        ecoFacilities.houseNumber,
                        ecoFacilities.streetName,
                        ecoFacilities.county,
                        ecoFacilities.town,
                        ecoFacilities.postcode,
                        ecoFacilities.lng,
                        ecoFacilities.lat,
                        ecoUser.username AS contributor
        {$baseQuery}
        GROUP BY ecoFacilities.id, ecoFacilities.title, ecoCategories.name,
                 ecoFacilities.description, ecoFacilities.streetName,
                 ecoFacilities.county, ecoFacilities.town, ecoFacilities.postcode,
                 ecoUser.username
        ORDER BY {$selectedSortColumn} {$direction};
    ";
        // Surround 'term' with % to allow usage with LIKE
        $filterArray['term'] = '%' . $filterArray['term'] . '%' ?? '%';
        // Prepare and execute the count query
        $countStmt = $this->_dbHandle->prepare($countQuery);
        $countStmt->bindValue(':term', $filterArray['term'], PDO::PARAM_STR);
        $countStmt->execute();
        // Set total results to output of count statement
        $totalResults = (int)$countStmt->fetchColumn();

        // Prepare and execute the data query
        $dataStmt = $this->_dbHandle->prepare($dataQuery);
        $dataStmt->bindValue(':term', $filterArray['term'], PDO::PARAM_STR);
        $dataStmt->execute();

        // Fetch results into FacilityData objects
        $dataSet = [];
        while ($row = $dataStmt->fetch()) {
            $dataSet[] = new FacilityData($row);
        }

        return [
            'dataset' => $dataSet,
            'count' => $totalResults
        ];
    }
}
```

Paginator.php

```php
<?php
require_once('FacilityDataSet.php');
class Paginator {
    protected $_pages, $_totalPages, $_rowLimit, $_pageMatrix, $_rowCount;

    public function __construct($rowLimit, $dataset) {
        $this->_rowLimit = $rowLimit;
        $this->_totalPages = $this->calculateTotalPages($dataset['count']);
        $this->_rowCount = $dataset['count'];
        $this->_pages = $dataset['dataset'];
        $this->_pageMatrix = $this->Paginate();
    }
    public function getTotalPages() {
        return $this->_totalPages;
    }
    private function calculateTotalPages(int $count): int {
        return $count > 0 ? ceil($count / $this->_rowLimit) : 0;
    }

    public function Paginate(): array {
        $pageMatrix = [];
        for ($i = 0; $i < $this->_totalPages; $i++) {
            $page = [];
            $start = $i * $this->_rowLimit;
            $end = min($start + $this->_rowLimit, $this->_rowCount); // Ensure within bounds

            for ($j = $start; $j < $end; $j++) {
                $page[] = $this->_pages[$j];
            }

            $pageMatrix[$i] = $page;
        }
        return $pageMatrix;
    }

    public function getPageFromUri(): int {
        // Retrieve 'page' parameter and default to 0 if missing or invalid
        return filter_input(INPUT_GET, 'page', FILTER_VALIDATE_INT, [
            'options' => ['default' => 0, 'min_range' => 0] // Default to 1 if invalid or missing
        ]);
    }

    public function getPage(int $pageNumber): array {

        if ($pageNumber < 0 || $pageNumber >= $this->_totalPages) {
            return []; // Return an empty array if the page number is invalid
        }
        return $this->_pageMatrix[$pageNumber];
    }

    public function countPageResults(int $pageNumber): int {
        if ($pageNumber < 0 || $pageNumber >= $this->_totalPages) {
            return 0; // Return 0 if the page number is invalid
        }
        return count($this->_pageMatrix[$pageNumber]);
    }
}
```

User.php

```php
<?php

require_once('UserDataSet.php');
class User {
    protected $_username, $_loggedIn, $_userId, $_accessLevel;

    public function getUsername() {
        return $this->_username;
    }
    public function getUserId() {
        return $this->_userId;
    }

    /**
     *  Open session, set field variables
```

```php
 */
public function __construct() {
    session_start();

    $this->_username = "None";
    $this->_loggedIn = false;
    $this->_userId = "0";
    $this->_accessLevel = null;
    // if user logged in, set variables.
    if(isset($_SESSION['login'])) {
        $this->_username = $_SESSION['login'];
        $this->_userId = $_SESSION['uid'];
        $this->_loggedIn = true;
        $this->_accessLevel = $_SESSION['accessLevel'];
    }
}

private function setAccessLevel($level) {
    $this->_accessLevel = $level;
    $_SESSION['accessLevel'] = $level;
}
public function getAccessLevel() {
    return $this->_accessLevel;
}

/**
 * @param $username
 * @param $password
 * @return bool
 * Using a username and password, authenticate a user and assign variables from query
 */
public function Authenticate($username, $password): bool
{
    $users = new UserDataSet();
    $userDataSet = $users->checkUserCredentials($username, $password);
    $accessLevel = $users->checkAccessLevel($username);
    if(count($userDataSet) > 0) {
        $_SESSION['login'] = $username;
        $_SESSION['uid'] = $userDataSet[0]->getId();
        $this->setAccessLevel($accessLevel);
        $this->_loggedIn = true;
        $this->_username = $username;
        $this->_userId = $userDataSet[0]->getId();
        return true;
    }
    else {
        $this->_loggedIn = false;
        return false;
    }
}

/**
 * @return void
 * Unset user variables from session, and set variables to default values - destroying session.
 */
public function logout() {
    unset($_SESSION['login']);
    unset($_SESSION['uid']);
    $this->_loggedIn = false;
    $this->_username = "None";
    $this->_userId = "0";
    session_destroy();
}

public function isLoggedIn(): bool
{
    return $this->_loggedIn;
}
public function __destruct()
{

