令和４年度卒業研究報告書

電子出席簿の作成

情報技術科　浅沼　麟　佐藤　琉希

指導教員　　石舘　勝好

目次

第1章 はじめに

私たちが電子席簿の作成に取り組もうとした理由は3つある.

まず1つ目は，卒業生が取り組んだ「PDF 自動生成とメール通知を利用した欠席連絡システムの開発」と「WEB データベースを用いた時間割アプリの作成」の卒業研究に興味を持ち，私たちでこの卒業研究について深められないかと考えた.

2つ目は，先生方が学生の出欠確認をする際に，紙の出席簿で記録するため手間がかかり集計時にミスが起きやすいといった話を聞き，出欠確認をデジタル化することにより教員の出欠確認にかかる負担や，無駄な時間を軽減できると考えた.

3つ目は，卒業研究を通してシステム開発をしてみたいと考えたから．授業では，システム開発に関して主にコーディングなどの下流工程の作業しか行わない．そのため，1から要件定義や設計などの上流工程から行うシステム開発を経験し，プログラミング技術，設計書作成スキルなどを身に着けたいと考えた.

これらの理由から，私たちは電子出席簿の作成を研究テーマにした.

第 2 章　研究概要

2.1 開発環境

システムの開発環境を以下の表 2.2.1 に示す.

表 2.2.1　システム開発環境

OS	Linux
使用言語	HTML5, CSS3, JavaScript, PHP7.0, SQL
データベース	MySQL
開発ソフトウェア	Visual Studio Code
Web サーバ	VartualBox

次に，システムの動作環境を以下の表 2.2.2 に示す．本研究では，欠席連絡システムと時間割システムの連携を図りたいため，校内サーバを使用した.

表 2.2.2　システム動作環境

OS	Windows10
使用言語	HTML, CSS, JavaScript, PHP7.3, SQL
データベース	MySQL
Web サーバ	情報技術科内部サーバ
使用サーバ	http://reciente.iwate-ti.ac.jp/rollbook

2.2 システム設計

図 2.2.1 は，電子出席簿システム設計のイメージ図である．

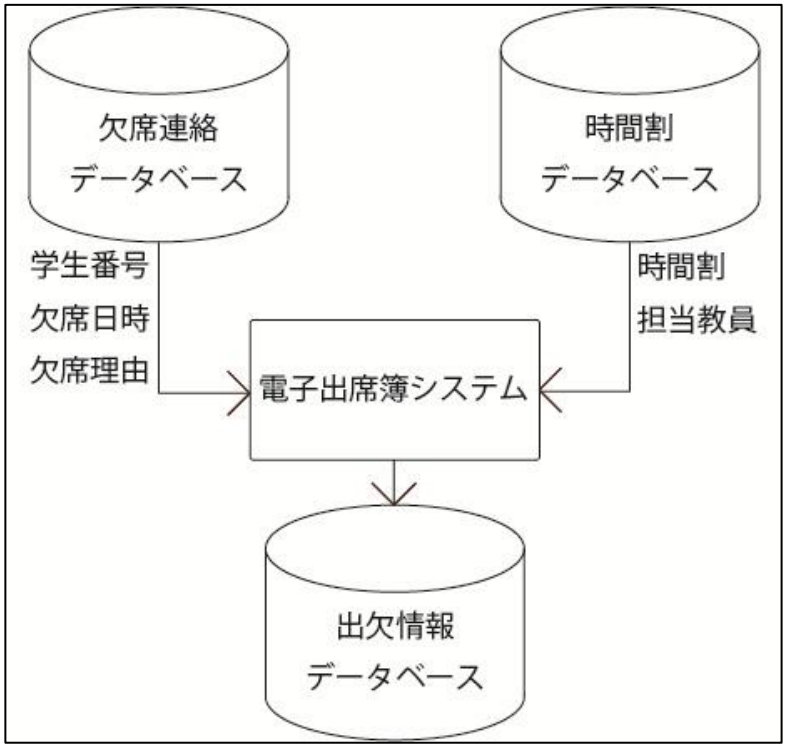


図 2.2.1　設計のイメージ図

全体の設計として既存の欠席連絡システム，時間割システムのほかに新規のシステムとして電子出席簿を作成する．電子出席簿では欠席連絡システムのデータベースから学生情報と欠席情報を，時間割システムのデータベースから時間割，教員，シラバスを参照する．そして，各データベースから取得した出欠席情報を，まとめて管理するようデータベースに保存する．

また使用端末として，スマートフォン・タブレットを想定している．

2.3 システム遷移図

図 2.3.1 は電子出席簿全体のシステム遷移図である.

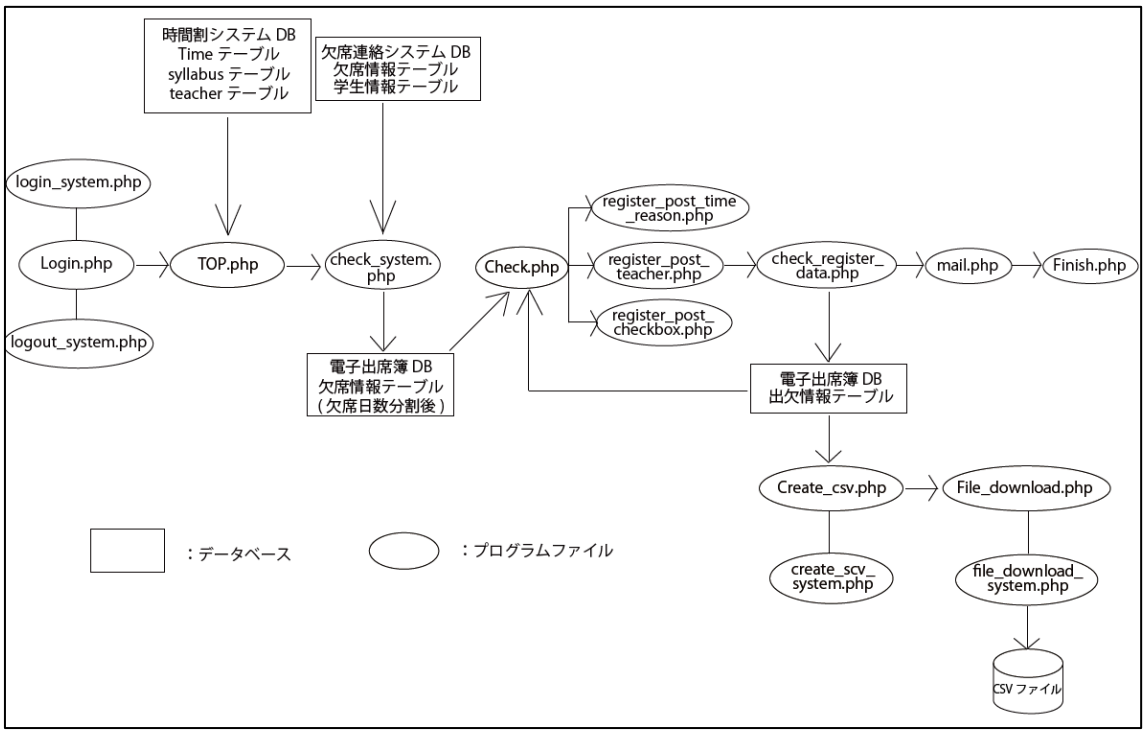


図 2.3.1　システム遷移図

図の上部は，基本的な出欠確認の流れである．また，図の右下部についてはエクスポート機能部分である．それぞれのシステムの詳細は第 3 章で後述する．ここで各プログラムの機能について軽く触れたものが表 2.3.1 である.

表 2.3.1　プログラム一覧

ファイル名	役割
connect.php	データベースへの接続
Login.php	ログインページの表示
login_system.php	ログイン機能
logout_system.php	ログアウト機能
TOP.php	TOP ページ表示
check_system.php	欠席データ分割機能
Check.php	出欠確認ページ表示
register_post_time_reason.php	欠席期間・理由取得
register_post_teacher.php	教員名取得
register_post_checkbox.php	出欠フラグ取得
check_data_register.php	出欠テーブルに登録・編集
mail.php	メール機能
Finish.php	出欠確認完了ページ表示
Create_csv.php	エクスポートページ表示
create_csv_system.php	エクスポート機能
File_download.php	ダウンロードページ表示
file_download_system.php	ダウンロード機能
rollbook.css	ページレイアウト用
rollbook.js	各データ取得用

2.4 データベース

図 2.3.1 は電子出席簿で使用するデータベースのイメージ図である.

