課題確認アプリ-TaktManager Gen2-

川上 拓飛　雲岡 広晃
白石 大智　瀧﨑 掌俐

１．研究概要

PHP を用いて､Web アプリケーション開発の技法を習得し、出題された課題の確認をクラスで共有できるプラットフォームを開発した。

２．研究の具体的内容

現在全世界の Web サイトの約 80%で使用されている言語である PHP を使用し、Web アプリケーション開発を行った。

開発環境には Visual Studio Code、XAMPP、Composer を用いた。Visual Studio Code は Microsoft が開発した強力なエディタであり、Windows、Linux、macOS、Web で使用することができる。シェアは 50%を超えている。コード保管の機能や拡張機能が豊富で使いやすいため採用した。

XAMPP は Apache Friends が開発し、Web アプリケーションの実行に必要なフリーソフトウェアをパッケージとしてまとめたものである。PHP と MySQL の環境構築が簡単にできるので採用した。

Composer は、PHP 向けのソフトウェアおよび必要なライブラリの依存関係を管理する標準形式を提供する管理システムである。WebPush を利用するために導入した。

開発当初のプロトタイプでは図１のようなデザインを作成し、PC 版・モバイル版ともに同じデザインで、シンプルなデザインと機能であった。そのため、PC 版では余白が大きく、ポストする際ポストウィンドウを開く必要があり、使いやすい UI とは言えなかった。またログイン画面を含む全体の UI に統一感を持たせるため、一から画面設計をやり直し、赤を基調とした配色やアクセスしやすいサイドメニューバーを導入した。

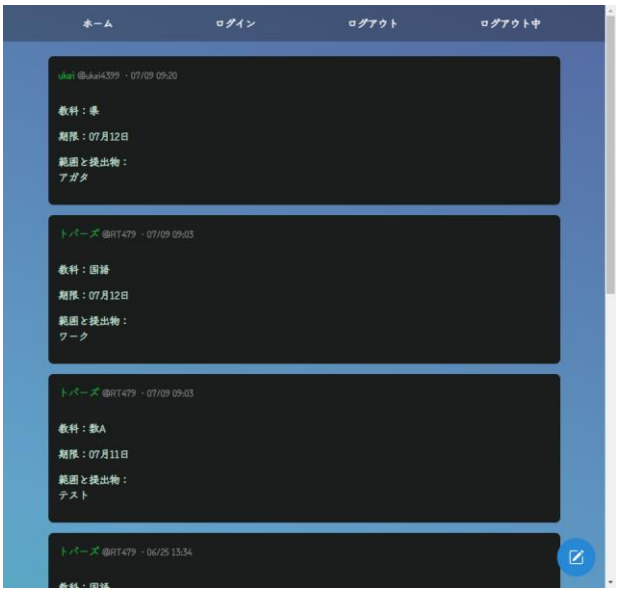

図１　プロトタイプの画面デザイン

図２が PC 版である。PC 版での最適化を進め、トップ画面（共有タスク）では左側にサイドメニューバー、中央にポスト内容、右側に投稿フォームを配置し、確認・ポストが素早くできるように変更した。

図２　ライトモードの画面

図３はダークモードに変更した際の画面である、好きなテーマに切り替える機能も搭載した。

図３　ダークモードの画面

図４、５がモバイル版である。モバイル版では、ポストボタンを引き続き採用し、操作性を担保させた。またボトムナビゲーションを配置し、主要な機能に素早くアクセスできる設計にした。

図４　入力フォーム

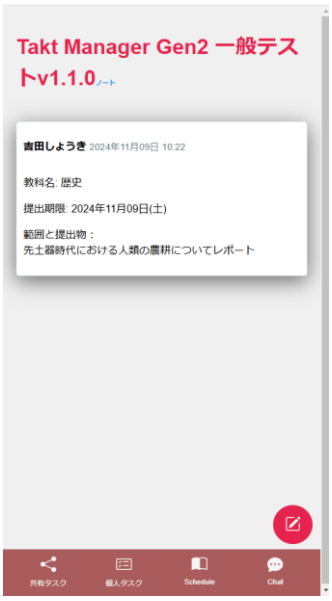

図５　モバイル版の画面デザイン

図６は主機能（クラス全体で確認できる共有タスク、個人タスク）の１つである共有タスクのデザインである。共有タスクは、提出物やテストスケジュールの確認・ポストが可能である。