}


}
```

UserData.php

```php
<?php
class UserData {
    protected $_id, $_username, $_name, $_password, $_usertype;

    public function __construct($dbRow) {
        $this->_id = $dbRow['id'];
        $this->_username = $dbRow['username'];
        $this->_password = $dbRow['password'];
        $this->_usertype = $dbRow['userType'];
    }

    public function getId() {
        return $this->_id;
    }

    public function getUsername() {
        return $this->_username;
    }

}
```

UserDataSet.php

```php
<?php
require_once ('Database.php');
require_once ('UserData.php');

class UserDataSet {
    protected $_dbHandle, $_dbInstance;

    public function __construct() {
        $this->_dbInstance = Database::getInstance();
        $this->_dbHandle = $this->_dbInstance->getDbConnection();
    }

    /**
     * @param $username
     * @return mixed
     * Query access level of a username, and return their usertype
     */
    public function checkAccessLevel($username) {
        $sqlQuery = "SELECT ecoUser.userType FROM ecoUser
                    LEFT JOIN ecoUsertypes ON ecoUser.userType = ecoUsertypes.userType
                    WHERE ecoUser.username = ?";
        $statement = $this->_dbHandle->prepare($sqlQuery);
        $statement->bindValue(1, $username);
        $statement->execute();
        return $statement->fetch(PDO::FETCH_ASSOC)['userType'];
    }

    /**
     * @param $username
     * @param $password
     * @return array
     * Authenticate user with query, and return their details
     */
    public function checkUserCredentials($username, $password): array
    {
        $sqlQuery = 'SELECT * FROM ecoUser WHERE username = ? AND password = ?;';
        $statement = $this->_dbHandle->prepare($sqlQuery);
        $statement->bindParam(1, $username);
        $statement->bindParam(2, $password);
        $statement->execute();
        $dataSet = [];
        while ($row = $statement->fetch()) {
            $dataSet[] = new UserData($row);
        }
        return $dataSet;
    }
}
```

Controller Files

Index.php

```php
<?php
// load dataset
require_once('Models/UserDataSet.php');
// make a view class
$view = new stdClass();
$view->pageTitle = 'Home';

// load login controller and pagination controller
require_once("logincontroller.php");
require_once('paginationcontroller.php');

$view->user = new User();

// load main view
require_once('Views/index.phtml');
```

logincontroller.php

```php
<?php

require_once("Models/User.php");

// create user and dataset object
$user = new User();
$userDataSet = new UserDataSet();

if (isset($_POST["loginButton"])) {
    $username = $_POST["username"];
    // hash password
    $password = (hash("sha256", $_POST["password"]));
    // if login error, show captcha
    if (isset($view->loginError)) {
        $generatedCaptcha = $_POST["generatedCaptcha"];
        $userCaptcha = $_POST["captcha"];

        // if captcha wrong, say so
        if ($generatedCaptcha !== $userCaptcha) {
            $view->loginError = "Incorrect CAPTCHA.";
            return;
        }
    }

    // create a new student dataset object that we can generate data from
    // Error handling is VERY hacky, because of the lack of JS usage.
    if($userDataSet->checkUserCredentials($username, $password)) {
        $user->Authenticate($username, $password);
        // Unset modal boolean to hide it's usage.
        unset($_GET['modal']);
    } else {
    // Add error message and redirect to display modal
    $view->loginError = "Invalid username or password.";
    // Set modal boolean to header to allow modal to reappear
    $queryParams = http_build_query(['modal' => 'true']);
    header("Location: {$_SERVER['PHP_SELF']}?$queryParams");
    exit;
    }
}

if(isset($_POST['closeButton'])) {
    unset($_GET['modal']);
}
if (isset($_POST["logoutButton"]))
{
    $user->logout();
}

// for login errors; show login modal until captcha solved
if (isset($_GET['modal']) && $_GET['modal'] === 'true') {
    $view->loginError = $view->loginError ?? "Please solve the Captcha and try again.";
}
```

paginationcontroller.php

```php
<?php
require_once('Models/FacilityDataSet.php');
require_once("Models/Paginator.php");

// Default Filters
$filters = [
    'category' => $_GET['category'] ?? '1', // Default category
    'term' => $_GET['term'] ?? '', // Default term
    'sort' => $_GET['sort'] ?? '1', // Default sort
    'dir' => $_GET['dir'] ?? 'asc', // Default direction
    'page' => $_GET['page'] ?? 0 // Default to first page
];

// If no query parameters exist (initial page load), redirect to set default ones
if (empty($_GET)) {
    redirectWithFilters($filters);
}

// Set row limit
$rowLimit = 7;
// create dataset object
$facilityDataSet = new FacilityDataSet();

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    /**
     * Unfortunately, ZERO time to fix this, too complex.
     */
    if(isset($_POST['updateButton'])) {
        $data = [
            'id' => $_POST['idUpdate'],
            'title' => $_POST['titlUpdate'],
            'category' => $_POST['cateUpdate'],
            'description' => $_POST['descUpdate'],
            'houseNumber' => $_POST['hnumUpdate'],
            'streetName' => $_POST['strtUpdate'],
            'county' => $_POST['cntyUpdate'],
            'town' => $_POST['townUpdate'],
            'postcode' => $_POST['postUpdate'],
            'lng' => $_POST['lngUpdate'],
            'lat' => $_POST['latUpdate'],
            'contributor' => $_POST['contUpdate'],
        ];
        $facilityDataSet->addFacility($data);
    }
    if(isset($_POST['createButton'])) {
        $data = [
            'title' => $_POST['titlCreate'],
            'category' => $_POST['cateCreate'],
            'description' => $_POST['descCreate'],
            'houseNumber' => $_POST['hnumCreate'],
            'streetName' => $_POST['strtCreate'],
            'county' => $_POST['cntyCreate'],
            'town' => $_POST['townCreate'],
            'postcode' => $_POST['postCreate'],
            'contributor' => $_POST['contCreate'],
        ];
        $facilityDataSet->addFacility($data);
    }
    // passes id to delete facility
    if (isset($_POST['deleteButton'])) {
        $facilityDataSet->deleteFacility($_POST['id']);
    }
    // Check if filters/sorting changed
    $filtersChanged = (
        $filters['category'] !== ($_POST['filterCat'] ?? $filters['category']) ||
        $filters['term'] !== ($_POST['filter'] ?? $filters['term']) ||
        $filters['sort'] !== ($_POST['sort'] ?? $filters['sort']) ||
        $filters['dir'] !== ($_POST['dir'] ?? $filters['dir'])
    );