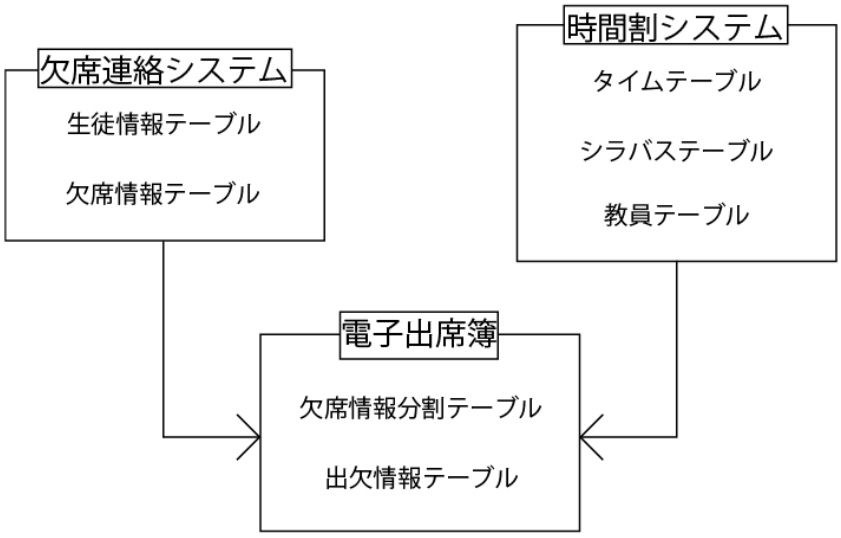


図 2.3.1　電子出席簿で使用するデータベース図

電子出席簿では，欠席連絡システム[1]からは生徒情報テーブル・欠席情報テーブルを使用し，時間割システム[2]からはタイムテーブル・シラバステーブル・教員テーブルを使用する．また，欠席情報テーブルの情報を分割保存するための欠席情報分割テーブルと，分割した欠席情報と出席情報を保存する欠席情報テーブルを新規のテーブルをして作成した.

(1)欠席連絡システム

図 2.3.4 と図 2.3.5 は，電子出席簿から参照する欠席連絡システムのテーブルである．

	#	名前	データ型	照合順序	属性	NULL	デフォルト値	コメント	その他	操作
☐	1	grade	int(11)			いいえ	なし	学生		変更 削除 その他
☐	2	student_number	varchar(6)	utf8mb4_general_ci		いいえ	なし	学籍番号		変更 削除 その他
☐	3	family_name	text	utf8mb4_general_ci		いいえ	なし	姓		変更 削除 その他
☐	4	first_name	text	utf8mb4_general_ci		いいえ	なし	名		変更 削除 その他
☐	5	mail_address	varchar(30)	utf8mb4_general_ci		いいえ	なし	メールアドレス		変更 削除 その他
☐	6	phone_number	text	utf8mb4_general_ci		いいえ	なし	電話番号		変更 削除 その他
☐	7	registration_datetime	date			いいえ	なし	登録日時		変更 削除 その他
☐	8	expiration_date	date			いいえ	なし	有効期限		変更 削除 その他
☐	9	username	varchar(10)	utf8mb4_general_ci		いいえ	なし	ユーザー名		変更 削除 その他
☐	10	password	varchar(64)	utf8mb4_general_ci		いいえ	なし	パスワード		変更 削除 その他

図 2.3.4　生徒情報テーブル

	#	名前	データ型	照合順序	属性	NULL	デフォルト値	コメント	その他	操作
☐	1	ID	int(11)			いいえ	なし	ID	AUTO_INCREMENT	変更 削除 その他
☐	2	student_number	varchar(6)	utf8_general_ci		いいえ	なし	学籍番号		変更 削除 その他
☐	3	username	varchar(10)	utf8_general_ci		いいえ	なし	ユーザー名		変更 削除 その他
☐	4	submission_date	date			いいえ	なし	提出日		変更 削除 その他
☐	5	absence_start_date	date			いいえ	なし	欠席開始日		変更 削除 その他
☐	6	absence_last_date	date			いいえ	なし	欠席最終日		変更 削除 その他
☐	7	absence_start_time	time			いいえ	なし	欠席開始時間		変更 削除 その他
☐	8	absence_end_time	time			いいえ	なし	欠席終了時間		変更 削除 その他
☐	9	absence_reason	text	utf8_general_ci		いいえ	なし	欠席理由		変更 削除 その他
☐	10	company_name	text	utf8_general_ci		はい	*NULL*	会社名		変更 削除 その他
☐	11	company_address	text	utf8_general_ci		はい	*NULL*	会社所在地		変更 削除 その他
☐	12	remarks	text	utf8_general_ci		はい	*NULL*	備考		変更 削除 その他
☐	13	check_flag	enum('true', 'false')	utf8_general_ci		いいえ	false	チェックフラグ		変更 削除 その他

図 2.3.5　欠席情報テーブル

(2)時間割システム

図 2.3.6 と図 2.3.7 と図 2.3.8 は，電子出席簿から参照する時間割システムのテーブルである．

	#	名前	データ型	照合順序	属性	NULL	デフォルト値	コメント	その他	操作
☐	1	**Date**	date			はい	*NULL*			変更 削除 その他
☐	2	**day**	varchar(1)	utf8_general_ci		はい	*NULL*			変更 削除 その他
☐	3	**Monday**	date			はい	*NULL*			変更 削除 その他
☐	4	**grade**	int(1)			はい	*NULL*			変更 削除 その他
☐	5	**period**	int(1)			はい	*NULL*			変更 削除 その他
☐	6	**sID**	varchar(10)	utf8_general_ci		はい	*NULL*			変更 削除 その他
☐	7	**sbefore**	varchar(10)	utf8_general_ci		はい	*NULL*			変更 削除 その他
☐	8	**flag**	int(1)			はい	*NULL*			変更 削除 その他
☐	9	**pID**	varchar(10)	utf8_general_ci		はい	*NULL*			変更 削除 その他
☐	10	**pbefore**	varchar(10)	utf8_general_ci		はい	*NULL*			変更 削除 その他
☐	11	**pflag**	int(1)			はい	*NULL*			変更 削除 その他

図 2.3.6　タイムテーブル

	#	名前	データ型	照合順序	属性	NULL	デフォルト値	コメント	その他	操作
☐	1	**id**	int(5)			はい	*NULL*			変更 削除 その他
☐	2	**subject**	varchar(14)	utf8_general_ci		はい	*NULL*			変更 削除 その他
☐	3	**period**	varchar(1)	utf8_general_ci		はい	*NULL*			変更 削除 その他
☐	4	**units**	int(2)			はい	*NULL*			変更 削除 その他
☐	5	**tID**	int(2)			はい	*NULL*			変更 削除 その他
☐	6	**flag**	int(1)			はい	*NULL*			変更 削除 その他

図 2.3.7　シラバステーブル

	#	名前	データ型	照合順序	属性	NULL	デフォルト値	コメント	その他	操作
☐	1	**id** 🔑	int(3)			いいえ	なし		AUTO_INCREMENT	変更 削除 その他
☐	2	**name**	varchar(17)	utf8_general_ci		はい	*NULL*			変更 削除 その他
☐	3	**color**	varchar(7)	utf8_general_ci		はい	*NULL*			変更 削除 その他

図 2.3.8　教員テーブル

(3)電子出席簿

図 2.3.2 と図 2.3.3 は，電子出席簿で新規作成したテーブルである．

	#	名前	データ型	照合順序	属性	NULL	デフォルト値	コメント	その他	操作
☐	1	ID 🔑	int(11)			いいえ	なし	ID	AUTO_INCREMENT	変更 削除 その他
☐	2	student_number	varchar(7)	utf8mb4_general_ci		いいえ	なし	学籍番号		変更 削除 その他
☐	3	absence_date	date			いいえ	なし	欠席日		変更 削除 その他
☐	4	absence_start_time	time			いいえ	なし	欠席開始時間		変更 削除 その他
☐	5	absence_end_time	time			いいえ	なし	欠席終了時間		変更 削除 その他
☐	6	reason	text	utf8mb4_general_ci		いいえ	なし	欠席理由		変更 削除 その他
☐	7	identity_absence_id	int(11)			いいえ	なし	個別欠席ID		変更 削除 その他