図６　共有タスク画面

図７は個人タスクのデザイン画面で、個人それぞれがやることをメモできるスペースとなっている。重要度を３段階（S 級・A 級・B 級）で色分けすることができる。

図７　個人タスク画面

図８は主機能以外にもクイックチャットを作成し、課題などについて TaktManager 内で質問できるようにした。課題やテストに関することが１つのアプリ内で完結するような設計とし、クイックチャットはアカウントを作成していれば誰でもメッセージ送信ができるオープンチャットタイプとした。

初期設計段階では、３D アニメーションを取り入れ視覚的に面白いアプリにすることを考えていたが、読み込みが遅くなることによって素早い確認ができない可能性があり無視できない問題であったため、３D アニメーションは作成しないこととした。

図８　クイックチャット画面

通知機能は、今回の開発の中で一番時間を要した機能である。メールアドレスを必要としない WebPush を利用した通知機能を作成したのだが、参考にできるサイトが少なく実装が難しかった。その過程で PWA にも対応し、ネイティブアプリのようにインストールして、PC ではタスクバーから起動したり、モバイルではホーム画面のアイコンをタップして起動できるようにした。OS の関係上 WebPush に対応している Windows、Android、ChromeOS のみ通知機能に対応し、iOS は通知機能に対応させることができなかった。（図９、１０）は Windows の通知画面）

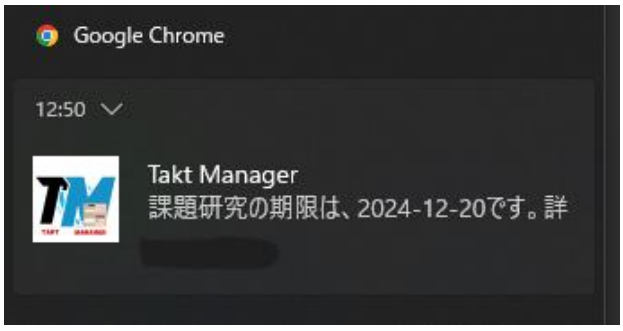

図９　通知バー

図 10　実際の通知の動作

図 11 はアカウント登録機能の処理の抜粋であり、トークンを生成し、CSRF 攻撃への対応を行った。また、パスワードに条件を設け、緩いパスワードが登録されないように if 文で制御した。

```
<?php
session_start();
require_once 'db_connection.php';

// CSRF トークンの生成と検証
if (empty($_SESSION['csrf_token'])) {
    $_SESSION['csrf_token'] = bin2hex(random_bytes(32));
}
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // CSRF トークンの検証
    if(!hash_equals($_SESSION['csrf_token'], $_POST['csrf_token'])) {
        die('CSRF トークンが無効です。');
    }
    // 入力値の検証と無害化
    $username=filter_input(INPUT_POST, 'username', FILTER_SANITIZE_FULL_SPECIAL_CHARS);
    $user_id = filter_input(INPUT_POST, 'user_id', FILTER_SANITIZE_FULL_SPECIAL_CHARS);
    $password = $_POST['password'];

    // 入力値の追加検証
    if (empty($username) || empty($user_id) || empty($password)) {
        $error = "すべてのフィールドを入力してください。";
    }elseif(strlen($password)<8||!preg_match("/[A-Z]/", $password) || !preg_match("/[a-z]/", $password)||!preg_match("/[0-9]/", $password)) {
        $error = "パスワードは 8 文字以上で大文字、小文字、数字を含める必要があります。";
    } else {
        $password_hash = password_hash($password, PASSWORD_DEFAULT);
        try {
            $stmt = $conn->prepare("INSERT INTO userinfo (username, user_id, password) VALUES
            (:username, :user_id, :password)");
            $stmt->bindParam(':username', $username, PDO::PARAM_STR);
            $stmt->bindParam(':user_id', $user_id, PDO::PARAM_STR);
            $stmt->bindParam(':password', $password_hash, PDO::PARAM_STR);
            $stmt->execute();
            $_SESSION['message'] = "登録が完了しました。さぁ、ログインしましょう！";
            header("Location: login.php");
            exit();
        } catch(PDOException $e) {
            $error = "既に他のユーザーがこの ID を使用しています。";
        }
    }
}
?>
```