    // load from post if exists and sanitise, otherwise use defaults
    $filters['category'] = filter_input(INPUT_POST, 'filterCat', FILTER_SANITIZE_FULL_SPECIAL_CHARS) ?? $filters['category'];
    $filters['term'] = filter_input(INPUT_POST, 'filter', FILTER_SANITIZE_FULL_SPECIAL_CHARS) ?? $filters['term'];
```

```
    $filters['sort'] = filter_input(INPUT_POST, 'sort', FILTER_SANITIZE_FULL_SPECIAL_CHARS) ?? $filters['sort'];
    $filters['dir'] = filter_input(INPUT_POST, 'dir', FILTER_SANITIZE_FULL_SPECIAL_CHARS) ?? $filters['dir'];

    // Reset page if filters changed
    $filters['page'] = $filtersChanged ? 0 : $_POST['paginationButton'] ?? $filters['page'];
    redirectWithFilters($filters);
}

// fetch page data from database
$view->allPageData = $facilityDataSet->fetchAll(
    ['category' => $filters['category'], 'term' => $filters['term']],
    ['sort' => $filters['sort'], 'dir' => $filters['dir']]
);

// set total facility count to view
$view->totalResults = $view->allPageData['count'];
// create paginator object
$view->paginator = new Paginator($rowLimit, $view->allPageData);
// assign page number to view
$view->pageNumber = $view->paginator->getPageFromUri();
// get current page
$view->pageData = $view->paginator->getPage($view->pageNumber);
// Send result count to view in format "showing x of y results"
$view->dbMessage = $view->paginator->countPageResults($view->pageNumber) == 0
    ? "No results"
    : "Showing " . $view->paginator->countPageResults($view->pageNumber) . " of " . $view->totalResults .  " result(s)";

// Redirect function, adds header parameters
function redirectWithFilters($filters) {
    // Ensure no unintended keys are passed
    $allowedKeys = ['category', 'term', 'sort', 'dir', 'page'];
    $filters = array_filter($filters, function($key) use ($allowedKeys) {
        return in_array($key, $allowedKeys);
    }, ARRAY_FILTER_USE_KEY);

    $queryString = http_build_query($filters);
    header("Location: ?" . $queryString);
    exit;
}
```

Views

Index.phtml

```
<?php require('template/header.phtml') ?>
    <div class="row">
        <div class="col-5 me-auto">
            <p><?php echo $view->dbMessage; ?></p>
        </div>
        <form class="col-auto">
            <?php require_once('template/createModal.phtml') ?>
        </form>
    </div>

<div class="row">
    <div class="container-fluid p-3" id="facilityContent">
        <table class="table table-bordered">
            <thead>
                <tr>
                    <th>Facility ID</th>
                    <th>Title</th>
                    <th>Category</th>
                    <th>Status</th>
                    <th>Description</th>
                    <th>Address</th>
                    <th>Postcode</th>
                    <th>Lat/Long</th>
                    <th>Contributor</th>
                    <?php if($view->user->getAccessLevel() == 1): ?>
                    <th>Actions</th>
                    <?php endif; ?>
                </tr>
            </thead>
            <tbody>
            <?php foreach ($view->pageData as $facilityData): ?>
                <tr>
                    <td><?= htmlspecialchars($facilityData->getId() ?? 'N/A') ?></td>
                    <td><?= htmlspecialchars($facilityData->getTitle() ?? 'N/A') ?></td>
```

```
                <td><?= htmlspecialchars($facilityData->getCategory() ?? 'N/A') ?></td>
                <td><?= htmlspecialchars($facilityData->getStatus() ?? 'N/A') ?></td>
                <td><?= htmlspecialchars($facilityData->getDescription() ?? 'N/A') ?></td>
                <td><?= htmlspecialchars(trim(($facilityData->getHouseNumber() ?? '') . ' ' .
                    ($facilityData->getStreetName() ?? '') . ' ' .
                    ($facilityData->getCounty() ?? '') . ' ' .
                    ($facilityData->getTown() ?? ''))) ?></td>
                <td><?= htmlspecialchars($facilityData->getPostcode() ?? 'N/A') ?></td>
                <td><?= htmlspecialchars(($facilityData->getLat() ?? 'N/A') . ', ' .
                    ($facilityData->getLng() ?? 'N/A')) ?></td>
                <td><?= htmlspecialchars($facilityData->getContributor() ?? 'N/A') ?></td>
                <?php if($view->user->getAccessLevel() == 1): ?>
                  <td class="btn-group">
                    <?php require("template/updateModal.phtml") ?>
                    <?php require("template/deleteModal.phtml") ?>
                  </td>
                <?php endif; ?>
              </tr>
            <?php endforeach; ?>
            </tbody>
          </table>
        </div>


<?php require('template/footer.phtml') ?>
```

Views – Templates

CreateModal.phtml