図 2.3.2　欠席情報分割テーブル

	#	名前	データ型	照合順序	属性	NULL	デフォルト値	コメント	その他	操作
☐	1	ID 🔑	int(11)			いいえ	なし	ID	AUTO_INCREMENT	変更 削除 その他
☐	2	student_number	varchar(7)	utf8_unicode_ci		いいえ	なし	学籍番号		変更 削除 その他
☐	3	class_id	int(11)			いいえ	なし	科目ID		変更 削除 その他
☐	4	date	date			いいえ	なし	日付		変更 削除 その他
☐	5	hours	int(11)			いいえ	なし	コマ目		変更 削除 その他
☐	6	absence_time_start	time			いいえ	なし	欠席開始時刻		変更 削除 その他
☐	7	absence_time_end	time			いいえ	なし	欠席終了時刻		変更 削除 その他
☐	8	absence_reason	text	utf8_unicode_ci		いいえ	なし	欠席理由		変更 削除 その他
☐	9	teacher	text	utf8_unicode_ci		いいえ	なし	教員		変更 削除 その他
☐	10	check_flag	int(1)			いいえ	なし	チェックフラグ		変更 削除 その他
☐	11	update_at	datetime		on update CURRENT_TIMESTAMP	いいえ	CURRENT_TIMESTAMP	更新日時	ON UPDATE CURRENT_TIMESTAMP	変更 削除 その他

図 2.3.2　出欠情報テーブル

第 3 章　電子出席簿の仕様

3.1 セキュリティ

図 3.1.1，図 3.1.2 は，ログインページである．

教員以外のユーザーが電子出席簿を操作できないようにするために，セキュリティとしてログイン機能[3]を電子出席簿に実装した．ログイン画面に入力するユーザーID とパスワードは教員共通のものを用意した．

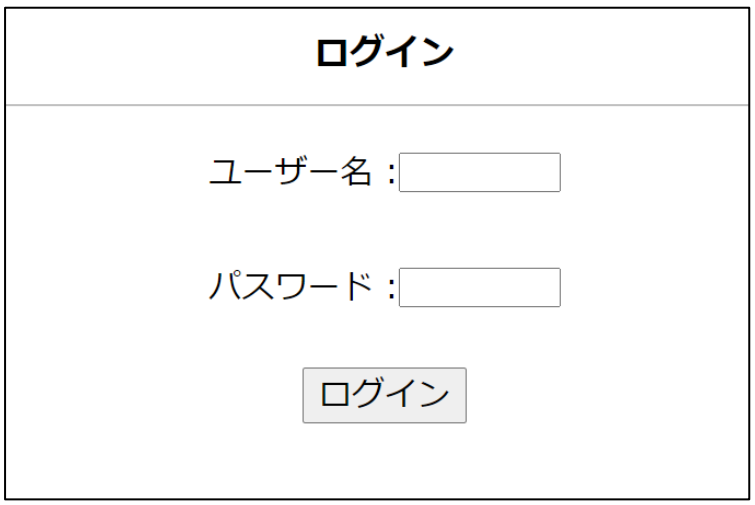


図 3.1.1　ログインページ(左)　図 3.1.2　レスポンシブ対応(右)

次にプログラムを説明する.

```
                                    (省略)
//======= ログイン状況の確認 ==============
session_start();

if(isset($_SESSION['data'])){
  $login = true;
  $data = $_SESSION['data'];
  $username = $data['username'];
  $role = $data['role'];
} else {
        $login = false;
}

// すでにログインしている場合，メインページに遷移
if($login){
  header('Location: TOP.php');
  exit;
}
 //======= ログイン状況の確認 END ==============
                                    (省略)
```

①

②

リスト 3.1-1　Login.php

①では，既にログイン画面から入力されたログイン情報がある際に，ログインをしている状態へ変化し，ログインデータからユーザーID を取り出す.

②では，既にログインしている状態なら TOP ページに遷移する．ログインしていない状態ならば login_system.php に遷移する.

```
if(!file_exists("./data/".$username.".txt")){ // ユーザーデータがない場合 ] ①
  header('Location: Login.php?
        message=ユーザーネームまたはパスワードが間違っています');
  exit;
  }else{ // ユーザーデータがある場合
     //ユーザーデータの取得
     $user_data = fopen('data/'.$username.'.txt', "r");
     // trim 関数：文字列の最初と最後にあるホワイトスペースを取り除く
     $password_hash = trim(fgets($user_data));
     $role = trim(fgets($user_data));
     fclose($user_data);
     // パスワードの照合
     if(md5($password) != $password_hash){ // パスワードが一致しない場合
        header('Location: Login.php?message=
         ユーザーネームまたはパスワードが間違っています. ');
        exit;                                                          ②
     }
     // ログイン OK
     session_start();
     $data = [];
     $data['username'] = $username;
     $data['role'] = $role;                                            ③
     $_SESSION['data'] = $data; // ユーザ情報をセッション変数に格納
     header('Location: TOP.php'); // メインページに遷移
     exit;
  }
```

リスト 3.1-2　login_system.php

login_system.php では，ログイン時に入力したユーザーID と同じ名前のテキストファイルが data フォルダに存在しない場合はログインを拒否する(①)．ユーザーID と同じ名前のテキストファイルが存在する場合は，ファイルの中にあるパスワードを入力したパスワードを比較して一致していればメインページに遷移(②)し，一致していなければログインを拒否(③)する．

また，ログイン時のユーザーID とパスワードを変更したいときは，data フォルダ内のテキストファイルの名前を変更するとユーザーID が，ファイルの中にある文字を変更するとパスワードを変更できる．パスワードに関しては，パスワードの文字列をそのまま入力するのはセキュリティ上よくないので，パスワードを MD5 方式のハッシュ値に変換して入力する．

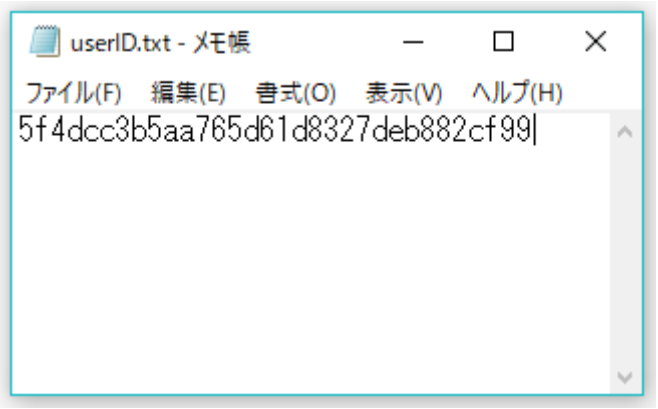


図 3.1.2　ログイン時に使用するファイルイメージ図と例

3.2 TOP

下の図 3.2.1，図 3.2.2 は電子出席簿の TOP ページである．

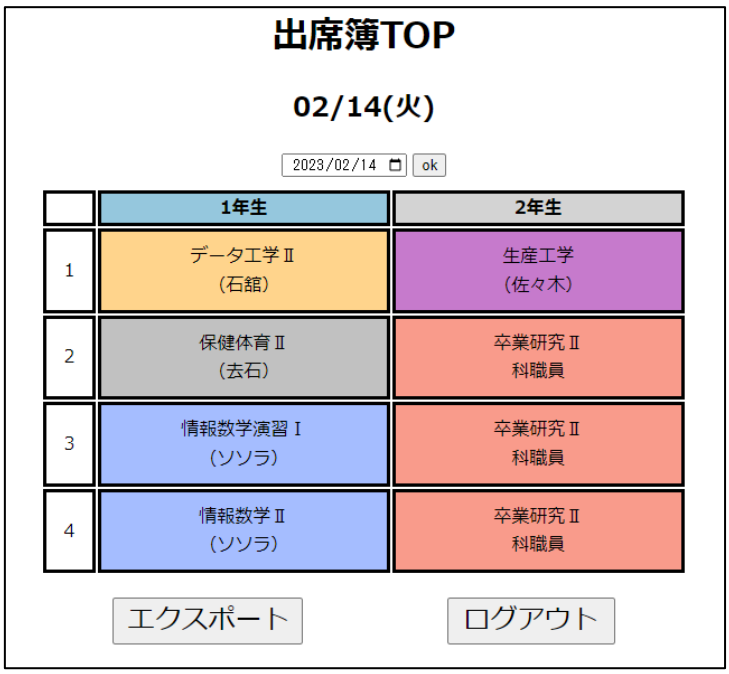


図 3.2.1　TOP ページ(左)　図 3.2.2　レスポンシブ対応(右)