図 11　アカウント登録機能のコード（抜粋）

図 12 はログイン機能の処理の抜粋であり、CSRF 攻撃対策やセッションの有効期限等を指定している。

```
<?php
session_start();
require_once 'db_connection.php';

// CSRF 対策: トークンの生成と検証
if (!isset($_SESSION['csrf_token'])) {
    $_SESSION['csrf_token'] = bin2hex(random_bytes(32));
}
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // CSRF 対策: トークンの検証
    if(!hash_equals($_SESSION['csrf_token'], $_POST['csrf_token'])) {
        die('CSRF token mismatch');
    }
    // ユーザー入力の安全な取得
    $user_id = trim($_POST['user_id'] ?? '');
    $password = $_POST['password'] ?? '';
    $remember = isset($_POST['remember']) ? $_POST['remember'] : '';

    if (empty($user_id) || empty($password)) {
        $error = "ユーザーIDとパスワードを入力してください。";
    } else {
        try {
            $stmt = $conn->prepare("SELECT * FROM userinfo WHERE user_id = :user_id");
            $stmt->bindParam(':user_id', $user_id, PDO::PARAM_STR);
            $stmt->execute();
            $user=$stmt->fetch(PDO::FETCH_ASSOC);
          if($user&&password_verify($password, $user['password'])) {
```

```
if($user&&password_verify($password, $user['password'])) {
        $_SESSION['user_id']=$user['user_id'];
        $_SESSION['username']=htmlspecialchars($user['username'],
        ENT_QUOTES, 'UTF-8');
if ($remember == 'on') {
// セッションの有効期限を 30 日に設定
        ini_set('session.gc_maxlifetime', 30 * 24 * 60 * 60);
        session_set_cookie_params(30 * 24 * 60 * 60);
// ログイン情報を暗号化してクッキーに保存
        $token= bin2hex(random_bytes(32));
                setcookie('remember_token', $token, [
        'expires' => time() + 30 * 24 * 60 * 60,
        'path' => '/',
        'secure' => true,
        'httponly' => true,
        'samesite' => 'Strict'
]);
// トークンをデータベースに保存
        $stmt= $conn->prepare("UPDATE userinfo SET remember_token = :token
        WHERE user_id = :user_id");
        $stmt->execute(['token' => $token, 'user_id' => $user['user_id']]);
                }
        header("Location: index.php");
        exit();
} else {
        $error = "ユーザーID またはパスワードが正しくありません。";
            }
        } catch(PDOException $e) {
            $error = "エラーが発生しました。";
            error_log("ログインエラー");
        }
```

図 12　ログイン機能のコード（抜粋）

３．研究のまとめ

今回 PHP をはじめ、HTML・CSS・JavaScript・MySQL といった言語や管理システムを用いて Web アプリケーションの開発を行うことができ、今まで触れてくることがなかった技術を使う良い機会となった。開発スケジュールをしっかり考えていても予期せぬエラーや作業が発生してシステム開発を行うのは大変だと実感した。またより使いやすい UI やボタン配置を考え、利便性を高めることができた。「ユーザーが使いやすいものを作る」という精神はこれからも大切であり、役立つと思うので、しっかりと今回の経験を生かしていきたい。

４．参考文献

【PWA】ServiceWorker を登録して push 通知を実装する！
https://express-it.site/163/

PHP で会員登録＆ログイン機能を作成してみた
https://wagtechblog.com/programing/php-register-login.html

PHP + MySQL による簡単なブログサイトの構築
https://web-svr.com/PHP%E7%B7%A8/60.php