```
<button type="button" class="col btn bg-primary btn-outline-primary text-light" data-bs-toggle="modal" data-bs-target="#createModal">
  <span class="bi bi-pen-fill"></span>
</button>
<div class="modal fade" id="createModal" tabindex="-1" aria-labelledby="updateModalLabel" aria-hidden="true">
  <div class="modal-dialog" role="document">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title" id="updateModalLabel">Add Facility</h5>
        <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
      </div>
      <div class="modal-body">
        <form method="post" action="<?php echo htmlspecialchars($_SERVER['PHP_SELF']); ?>">
          <form class="form-inline" method="post" action="<?php echo $_SERVER['PHP_SELF']; ?> ">
            <input name="titlCreate" class="form-control rounded mb-2" placeholder="Title">
            <input name="cateCreate" class="form-control rounded mb-2" placeholder="Category">
            <input name="descCreate" class="form-control rounded mb-2" placeholder="Description">
            <input name="hnumCreate" class="form-control rounded mb-2" placeholder="House Number">
            <input name="strtCreate" class="form-control rounded mb-2" placeholder="Street Name">
            <input name="cntyCreate" class="form-control rounded mb-2" placeholder="County">
            <input name="townCreate" class="form-control rounded mb-2" placeholder="Town">
            <input name="postCreate" class="form-control rounded mb-2" placeholder="Postcode">
            <input name="contCreate" class="form-control rounded mb-2" placeholder="Contributor">
          </form>
          <button type="submit" class="btn bg-primary btn-outline-primary text-light" name="createButton">Add</button>
        </form>
      </div>
      <div class="modal-footer">
        <button type="button" class="btn btn-warning" data-bs-dismiss="modal">Close</button>
      </div>
    </div>
  </div>
</div>
```

deleteModal.phtml

```
<button type="button" class="col btn bg-danger btn-outline-danger text-light" data-bs-toggle="modal" data-bs-target="#deleteModal">
  <span class="bi bi-trash-fill">
</button>
<div class="modal fade" id="deleteModal" tabindex="-1" aria-labelledby="deleteModalLabel" aria-hidden="true">
  <div class="modal-dialog" role="document">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title" id="deleteModalLabel">Delete Facility Record</h5>
        <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
```

```
        </div>
        <div class="modal-body">
          <form method="post" action="">
            <button type="submit" value="delete" class="btn bg-danger btn-outline-danger text-light" name="deleteButton"">Yes</button>
            <input type="hidden" name="id" value="<?= $facilityData->getId()?>">
            <button type="button" class="btn btn-outline primary btn-primary" data-bs-dismiss="modal">No</button>
          </form>
        </div>
      </div>
    </div>
</div>
```

footer.phtml