このページではアクセスした日付の時間割を，時間割システムのデータベースから参照して表示する．画像上部のテキストボックスをクリックすることで，カレンダーから日付を選択でき，隣の ok ボタンを押下することで選択した日付の時間割を表示できる．時間割の各科目はそれぞれボタンになっており，そのボタンを押下すると各科目の出欠確認ページに遷移する．エクスポートボタンではファイルを CSV 出力することができる画面に遷移する．

次にプログラムを説明する.

```
                                    (省略)
      //5 コマ目の有無（false:true）
  $flag_5 = empty($pdo->query("SELECT Date FROM `timetable` WHERE Date='".
        $_SESSION['date'] ."' AND period = 5")->fetchAll(PDO::FETCH_ASSOC)); ①
  $count1=$pdo->query("SELECT Date FROM `timetable` WHERE Date >='" .
          $_SESSION['monday'] . "' AND Date <='" . $_SESSION['yestday'] . "' AND
grade = 1          ")->fetchAll(PDO::FETCH_ASSOC);
  $count2 = $pdo->query("SELECT Date FROM `timetable` WHERE Date >='" .
           $_SESSION['monday'] . "' AND Date <='" . $_SESSION['yestday'] . "'
AND grade = 2      ")->fetchAll(PDO::FETCH_ASSOC);
  $flagcount1 = count($count1);
  $flagcount2 = count($count2);
  $border = ($flag_5) ? 4 : 5; ②

  $date = date('Y/n/j', strtotime(isset($i) . ' day', strtotime($Mon)));
  for ($j = 1; $j <= $border; $j++) {
        $result = $pdo->query("SELECT sID FROM `timetable` WHERE Date ='" .
                  $_SESSION['date'] . "' AND grade =" . $_SESSION['grade'] .
                  " AND period =" . $j)->fetchAll(PDO::FETCH_ASSOC);
        $result2 = $pdo->query("SELECT sID FROM `timetable` WHERE Date ='" .
                   $_SESSION['date'] . "' AND grade = 2  AND period =" .
                   $j ."")->fetchAll(PDO::FETCH_ASSOC);
        if (!empty($result)) {
                if (is_numeric($result[0]['sID'])) {
  ③                     $info = $pdo->query("SELECT s.subject,t.name,t.color
                               FROM `syllabus` AS s,`teacher` AS t
                              WHERE s.id=" . $result[0]['sID'] . "
                              AND s.tID=t.id")->fetchAll(PDO::FETCH_ASSOC);
                      array_push($sub, $info[0]['subject']);
                      array_push($teacher, $info[0]['name']);
                      array_push($color, $info[0]['color']);;
                } else {
```

```
                        array_push($sub, $result[0]['sID']);
                        array_push($teacher, "");
                }
        }else{
                array_push($sub, "");
                array_push($teacher, "");
        }

        if (!empty($result2)) {
                if (is_numeric($result2[0]['sID'])) {
                        $info2 = $pdo->query("SELECT s.subject,t.name,t.color
                                FROM  `syllabus` AS s,`teacher` AS t
                                WHERE s.id=" . $result2[0]['sID'] . "
③                               AND s.tID=t.id")->fetchAll(PDO::FETCH_ASSOC);
                        array_push($sub2, $info2[0]['subject']);
                        array_push($teacher2, $info2[0]['name']);
                        array_push($color2, $info2[0]['color']);
                } else {
                        array_push($sub2, $result2[0]['sID']);
                        array_push($teacher2, "");
                }
        }else{
                array_push($sub2, "");
                array_push($teacher2, "");
        }
}
                                        (省)
$day = $_SESSION['date'];
$nowweek = $_SESSION['week'] - 1;
for($j = 0; $j < $border; $j++){
        echo "<tr class=¥"TOP-today¥" id=¥"today¥">";
        //何コマ目
        echo "<td class=¥"class0¥" name=¥"hours_class¥">", $j+1 ,"</td>";
```
④

```
        //1年生
        echo "<td class=¥"timetable-class1¥" style=¥"background-color:",
              $colors[$teacher[$j]], "¥">";
        echo "<form method=¥"post¥" action=¥"check_system.php¥">";
        echo "<input type=¥"hidden¥" name=¥"grade¥" value=¥"1¥">";
        echo "<input type=¥"hidden¥" name=¥"subject¥" value=¥"$sub[$j]¥">";
        echo "<input type=¥"hidden¥" name=¥"hours_class¥" value=¥"$j¥">";
        echo "<input type=¥"hidden¥" name=¥"teacher¥" value=¥"$teacher[$j]¥">";
        echo "<button type=¥"submit¥" class=¥"tmtable-submit-class1¥"
              style=¥"background-color:", $colors[$teacher[$j]], "¥">
              <p class=¥"class-name¥" for=¥"two¥">", $sub[$j], "</p>";
        echo "<p class=¥"teacher-name¥" for=¥"two¥">", $teacher[$j], "</p>";
        echo "</button></form></td>";
        //2年生
④       echo "<td class=¥"timetable-class2¥" style=¥"background-color:",
              $colors[$teacher2[$j]], "¥">";
        echo "<form method=¥"post¥" action=¥"check_system.php¥">";
        echo "<input type=¥"hidden¥" name=¥"grade¥" value=¥"2¥">";
        echo "<input type=¥"hidden¥" name=¥"subject¥" value=¥"$sub2[$j]¥">";
        echo "<input type=¥"hidden¥" name=¥"hours_class¥" value=¥"$j¥">";
        echo "<input type=¥"hidden¥" name=¥"teacher¥"
              value=¥"$teacher2[$j]¥">";
        echo "<button type=¥"submit¥" class=¥"tmtable-submit-class2¥"
              style=¥"background-color:", $colors[$teacher2[$j]], "¥">
              <p class=¥"class-name¥" for=¥"two¥">", $sub2[$j], "</p>";
        echo "<p class=¥"teacher-name¥" for=¥"two¥">", $teacher2[$j], "</p>";
        echo "</button></form></td></tr>";
}

//check_system.phpに必要データを配列へ格納後
$date_data = [];
$date_data['date'] = $day;
$_SESSION['date_data'] = $date_data; // 日付ををセッション変数に格納
```

リスト 3.2-1　TOP.php

まず，$flag_5 にその日の時間割が４時限目までなら true， ５時限目までなら false を挿入し，$border で４もしくは５を挿入する(①)．その後の for 文の条件を$border までとすることで時間割が何時限目まであるのかを判別する(②)．

for 文の中は，科目，担当教員，科目ごとの色の３つの配列に，シラバステーブル・教員テーブルから配列$info に挿入した科目，教員名，科目ごとの色を１行ずつ挿入する．(③)

そして for 文で挿入した３つの配列を使用して，時間割を表示する．(④)

3.3 出席確認

出欠確認で作成したファイルを表 3.3-1 に示す.

表 3.3-1　出欠確認機能ファイル一覧

ファイル名	役割
Check.php	出欠確認ページ
rollbook.js	各データ取得用

(1) 出欠確認ページ上部について

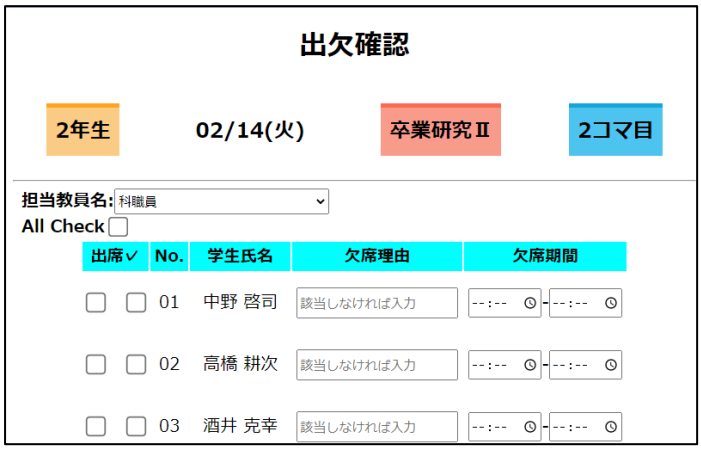


図 3.3-1　出席確認ページ上部(左)　図 3.3-2　レスポンシブ対応(右)

TOP ページから選択した科目に遷移した画面となっている．図のヘッダー部分では，学年・日付・科目名・コマ目を表示する．ボディ部分の担当教員名は，科目ごとの担当教員を入力するセレクトボックストなっている．しかし，システム側が自動で担当教員を入力するようにした．