```
  </div>
  <div class="site-footer fixed-bottom mt-auto">
    <div class="col-auto">
      <?php require_once('pagination.phtml'); ?>
    </div>
    <div class="row">
      <div id="footer" class="col-xs-12">
        <p class="m-0">George Wilkinson @2024</p>
        <p class="m-0">Powered by Bootstrap</p>
      </div>
    </div>
  </div>
<!-- Bootstrap core JavaScript
================================================== -->
<!-- Placed at the end of the document so the pages load faster -->
<!-- script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.0/jquery.min.js"></script -->
<script src="/js/bootstrap.min.js"></script>
</body>
</html>
```

header.phtml

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <meta name="description" content="">
  <meta name="author" content="">
  <link rel="icon" type="image/x-icon" href="/images/ecoBuddy_x32.png"
  <!-- Bootstrap core CSS -->
  <link href="/css/bootstrap.css" rel="stylesheet">
  <!-- Bootstrap theme -->
  <link href="/css/bootstrap-theme.css" rel="stylesheet">
  <link href="/css/my-style.css" rel="stylesheet">
  <!-- Bootstrap Icons -->
  <link href="/css/bootstrap-icons.css" rel="stylesheet">

  <title>Ecobuddy - <?php echo $view->pageTitle; ?></title>
</head>

<nav class="navbar navbar-expand-lg p-0 m-2 border rounded-2">
  <div class="container-fluid">
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarTogglerDemo03" aria-controls="navbarTogglerDemo03" aria-
expanded="false" aria-label="Toggle navigation">
      <span class="navbar-toggler-icon"></span>
    </button>
    <a class="navbar-brand" href="/index.php"><img id="navIcon" class="img-thumbnail bg-transparent border-3 border-success border-opacity-25 rounded my-1 me-2"
height="64px" width="64px" src="/images/ecoBuddy_x64.png" alt=""/><span class="pt-5 mb-auto">Ecobuddy</span></a>
    <div class="collapse navbar-collapse" id="navbarTogglerDemo03">
      <ul class="navbar-nav me-auto mb-2 mb-lg-0">
      </ul>
      <form class="row m-0 me-2 align-content-center align-content-center align-items-center" role="search" action="" method="POST">
        <div class="col">
          <div class="form-floating input-group">
            <select name="sort" class="form-select border-3 border-success-subtle" id="sort">
              <option value="1" <?php if(isset($_GET['sort']) && $_GET['sort'] == '1') echo 'selected'; ?>>Title</option>
              <option value="2" <?php if(isset($_GET['sort']) && $_GET['sort'] == '2') echo 'selected'; ?>>Category</option>
              <option value="0" <?php if(isset($_GET['sort']) && $_GET['sort'] == '0') echo 'selected'; ?>>Status</option>
              <option value="3" <?php if(isset($_GET['sort']) && $_GET['sort'] == '3') echo 'selected'; ?>>Description</option>
              <option value="4" <?php if(isset($_GET['sort']) && $_GET['sort'] == '4') echo 'selected'; ?>>Street Name</option>
              <option value="5" <?php if(isset($_GET['sort']) && $_GET['sort'] == '5') echo 'selected'; ?>>County</option>
              <option value="6" <?php if(isset($_GET['sort']) && $_GET['sort'] == '6') echo 'selected'; ?>>Town</option>
```

```
            <option value="7" <?php if(isset($_GET['sort']) && $_GET['sort'] == '7') echo 'selected'; ?>>Postcode</option>
            <option value="8" <?php if(isset($_GET['sort']) && $_GET['sort'] == '8') echo 'selected'; ?>>Contributor</option>
          </select>
          <span class="form-floating input-group">
            <select class="form-select border-3 border-start-0 rounded-end border-success-subtle" name="dir" id="dir">
              <option value="asc" <?php if($_GET['dir'] == 'asc') echo 'selected'; ?>>Asc</option>
              <option value="desc" <?php if($_GET['dir'] == 'desc') echo 'selected'; ?>>Desc</option>
            </select>
            <label for="dir">Order</label>
          </span>
          <label for="sort">Sort By</label>
        </div>
      </div>
      <div class="col">
        <div class="form-floating input-group">
          <select name="filterCat" class="form-select border-3 border-success-subtle" id="filterCat">
            <option value="1" <?php if(isset($_GET['category']) && $_GET['category'] == '1') echo 'selected'; ?>>Title</option>
            <option value="2" <?php if(isset($_GET['category']) && $_GET['category'] == '2') echo 'selected'; ?>>Category</option>
            <option value="0" <?php if(isset($_GET['category']) && $_GET['category'] == '0') echo 'selected'; ?>>Status</option>
            <option value="3" <?php if(isset($_GET['category']) && $_GET['category'] == '3') echo 'selected'; ?>>Description</option>
            <option value="4" <?php if(isset($_GET['category']) && $_GET['category'] == '4') echo 'selected'; ?>>Street Name</option>
            <option value="5" <?php if(isset($_GET['category']) && $_GET['category'] == '5') echo 'selected'; ?>>County</option>
            <option value="6" <?php if(isset($_GET['category']) && $_GET['category'] == '6') echo 'selected'; ?>>Town</option>
            <option value="7" <?php if(isset($_GET['category']) && $_GET['category'] == '7') echo 'selected'; ?>>Postcode</option>
            <option value="8" <?php if(isset($_GET['category']) && $_GET['category'] == '8') echo 'selected'; ?>>Contributor</option>
          </select>
          <span class="input-group-text bi bi-filter-circle bg-success-subtle border-0 rounded-end" id="filterCat"></span>
          <label for="filterCat">Column Filter</label>
        </div>
      </div>
      <div class="col">
        <div class="form-floating input-group">
          <label for="search"></label>
          <input placeholder="<?php if(isset($_GET['filter'])) echo $_GET['filter']; ?>" class="form-control border-3 border-success-subtle" id="search" type="search"
name="filter" aria-label="Search">
          <span class="input-group-text bg-success-subtle border-0 rounded-end" id="search">
            <button class="btn bg-light bg-success-subtle" type="submit"><span class="bi bi-search"></span></button>
          </span>
        </div>
      </div>
    </form>
    <div class="me-2 ms-2">
      <div class="col-sm" id="loginStatus">
        <?php

        if(!$view->user->isLoggedIn()) {
          require_once('Views/template/loginModal.phtml');
        }
        if($view->user->isLoggedIn()) {
          require_once('Views/template/logoutButton.phtml');
        }
        ?>
      </div>
    </div>
  </div>
</nav>
<body role="document">

<div class="main container-fluid">
  <div class="col" id="content">
```

loginError.phtml

```
<span class="ms-5 me-5 row alert alert-danger" role="alert"><?= $view->loginError ?></span>
<div class="row captcha-container">
  <!-- CAPTCHA Display -->
  <div class="form-floating mb-3 col">
    <input type="text" class="form-control" id="captchaCode" value="<?php
    // Generate a simple 5-character CAPTCHA
    $captcha = substr(str_shuffle("ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"), 0, 5);
    echo $captcha;
    ?>" readonly>
    <label for="captchaCode">CAPTCHA Code</label>
```

```
  </div>

  <!-- CAPTCHA Input -->
  <div class="form-floating mb-3 col">
    <input type="text" class="form-control" id="captchaInput" name="captchaInput" placeholder="Enter CAPTCHA" required>
    <label for="captchaInput">Enter CAPTCHA</label>
  </div>
</div>
```

loginModal.phtml

```
<button type="button" class="btn bg-primary btn-outline-primary text-light m-auto" data-bs-toggle="modal"
      data-bs-target="#loginModal">
  Login
</button>
<?= isset($view->loginError) ? '<div class="modal-backdrop fade show"></div>' : '' ?>
<div class="modal fade <?= isset($view->loginError) ? 'show' : '' ?>" id="loginModal" tabindex="-1"
    aria-labelledby="loginModalLabel" aria-hidden="<?= isset($view->loginError) ? 'false' : 'true' ?>"
    style="<?= isset($view->loginError) ? 'display: block;' : '' ?>">
  <div class="modal-dialog" role="document">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title" id="loginModalLabel">Login</h5>
        <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
      </div>
      <div class="modal-body">
        <form method="post" action="<?php echo htmlspecialchars($_SERVER['PHP_SELF']); ?>">
          <div class="mb-3">
            <label for="username" class="form-label">Username</label>
            <input type="text" class="form-control" id="username" name="username" placeholder="Username"
                   required>
          </div>
          <div class="mb-3">
            <label for="password" class="form-label">Password</label>
            <input type="password" class="form-control" id="password" name="password" placeholder="Password"
                   required>
          </div>
          <?php if (isset($view->loginError)) { include('Views/template/loginError.phtml');} ?>
          <button type="submit" class="btn bg-primary btn-outline-primary text-light" name="loginButton">Login
          </button>
        </form>
      </div>
      <div class="modal-footer">
        <a href="/index.php <?php unset($_GET['modal'])?>" type="button" class="btn btn-warning btn-outline-warning text-light" data-bs-dismiss="modal">
          Close
        </a>
      </div>
    </div>
  </div>
</div>
```