All Check では，出席している学生全員を出席としてチェックすることができる．出欠確認のチェックの簡略化をする役割を持つ．

テーブル部分では，学生の出欠情報を表示する．その際に欠席連絡システムで欠席連絡をしていた学生は自動的に欠席理由・期間が入力され，出席✓でも欠席した時間に応じてチェックがされるようにした．出席✓で 2 つチェックボックスがある理由としては，本校の出席点の算出方法が関係している．本校は 1 コマの時間が 90 分でありこの時間出席することで出席点を満点取得することができる．しかし，1 コマの時間の半分である 45 分出席することでも出席点を半分取得できる．このような算出方法となっているので出席✓を 2 つ用意し，0 つチェックで欠席，1 つチェックで 45 分出席，2 つチェックで 90 分出席として分けるようにした．

次にプログラムを説明する.

```
（省略）
session_start();
isset($_SESSION['check_data']);
$check_data = $_SESSION['check_data'];
$day_week = $check_data['day_week']; //日付と曜日
$grade = $check_data['grade']; //学年
$subject = $check_data['subject']; //科目名
$hours_class = $check_data['hours_class'];
$teacher = $check_data['teacher']; //教員名
isset($_SESSION['date_data']);
$date_data = $_SESSION['date_data']; //日付
$date = $date_data['date'];
（省略）
```

リスト 3.3-1　Check.php

リスト 3.3-1 は TOP ページから SESSION されたデータを取得するプログラムである. 取得するデータは，日付と曜日・学年・科目名・コマ目・教員名・日付である.

```
(省略)
//取得データ（時間）が何コマ目に対応しているか判断する
for ($i=0; $i<count($absence_student); $i++) {
    $get_time_data = ""; //欠席学生の欠席期間と欠席理由
    $get_time_data = Hours_Absence_Time($start_second[$i], $end_second[$i],
                    $start_time[$i][0], $end_time[$i][0], $hours_class,
                    $absence_student[$i][3]);
    $start_time[$i][0] = $get_time_data[0];
    $end_time[$i][0] = $get_time_data[1];
    $absence_student[$i][3] = $get_time_data[2];
}
```

①

```
//コマ数で時間を指定
function Hours_Absence_Time($start_second, $end_second, $start_time, $end_time,
$hours_class, $absence_reason){
    //「時:分」を「秒(数値)」に対応
    //対応データ配置 08:50, 10:20,  10:30, 12:00,  13:00, 14:30,  14:40, 16:10,
                    16:20, 17:50
    $class_seconds  =  [[31800,  37200], [37800,  43200], [46800,  52200], [52800,
58200],
                  [58800, 64200]];

    //次のコマをはさむの時間の場合
    if($start_second <= $class_seconds[$hours_class-1][0]){
        switch($hours_class){
            case 1 :$start_time = "08:50"; break;
            case 2 :$start_time = "10:30"; break;
            case 3 :$start_time = "13:00"; break;
            case 4 :$start_time = "14:40"; break;
            case 5 :$start_time = "16:20"; break;
        }
    }
    if($end_second >= $class_seconds[$hours_class-1][1]){
        switch($hours_class){
            case 1 :$end_time = "10:20"; break;
            case 2 :$end_time = "12:00"; break;
            case 3 :$end_time = "14:30"; break;
            case 4 :$end_time = "16:10"; break;
            case 5 :$end_time = "17:50"; break;
        }
    }
```

②

③

```
    //n コマ目にかぶらない場合                                          ④
    if($end_second <= $class_seconds[$hours_class-1][0] ||
      $start_second >= $class_seconds[$hours_class][0]){
        $start_time = "";
        $end_time = "";
        $absence_reason = "";
    }
    //調整した欠席データを戻す                                          ⑤
    return array($start_time, $end_time, $absence_reason);
}
```

（省略）

リスト 3.3-3　Check.php

リスト 3.3-3 は取得した欠席データの欠席期間が何コマ目のところに対応しているか判断するプログラムである.

①は欠席した学生一人一人の欠席時間帯と欠席理由を判断し，欠席開始時間・欠席終了時間・欠席理由をデータ挿入用の配列に格納する処理である.

関数 Hours_Absence_Time()は，この出欠確認が何コマ目の出欠確認か判断し，そのコマに対応した欠席開始時間・欠席終了時間を格納する関数である．変数$start_second[$i],$end_second[$i]は欠席開始・終了時刻を秒数に変換したものである．変数 $start_time[$i][0],$end_time[$i][0]は欠席開始・終了時刻である．変数$hours_class はコマ目である．変数$absence_student[$i][3]は欠席学生の学籍番号である.

②・③は次のコマを挟んだ時（例：08:50～11:00 など）に欠席開始時間・欠席終了時間を 1 コマの区切りごとに変更する処理である．②は欠席開始時間，③は欠席終了時間を変更する．通常 4 コマで授業時間が終了するが，イレギュラーの対応をするときに 5 コマ目を作る必要があるときが出てくる．これに対応できるように 5 コマ目でも問題なくプログラムが動くように工夫した.

④は他のコマを挟まない場合（例：09:00～10:00 など）に欠席時間を変更させないようにする処理である.

⑤はコマごとの時間帯で変更した欠席開始時間・欠席終了時間・欠席理由を返す処理である.

```
（省略）
//科目名をシラバスデータの科目名と一致させる
for($i=0; $i<count($syllabus); $i++){
    if($subject == $syllabus[$i][1]){
        $subject_id = $syllabus[$i][0];  }  } //$subject_id 科目 ID
（省略）
```

リスト 3.3-4　Check.php

リスト 3.3-4 は SESSION から取得した科目名を，時間割システムのデータベースの syllabus テーブルより同名の科目名の科目 ID を取得するプログラムである．科目 ID は登録された出欠情報テーブルを参照するとき科目の分別をするときに使用する．

```
（省略）
//欠席人数
$count = count($absence_student); //欠席連絡した欠席学生の人数
（省略）
//欠席人数を追加
$addcount = 0; //出欠情報テーブルに登録されている欠席学生の人数
for($i=0; $i<count($student); $i++){
    if($student_data[$i][7] != 2){
        $addcount++;  }
    if($i == count($student)-1 && ($addcount - $count) != -$count){
        $count = $count + ($addcount - $count);  }  }
（省略）
```

リスト 3.3-5　Check.php

リスト 3.3-5 は欠席人数をカウントするプログラムである．プログラム内の$count 後述するリスト 3.3-9 で使用する．$count は欠席人数の変数である．初期値として出欠情報テーブルに登録されている欠席学生をカウントした値が入っている．この時テーブルにデータが登録されていなければ値は 0 となる．さらに，初期値に追加する値として欠席連絡システムで欠席連絡のあった学生をカウントした値を挿入する．

```
（省略）
<?php
    echo "<label for=¥"teacher_select¥">担当教員名:</label>";
    echo "<select name=¥"teachers¥" id=¥"teacher_select¥">";
    $teacher_flag = 0; //取得したデータと同一の教員がいるかどうか
    for($i=1; $i<count($teachers); $i++){
        if($teacher == $teachers[$i][1]){
           $teacher_flag = 1;
           echo "<option value=¥"",$teachers[$i][1],"¥">",
                 $teachers[$i][1],"</option>"; //教員名が入力される
           break;
        }
    }
    if($teacher_flag = 0){
       echo "<option value=¥"¥" hidden>選択</option>";
    }
    for($i=1; $i<count($teachers); $i++){
        echo "<option value=¥"",$teachers[$i][1],"¥">",
              $teachers[$i][1],"</option>";
    }
    echo "</select>";
?>
（省略）
```

①

②

リスト 3.3-7　Check.php

リスト 3.3-7 は担当教員名の入力セレクトボックスのプログラムである.

①は教員名の初期値を決定する処理である. TOP ページから SESSION で取得した教員名と時間割システムの teacher テーブルの教員名で名前が一致する教員がいた場合は初期値にその教員名が挿入され，一致しなかったときは教員名の選択を促すような初期値が挿入される.