logoutButton.phtml

```
<form class="form-floating my-auto" method="post" action="<?php echo $_SERVER['PHP_SELF']; ?> ">
  <?php echo "<p class='text-center bg-light border-0 rounded mb-1' style='color: black;'>" . $user->getUsername() . "<span class='bi bi-person-fill'></span></p>"?>
  <button class="btn bg-danger btn-outline-danger text-light" type="submit" name="logoutButton">Logout</button>
</form>
```

pagination.phtml

```
<div>
  <div class="row mb-2">
    <!-- Form for Pagination -->
    <div id="paginationButtons" class="col-auto m-auto btn-group">

      <?php
      $param = $_GET;
      unset($param['page']); // Remove the page parameter to avoid duping
      function buildUrl($page, $param): string
      {
        $param['page'] = $page;
        return '?' . http_build_query($param);
      }
      ?>
      <!-- Start Button -->
      <a class="btn btn-outline-primary" href="<?= buildUrl(0, $param) ?>0" <?= $view->pageNumber <= 0 ? 'disabled' : '' ?>><i class="bi bi-chevron-double-left"></i> Start</a>
      <!-- Back Button -->
```

```
        <a class="btn btn-outline-primary" href="<?= buildUrl(max($view->pageNumber - 1, 0), $param)?> " <?= $view->pageNumber <= 0 ? 'disabled' : '' ?>><i class="bi bi-chevron-left"></i> Back</a>
        <!-- Dynamic Page Buttons -->
        <?php
        $totalPages = $view->paginator->getTotalPages();
        for ($i = $view->pageNumber - 2; $i <= $view->pageNumber + 2; $i++) {
           if ($i >= 0 && $i < $totalPages): ?>
              <a href="<?= buildUrl($i, $param) ?>"
                    class="btn <?= $i === $view->pageNumber ? 'btn-dark' : 'btn-outline-primary' ?>"
                 <?= $i === $view->pageNumber ? 'disabled' : '' ?>>
                 <?= $i + 1 ?>
              </a>
           <?php endif;
        } ?>
        <!-- Forward Button -->
        <a class="btn btn-outline-primary" href="<?=buildUrl(min($view->pageNumber + 1, $totalPages), $param)?>" <?= $view->pageNumber >= $totalPages - 1 ? 'disabled' : '' ?>>Forward <i class="bi bi-chevron-right"></i></a>
        <!-- End Button -->
        <a class="btn btn-outline-primary" href="<?= buildUrl($totalPages - 1, $param) ?>"<?= $view->pageNumber >= $totalPages - 1 ? 'disabled' : '' ?>>End <i class="bi bi-chevron-double-right"></i></a>
      </div>
   </div>
</div>
```

updateModal.phtml

```
<button type="button" class="col btn bg-primary btn-outline-primary text-light" data-bs-toggle="modal" data-bs-target="#updateModal">
   <span class="bi bi-pen-fill"></span>
</button>

<div class="modal fade" id="updateModal" tabindex="-1" aria-labelledby="updateModalLabel" aria-hidden="true">
   <div class="modal-dialog" role="document">
      <div class="modal-content">
         <div class="modal-header">
            <h5 class="modal-title" id="updateModalLabel">Update Facility</h5>
            <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
         </div>
         <div class="modal-body">
            <form method="post" action="<?php echo htmlspecialchars($_SERVER['PHP_SELF']); ?>">
               <input name="titlUpdate" class="form-control rounded mb-2" value="<?= $facilityData->getTitle() ?? '' ?>" placeholder="Title">
               <input name="cateUpdate" class="form-control rounded mb-2" value="<?= $facilityData->getCategory() ?? '' ?>" placeholder="Category">
               <input name="descUpdate" class="form-control rounded mb-2" value="<?= $facilityData->getDescription() ?? '' ?>" placeholder="Description">
               <input name="hnumUpdate" class="form-control rounded mb-2" value="<?= $facilityData->getHouseNumber() ?? '' ?>" placeholder="House Number">
               <input name="strtUpdate" class="form-control rounded mb-2" value="<?= $facilityData->getStreetName() ?? '' ?>" placeholder="Street Name">
               <input name="cntyUpdate" class="form-control rounded mb-2" value="<?= $facilityData->getCounty() ?? '' ?>" placeholder="County">
               <input name="townUpdate" class="form-control rounded mb-2" value="<?= $facilityData->getTown() ?? '' ?>" placeholder="Town">
               <input name="postUpdate" class="form-control rounded mb-2" value="<?= $facilityData->getPostcode() ?? '' ?>" placeholder="Postcode">
               <input name="contUpdate" class="form-control rounded mb-2" value="<?= $facilityData->getContributor() ?? '' ?>" placeholder="Contributor">
               <button type="submit" class="btn bg-primary btn-outline-primary text-light" name="updateButton">Update</button>
               <input type="hidden" name="idUpdate" value="<?= $facilityData->getId()?>">
            </form>
         </div>
         <div class="modal-footer">
            <button type="button" class="btn btn-warning" data-bs-dismiss="modal">Close</button>
         </div>
      </div>
   </div>
</div>
```