②は選択できる教員名をセレクトボックスに挿入する処理である. teacher テーブルから教員名の情報を参照して挿入している.

```
（省略）
<label for="Allcheck" class="allcheck-button">All Check</label>
<input onclick="FullCheck(<?php echo $hours_class; ?>)"
 type="checkbox" id="Allcheck">
（省略）
```

リスト 3.3-8　Check.php

リスト 3.3-8 は AllCheck 機能についてのプログラムである．この機能については JavaScript で機能を作成したためリスト 3.3-12 で詳細を後述する．

```
（省略）
//学生データ（姓名）結合
    for($i=0; $i<count($student); $i++){
       $student_name[$i] = ""; //学生名（姓名）
       for($j=1; $j<3; $j++){
          $student_name[$i] .= $student[$i][$j]; //学生名（姓→名の順に追加）
          if($j == 1){
              $student_name[$i] .= " ";
          }
       }
    }
（省略）
```

リスト 3.3-9　Check.php

リスト 3.3-9 は学生の姓と名を結合するプログラムである．学生の名前は欠席連絡システムの internal_student_user テーブルから参照している．このテーブルの仕様上，姓と名が分別されて登録されていたためプログラム内で名前を結合した．

```
                                    (省略)
//学生データ表示
for($i=0; $i<count($student); $i++){

    //配列エラー回避
    $start_value[$i] = ""; //欠席開始時刻
    $end_value[$i] = ""; //欠席終了時刻
    $reason_value[$i] = ""; //欠席理由

    //学籍番号で比較し，欠席データを挿入する
    for ($j=0; $j<$count; $j++) {
        //学生一人ごとにすべての欠席データと比較する
        if(strcmp($student[$i][0], $absence_student[$j][0]) == 0 ||
           strcmp($student[$i][0],
          $student_data[$j][6]) == 0){ //欠席した学生を検索
            if($subject_id == $student_data[$i][0] &&
                $date == $student_data[$i][1] &&
                $hours_class == $student_data[$i][2]){
                //無断欠席の学生が欠席連絡システムで欠席届を提出したとき
                if($student_data[$i][5] == "" && $student_data[$i][7] != 2 &&
                  strcmp($student[$i][0], $absence_student[$j][0]) == 0){
                    //欠席期間
                    $start_value[$i] = $start_time[$j][0];
                    $end_value[$i] = $end_time[$j][0];
                    //欠席理由
                    $reason_value[$i] = $absence_student[$j][3];
                }else if($student_data[$i][5] == ""){
                    $start_value[$i] = "--:--:--";
                    $end_value[$i] = "--:--:--";
                }else{
                    //欠席期間
                    $start_value[$i] = $student_data[$i][3];
                    $end_value[$i] = $student_data[$i][4];
                    //欠席理由
                    $reason_value[$i] = $student_data[$i][5];
                }
```

①

②

③

④

```
            }
            if($subject_id != $student_data[$i][0] &&
               $date != $student_data[$i][1] &&
               $hours_class != $student_data[$i][2]){
                //欠席期間
                $start_value[$i] = $start_time[$j][0];
                $end_value[$i] = $end_time[$j][0];
                //欠席理由
                $reason_value[$i] = $absence_student[$j][3];
            }
        }
    }
```

⑤

（省略）

リスト 3.3-10　Check.php

リスト 3.3-10 では実際のページ上に学生のデータを表示するプログラムである．学生のデータの中でも欠席データを挿入する処理である．

①は学生が 1 コマごとに欠席データを持っているか判別する処理である．欠席連絡システムのデータベースと出欠情報テーブルのどちらも参照して判別する．

②は無断欠席と判断された学生が欠席連絡システムから欠席届を提出したときに提出された欠席データを挿入する処理である．

③は出席学生の欠席時間を null 値にする処理である．不当な欠席データが挿入される危険を回避するためにも使用される．

④は出欠情報テーブルから取得した欠席データを挿入する処理である．授業中や授業後に出欠情報を編集したいときに使用される．

⑤は欠席連絡システムのデータベースから取得した欠席データを挿入する処理である．授業時の最初の出欠確認で使用される．

```
（省略）
echo "<tr class=¥"checktable-body¥">";
echo "<td class=¥"t-td1¥" hidden><input class=¥"checks¥"
     onclick=¥"Check_Start_End($i);¥"
     id=¥"checkbox_$i¥" type=¥"checkbox¥"></td>";                    ①
//編集時に出席者をチェックする
switch($student_data[$i][7]){ //チェックフラグの値
    case 1 : $check_flag1 ='checked'; $check_flag2 ='checked'; break;
    case 2 : $check_flag1 ='checked'; $check_flag2 ='checked'; break;
    default: $check_flag1 =''; $check_flag2 =''; break;             ②
}
（省略）
```

リスト 3.3-11　Check.php

リスト 3.3-11 は 2 つのチェックボックスを同時にチェックする機能と，チェックボックス関するプログラムである．

①の処理は JavaScript で行っているため，後述するリスト 3.3-12 に詳細を記述する．

②は出欠情報の編集時に自動で出席学生全員にチェックがされている状態にする処理である．出欠情報テーブルのチェックフラグを参照して，チェックボックスを何個チェック済みにするか決定する．

```
//Allcheck の全選択
function FullCheck(hour) {
    //AllCheck のチェックボックス
    let allcheck = document.querySelector("#Allcheck");
    //AllCheck 以外のチェックボックス
    let checks = document.querySelectorAll(".checks");
    //欠席理由があるかどうか判断するための変数 reason
    let reason_array = [];
    let start_array = []; //開始時刻の秒数
    let end_array = []; //終了時刻の秒数
    //半分出席しているか判断するための変数
    let harf_time = 2700;
    for(let i=0; i<checks.length/3; i++){ //1 名につきチェックボックスが 3 つ
        let reason = document.getElementById("reason"+i).value;
        let start = document.getElementById("start"+i).value;
        let end = document.getElementById("end"+i).value;
        //時間を秒数に変換
        let start_split = start.split(':');
        let end_split = end.split(':');
        let start_second = start_split[0] * 3600 + start_split[1] * 60;
        let end_second = end_split[0] * 3600 + end_split[1] * 60;
        reason_array.push(reason);
        start_array.push(start_second);
        end_array.push(end_second);
    }
```

①

②

```
（省略）
//全選択のチェックボックスイベント
allcheck.addEventListener('change', function() {
    //チェックされているか
    let count = 0; //学生人数分
    if (allcheck.checked) {
        //全て選択
        for (let i=0; i<checks.length; i++) {
            //欠席理由があり，半分は出席した場合
            （45 分より出席した場合半分出席）
            if(i % 3 == 1 &&
               end_array[count] - start_array[count] < harf_time){
                if (checks.hasOwnProperty(i)){
                    checks[i].checked = true; } }

            //欠席理由がなく，全ての時間出席した場合
            if(reason_array[count] == "" &&
                (i % 3 == 1 || i % 3 == 2)){
                if (checks.hasOwnProperty(i)){
                    checks[i].checked = true; } }

            if(i % 3 == 2){
                count++; }
            if(i % 3 == 0){
                if (checks.hasOwnProperty(i)){
                    checks[i].checked = true; } }

    } else {
        //全て解除
        for (let i in checks) {
            if (checks.hasOwnProperty(i)) {
                checks[i].checked = false; } }

}); }
```

③

④

⑤

⑥

リスト 3.3-12　rollbook.js

リスト 3.3-12 は JS で AllCheck の機能をプログラムしたものである．

①はどのチェックボックスで全選択を行うか設定する処理である．全選択チェックボックスは id，その他のチェックボックスは class で設定する．

②は出席のチェックをする学生を選別するために行う準備の処理である．欠席理由と欠席期間で出席か欠席か判断する．

③は 45 分以上出席した学生または，90 分間出席した学生に 1 つチェックをする処理である．

④は 90 分間出席した学生に③のチェックに追加して 1 つチェックをする処理である．この処理を通ることで 2 つチェックされた状態になる．

⑤は hidden で隠れているチェックボックスをチェックする処理である．hidden で隠れている理由はリスト 3.3-13 で後述する．

⑥はチェックをすべて解除する処理である．

この機能については参考資料[4]を参考に作成した．

```
                                    (省略)
//前半と後半のチェックボックスを選択する(No.又は氏名押下時)
function Check_Start_End(j){
    let doublecheck = document.querySelector("#checkbox_"+j); //2 つ同時チェック
    let checks_s = document.querySelectorAll(".checks_start"+j); //1 つ目のチェック
    let checks_e = document.querySelectorAll(".checks_end"+j); //2 つ目のチェック

    doublecheck.addEventListener('change', function() {
        //チェックされているか
        if(doublecheck.checked) {
            //全て選択
            for (let i in checks_s) {
                if(checks_s.hasOwnProperty(i)) {
                    checks_s[i].checked = true;
                    checks_e[i].checked = true;
                }
            }
        }else{
            //全て解除
            for (let i in checks_s) {
                if(checks_s.hasOwnProperty(i)) {
                    checks_s[i].checked = false;
                    checks_e[i].checked = false;
                                    (省略)
    });
}
                                    (省略)
```