Custom Stylesheets

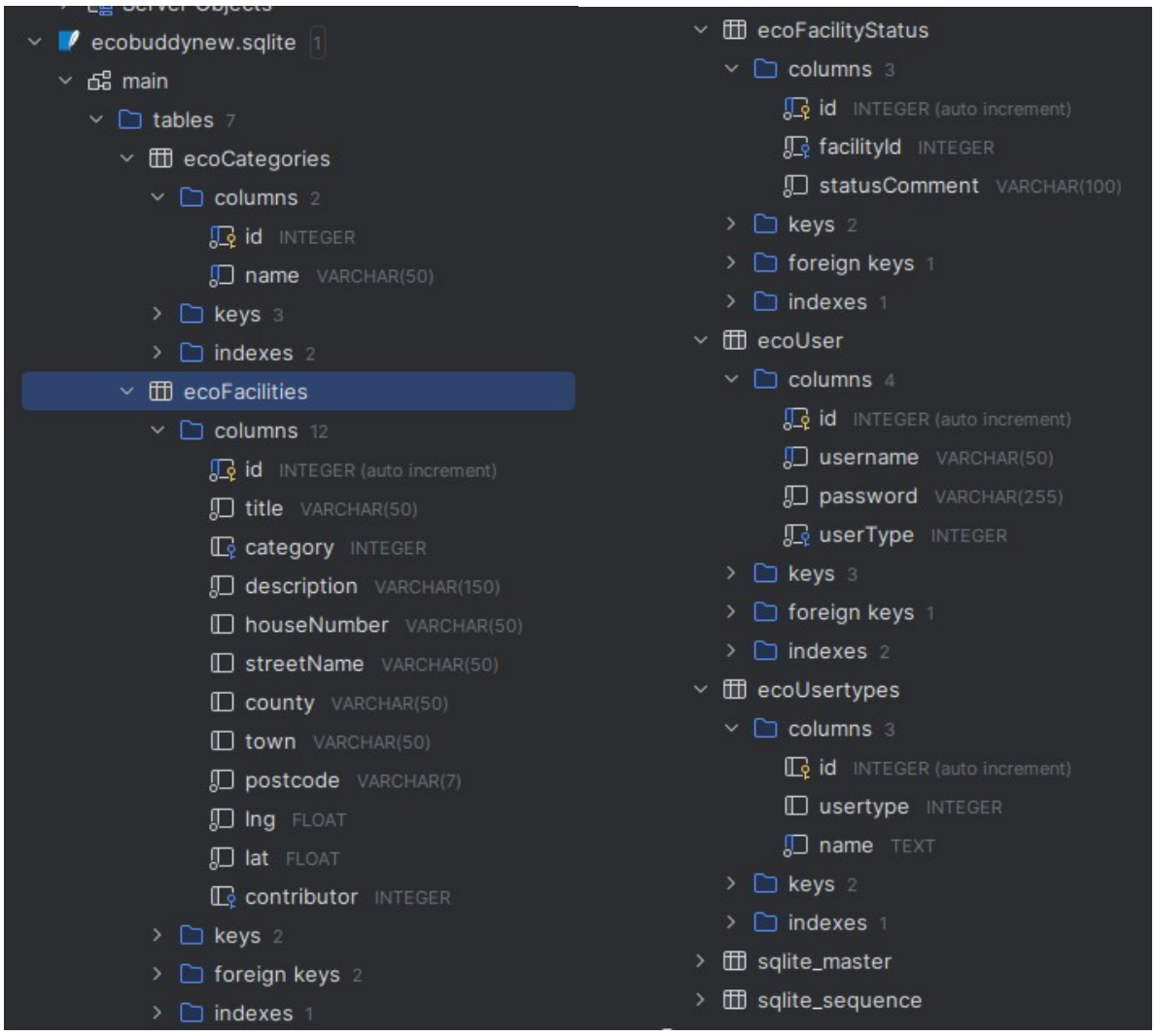
my-style.css

```
nav, #loginStatus, #filters {
   background-color: #3cc471;
   color: #111
}
#content.full-height {
   /*height: calc(100vh - 413px);*/
   flex: 1 0 auto;
}
.main {
   display: flex;
   flex-direction: column;
   min-height: 100vh;
}
```

```css
.facilityContent {
    overflow-y: auto;
}

#title {
    margin-top: 12px;
    background-color: #fff;
    color: #000;

}
#menu {
    border-top: solid 6px #000;
    background-color: #fff;
    color: #fff;
    height: 400px;
}

#menu a {
    /*background-color: #f00;*/
    color: #fff;
    text-decoration: none;
    display: block;
}
#menu a:hover {
    /*background-color: #f00;*/
    color: #ddd;
    text-decoration:underline;
    display: block;
}

#content {
    background-color: #fff;
    /*border-top: solid 6px #f00;*/

}

#footer {
    margin-top: 20px;
    text-align: center;
    background-color: #bbb;
    color: #111;
}

.modal {
    z-index: 1055
}

.modal-backdrop {
    z-index: 1040;
}
.site-footer {
    flex: 0 0 auto;
}
```

Database Records / Schema

Tables



EcoCategories

	id	name
1	1	Recycling Bins
2	2	e-Scooters
3	3	Bike Share Stations
4	4	Public EV Charging Stations
5	5	Battery Recycling Points
6	6	Community Compost Bins
7	7	Solar-Powered Benches
8	8	Green Roofs
9	9	Public Water Refill Stations
10	10	Waste Oil Collection Points
11	11	Book Swap Stations
12	12	Pollinator Gardens
13	13	E-Waste Collection Bins
14	14	Clothing Donation Bins
15	15	Community Tool Libraries

EcoFacilities

(way too many columns to display in a reasonable size, I apologise but it is readable with zoom)

	id	title	catego…	description	houseNum…	street…	coun…	town	postco…	lng	lat	contribut…
1	1	E-scooters at Salford Uni	2	Next to Maxwell Hall, a bunch of e-scooters	1	The Crescent	Greater Manchester	Salford	M5 4WT	-2.3467836	54.8764767	1
2	2	Recycling Bins at University Campus	1	Located outside the main library	10	Upper Brook Street	Greater Manchester	Manchester	M13 9PL	-2.2345	53.4669	2
3	3	Bike Share Station at Piccadilly Gardens	3	Easily accessible bike rentals	25	Piccadilly Gardens	Greater Manchester	Manchester	M1 1RG	-2.2335	53.4822	3
4	4	Public EV Charging at Deansgate	4	Charging points available at the parking garage	100	Deansgate	Greater Manchester	Manchester	M3 2GP	-2.2461	53.4785	1
5	5	Battery Recycling Point at St Peters Square	5	Batteries can be disposed here safely	10	St Peters Square	Greater Manchester	Manchester	M2 5GP	-2.2474	53.4784	2
6	6	Community Compost Bins at Whitworth Park	6	Bins for composting available in the park	30	Grafton Street	Greater Manchester	Manchester	M14 5SL	-2.2293	53.4651	1
7	7	Solar-Powered Benches at Sackville Gardens	7	Benches with solar panels for charging devices	5	Sackville Street	Greater Manchester	Manchester	M1 3HG	-2.235	53.4731	3
8	8	Public Water Refill Station at Northern Quarter	9	Refill station located at the park	15	Tib Street	Greater Manchester	Manchester	M4 1NB	-2.236	53.485	2
9	9	Waste Oil Collection at Ancoats	10	Dispose of used cooking oil responsibly	20	Cutting Room Square	Greater Manchester	Manchester	M4 6BU	-2.2337	53.4791	1
10	10	Community Tool Library at Longsight	15	Borrow gardening and DIY tools	12	Stockport Road	Greater Manchester	Manchester	M13 0XR	-2.2173	53.4551	2
11	11	Community Bike Hub	3	Centrally located bike rentals for locals.	25	Oxford Street	Greater Manchester	Manchester	M1 6FQ	-2.2409	53.4795	2
12	12	Solar-Powered Water Fountain	9	Eco-friendly water fountain powered by solar energy.	10	Deansgate	Greater Manchester	Manchester	M3 2RP	-2.2502	53.4767	1
13	13	E-Waste Collection Point	13	Drop-off point for recycling old electronics.	5	Portland Street	Greater Manchester	Manchester	M1 6DF	-2.2382	53.4784	3
14	14	Battery Recycling Station	5	Secure drop-off for household batteries.	15	King Street	Greater Manchester	Manchester	M2 6EJ	-2.2485	53.4774	2
15	15	Urban Green Space	8	A small park area with native plants and seating.	40	Cross Street	Greater Manchester	Manchester	M2 4FW	-2.2403	53.4791	1
16	16	Community Composting Site	6	A local site for composting organic waste.	12	Albert Square	Greater Manchester	Manchester	M2 5DB	-2.2427	53.4811	3
17	17	Public EV Charging Hub	4	Charging station for electric vehicles in the city center.	30	Chepstow Street	Greater Manchester	Manchester	M1 5FW	-2.2374	53.4741	1
18	18	Book Exchange Kiosk	11	A small kiosk for exchanging books in the community.	8	St. Mary's Gate	Greater Manchester	Manchester	M3 2WQ	-2.2518	53.4762	2
19	19	Clothing Donation Drop Box	14	Drop-off box for donating clothes to those in need.	100	Grafton Street	Greater Manchester	Manchester	M13 9WT	-2.2356	53.4654	3
20	20	Pollinator Garden	12	A garden designed to attract and support local pollinators.	50	Blossom Street	Greater Manchester	Manchester	M12 6BG	-2.2275	53.4728	1
21	21	Facility 1	1	Description for facility 1	8799	Street 26	County 15	Town 29	AB45 4CD	-177.798557	-84.694977	7
22	22	Facility 2	5	Description for facility 2	6453	Street 48	County 5	Town 38	AB38 2CD	-129.834741	69.28418	33
23	23	Facility 3	8	Description for facility 3	9971	Street 3	County 14	Town 6	AB98 4CD	-18.468364	-4.381366	45
24	24	Facility 4	4	Description for facility 4	9369	Street 18	County 12	Town 4	AB68 2CD	104.423251	12.354898	37
25	25	Facility 5	10	Description for facility 5	1462	Street 6	County 16	Town 38	AB41 1CD	111.886973	16.968499	13
26	26	Facility 6	8	Description for facility 6	9328	Street 6	County 20	Town 12	AB37 2CD	-78.180274	-83.953428	39
27	27	Facility 7	7	Description for facility 7	1757	Street 17	County 11	Town 8	AB12 9CD	-54.371941	66.351242	55
28	28	Facility 8	2	Description for facility 8	1928	Street 28	County 18	Town 13	AB98 8CD	-61.892343	14.535855	96
29	29	Facility 9	4	Description for facility 9	2326	Street 37	County 9	Town 28	AB83 8CD	42.804902	-64.898675	43
30	30	Facility 10	2	Description for facility 10	4543	Street 5	County 6	Town 21	AB68 2CD	121.696471	59.845017	26

EcoFacilityStatus

	id	facilityId	statusComment
1	1	13	Bin is full
2	2	12	Not working
3	3	4	Often busy
4	4	4	One charger not working
5	5	1	Always lots available
6	6	1	Great way to get around
7	7	7	Great to charge your phone but bring a cable

EcoUser

WHERE ORDER BY

	id	username	password	userType
1	1	Lee	6a9ec21e4d9eb23f080f4332d5aae8696dacceecd9257f66b91f01e258bf3262	2
2	2	Admin	4a7bfbdfa0e97747eda19e5f7b217f3c9844b263a96b580028273b18b39e1dba	1
3	3	Benny	98f58a19f1fdf4241d0562fb41b09892a008e54befd7fa81345bfa9930dc68aa	2

EcoUserTypes

	id	usertype	name
1	1	1	Manager
2	2	2	User