リスト 3.3-13　rollbook.js

リスト 3.3-13 は学生一人一人にある 2 つのチェックボックスをチェック状態にするプログラムである．基本的にはリスト 3.3-12 と同様の処理となっている．

実際のソースコードではチェックボックスが 3 つあるが, 1 つを hidden にしている. 理由としては, リスト 3.3-12 と同様の動きにする際に全選択するためのチェックボックスが必要なため 1 つ追加した．また，このチェックボックスはユーザー側に見える必要はないため隠している．

(2) 画面下部について

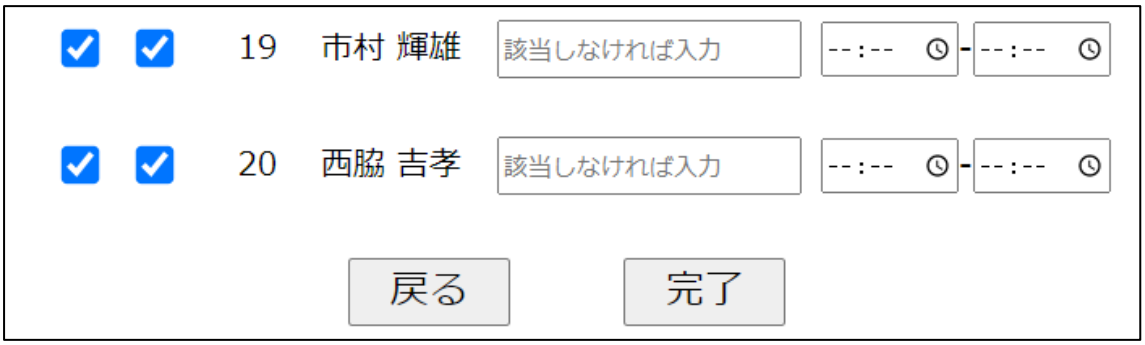


図 3.3-3　出席確認ページ下部

戻るボタンを押下することで，TOP ページに戻ることができる．

完了ボタンを押下することで，出欠情報テーブルにデータを登録，無断欠席している学生に確認メールを送信する．

次にプログラムを説明する.

```
（省略）
<button type="button" class="backpage-button" name="edit"
 onclick="location.href='./TOP.php'">戻る</button>
<?php
//check_data_register.php に必要データを格納
for($i=0; $i<count($student); $i++){
    $_SESSION['register_number'][$i] = $student[$i][0]; //学生番号
}
$check_register_data = [];
$check_register_data['subject_id'] = $subject_id; //科目 ID

$_SESSION['check_register_data'] = $check_register_data;
$_SESSION['teacher_count'] = count($teachers); //教員の人数
?>
<button onclick="Set_Value(<?php echo count($student); ?>);" type="submit"
class="nextpage-button">完了</button>
（省略）
```

リスト 3.3-14　Check.php

リスト 3.3-14 は SESSION でのデータの格納[5]と完了ボタンのプログラムである.

SESSION では学籍番号・科目 ID・教員の人数を格納している. これは出欠情報テーブルに情報を登録する際に使用する.

完了ボタンでは JavaScript で出欠情報テーブルに登録するプログラムにデータの受け渡しを行う. このプログラムの詳細は第 3 章の「3.6 出欠情報テーブル」で後述する.

(3) 出席簿の使用例について

図 3.3-3　出席簿の使用例

1 番はチェックが 2 つで，欠席理由が記載されていないため 1 コマ 90 分すべて出席している状態.

2 番はチェックがなく，欠席理由もないため無断欠席している状態.

3 番はチェックが 1 つで，欠席理由があり，欠席期間が 45 分より少ないため欠席連絡をしていて 45 分出席している状態.

4 番はチェックがなく，欠席理由があり，欠席期間が 45 分以上のため欠席連絡をしていて 1 コマ 90 分すべて欠席している状態.

3.4 出欠確認完了

出欠確認完了で作成したファイルを表 3.4-1 に示す.

表 3.4-1　出欠確認完了機能ファイル一覧

ファイル名	役割
Finish.php	出欠確認完了ページ

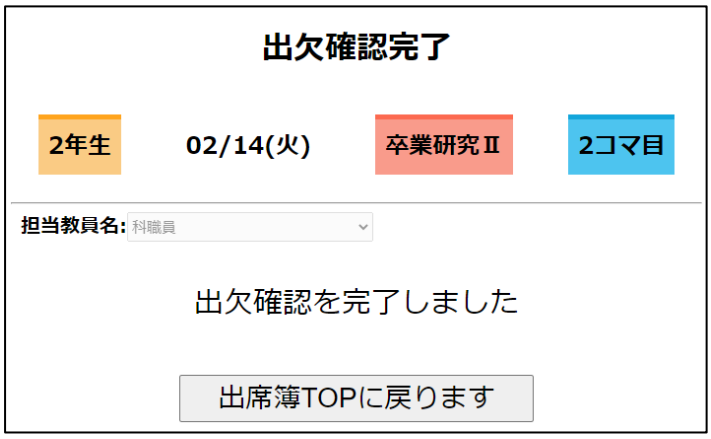


図 3.4-1　出欠確認完了ページ(左)　図 3.4-2　レスポンシブ対応(右)

出欠確認ページから遷移した画面となっている.

ボディ部の担当教員名は，このページは出欠確認が終了したことを表すページであるので変更できないようにした．画面下部のボタンを押下することで TOP ページに遷移する.

次にプログラムの説明をする.

```
（省略）
session_start();
isset($_SESSION['check_data']);
$check_data = $_SESSION['check_data'];
$day_week = $check_data['day_week']; //日付と曜日
$grade = $check_data['grade']; //学年
$subject = $check_data['subject']; //科目名
$hours_class = $check_data['hours_class']; //コマ目

isset($_SESSION['date_data']);
$date = $_SESSION['date_data']; //日付

//学籍番号が配列にある
isset($_SESSION['register_number']);
$student_number = $_SESSION['register_number']; //学籍番号

//科目 ID
isset($_SESSION['check_register_data']);
$subject_id = $_SESSION['check_register_data']; //科目 ID

//教員名，欠席理由，欠席開始・終了時刻
isset($_SESSION['register_teacher']);
$teacher = $_SESSION['register_teacher']; //教員名
isset($_SESSION['check_flag']);
$check_flag = $_SESSION['check_flag']; //チェックフラグ
isset($_SESSION['register_start']);
$absence_time_start = $_SESSION['register_start']; //欠席開始時刻
isset($_SESSION['register_end']);
$absence_time_end = $_SESSION['register_end']; //欠席終了時刻
isset($_SESSION['register_reason']);
$absence_reason = $_SESSION['register_reason']; //欠席理由
```

```
//教員の数,名前
$teacher_count = $_SESSION['teacher_count']; //教員の人数
$teacher_name = $_SESSION['teacher_name'];
                                        (省略)
```

リスト 3.4-1　Finish.php

リスト 3.4-1 は TOP・出席確認ページから SESSION で渡されたデータを受けとっているプログラムである．受けとっているデータは，日付と曜日・学年・科目名・コマ目・日付・学籍番号・科目 ID・教員名・チェックフラグ・欠席理由・欠席期間・教員の人数である．

```
unset($_SESSION['check_data']);
unset($_SESSION['date_data']);
unset($_SESSION['register_number']);
unset($_SESSION['check_register_data']);
unset($_SESSION['register_teacher']);
unset($_SESSION['check_flag']);
unset($_SESSION['register_start']);
unset($_SESSION['register_end']);
unset($_SESSION['register_reason']);
unset($_SESSION['teacher_count']);
unset($_SESSION['teacher_name']);
```

リスト 3.4-2　Finish.php

リスト 3.4-2 は使用した SESSION 情報を削除するプログラムである．個人情報漏洩の危険を減らすために，ログインの SESSION 情報のみ残してその他すべての SESSION を削除する．

3.5 欠席情報の分割・整理・統合

欠席情報の分割・整理・統合で作成したファイルを表 3.5-1 に示す.

表 3.5-1　欠席情報の分割・整理・統合機能ファイル一覧

ファイル名	役割
check_system.php	欠席情報の分割・整理・統合

欠席情報の分割について

当初の計画では欠席連絡システムのデータベースから直接欠席情報を参照して電子出席簿に使用する予定だった．しかし，欠席連絡システムのデータベースの仕様上欠席期間が三日以上になると，欠席情報を参照することがうまくいかなかった.

(例)「 1/13~1/15」欠席の場合 → 参照可能な日付「1/13・1/15」のみ．1/14 が参照不可能.

よって，この期間を参照するため新たなテーブルを作成し，欠席情報の分割・整理・統合を行った.

下の図に分解した例を示す．左が従来の欠席連絡システムのデータベース，右が電子出席簿で作成した分割・整理後の欠席情報のテーブルである.

ID	150
学籍番号	y21513
欠席開始日	2023-01-13
欠席最終日	2023-01-15
欠席開始時刻	13:00
欠席終了時刻	15:00
欠席理由	就活

ID	6	7	8
学籍番号	y21513	y21513	y21513
欠席日	2023-01-13	2023-01-14	2023-01-15
欠席開始時刻	13:00	08:50	08:50
欠席終了時刻	16:10	16:10	15:00
欠席理由	就活	就活	就活
個人欠席ID	150	150	150

図 3.5-1　欠席情報分割例

オレンジの部分は欠席した日付となっている．左のテーブルでは 13 日～15 日となっているが，右のテーブルでは 1 日ごとに分割されている．

灰色の部分は欠席した時間となっている．1 日目に 13:00～授業終了時刻，2 日目は 1 日中，3 日目に授業開始時刻～15:00 と欠席の開始・終了時刻を考慮して分割した．

緑の部分は個人欠席 ID となっている．欠席連絡システムでは 1 つの欠席情報であったことを表す．

次にプログラムの説明をする．

```
                                    (省略)
//欠席開始日と欠席終了日に差異がある場合にデータを分割する
$count = 0;
for ($i=0; $i<count($absence_student); $i++) { //欠席人数分ループ
    //欠席日を unixtime 化する
    $start_unixtime[] = strtotime($absence_student[$i][1]); //開始日時
    $end_unixtime[] = strtotime($absence_student[$i][2]); //終了日時
    //欠席している期間（日単位）を数値化する（差を求める）
    $difference_date[] = $end_unixtime[$i] - $start_unixtime[$i];

    //欠席期間により格納するデータを編集する
    if($difference_date[$i] == 0 &&
      Insert_Flag($absence_student[$i][6]) == false){
        //登録するデータを配列に追加
        $strage[$count] = [$absence_student[$i][0],
                    $absence_student[$i][1],$absence_student[$i][3],
                    $absence_student[$i][4], $absence_student[$i][5],
                    $absence_student[$i][6]];
        Insert_Data($count, $strage);
        $count++;
    }
```

①

②

```
if($difference_date[$i] == 86400 &&
  Insert_Flag($absence_student[$i][6]) == false){
    //初日
    $strage[$count] = [$absence_student[$i][0], $absence_student[$i][1],
      $absence_student[$i][3], '16:10',
      $absence_student[$i][5], $absence_student[$i][6]];
    Insert_Data($count, $strage);
    $count++;

    //最終日
    $strage[$count] = [$absence_student[$i][0],
                  $absence_student[$i][2], '08:50',
                  $absence_student[$i][4], $absence_student[$i][5],
                  $absence_student[$i][6]];
    Insert_Data($count, $strage);
    $count++;
}
```

③

```
    if($difference_date[$i] > 86400 &&
     Insert_Flag($absence_student[$i][6]) == false){
        //初日
        $strage[$count] = [$absence_student[$i][0],
                        $absence_student[$i][1],$absence_student[$i][3],
                        '16:10', $absence_student[$i][5],
                        $absence_student[$i][6]];
        Insert_Data($count, $strage);
        $count++;

        //中間日
        for ($j=0; $j<($difference_date[$i]-86400)/86400; $j++) {
            //2日目以降の日付データを取得
            $divide_date = date('Y-m-d' ,strtotime('+1 day',
                         strtotime($absence_student[$i][1])));
            $strage[$count] = [$absence_student[$i][0], $divide_date,
                           '08:50', '16:10',$absence_student[$i][5],
                           $absence_student[$i][6]];
            Insert_Data($count, $strage);
            $count++;
        }

        //最終日
        $strage[$count] = [$absence_student[$i][0],
                        $absence_student[$i][2], '08:50',
                        $absence_student[$i][4], $absence_student[$i][5],
                        $absence_student[$i][6]];
        Insert_Data($count, $strage);
        $count++;
    }
}
```

④

(省略)

リスト 3.5-1　check_system.php

リスト 3.5-1 は欠席期間の日数に応じて欠席データを分割するプログラムである.

①は欠席開始・終了日を unixtime 化[6]してその差を求めることで欠席している日数を計算する処理である. 計算した差の秒数を 1 日の秒数である 86400 秒で割った結果が欠席日数となる.

②は 1 日のみ欠席した場合の処理である. この場合は欠席日を分割する必要がないためそのまま欠席情報テーブルに登録する. また, 登録するデータは, 学籍番号・欠席日・欠席開始時刻・欠席終了時刻・欠席理由・個人欠席 ID である.

③は 2 日間欠席した場合の処理である. この場合では欠席日は開始日・終了日のみなため欠席日で特別必要な処理はない. しかし, 欠席開始時刻・欠席終了時刻は考慮して授業開始・終了時刻を当てはめて欠席情報テーブルに登録する必要がある.

④は 3 日以上欠席した場合の処理である. この場合では欠席初日・最終日は 2 日目と同様の処理だが, 中間日は考慮しなければならない. strototime()関数[7]を用いて日付をずらす処理をすることで中間日を必要な日数分欠席情報テーブルに登録することができる.

```
//インサート済みの欠席データかどうかを調べる
function Insert_Flag($flag_id){
                                        (省略)
        //配列検索のために変数に入れ替える
        $id = "";
        for($i=0; $i<count($absence_id); $i++){
            $id[$i] = $absence_id[$i][0]; //個人欠席 ID 取得
        }
        //個人欠席 ID がテーブルに存在するか
        $flag = in_array($flag_id, (array)$id, true);

        //配列検索をし，一致しなければデータ挿入
        return $flag;
                                        (省略)
}
```

リスト 3.5-2　check_system.php

リスト 3.5-2 は，欠席連絡システムの欠席データがすでに電子出席簿の欠席情報テーブルに存在するか調べるプログラムである．ここでは欠席データが存在するときは true，存在しないときは false を返している．欠席データを分割する処理（リスト 3.5-1）で行われる処理である．

```
//整理した欠席データを Check.php 用にデータベースへ格納
function Insert_Data($i, $strage){
                                        (省略)
        //欠席データを格納する
        $insert_sql = "insert into check_absence_table (student_number,
                    absence_date, absence_start_time, absence_end_time,
                    reason, identity_absence_id)
                    values (:number, :date, :start, :end, :reason, :identity_id);";
        $stmt = $pdo->prepare($insert_sql);
        $stmt -> bindValue(":number", $strage[$i][0], PDO::PARAM_STR); //学籍番号
        $stmt -> bindValue(":date", $strage[$i][1], PDO::PARAM_STR); //欠席日
        $stmt -> bindValue(":start", $strage[$i][2], PDO::PARAM_STR); //開始時刻
        $stmt -> bindValue(":end", $strage[$i][3], PDO::PARAM_STR); //終了時刻
        $stmt -> bindValue(":reason", $strage[$i][4], PDO::PARAM_STR); //欠席理由
        $stmt ->
          bindValue(":identity_id", $strage[$i][5], PDO::PARAM_STR); //個人欠席 ID
        $stmt->execute();
}
                                        (省略)
```

リスト 3.5-3　check_system.php

リスト 3.5-3 は，実際に分割した欠席データを欠席情報テーブルに登録するプログラムである．ここで欠席情報テーブルに登録するデータは，学籍番号・欠席日・欠席開始時刻・欠席終了時刻・欠席理由・個人欠席 ID となっている．

3.6 出欠情報テーブル

出欠情報テーブルへの登録・編集で作成したファイルを表 3.6-1 に示す.

表 3.6-1　出欠情報テーブルへの登録・編集機能ファイル一覧

ファイル名	役割
register_post_time_reason.php	欠席期間・理由取得
register_post_checkbox.php	出欠フラグ取得
register_post_teacher.php	教員名取得
check_data_register.php	出欠テーブルに登録・編集

出欠情報テーブルについて

　このテーブルは出欠確認で送信された出欠情報を登録・編集するテーブルである．データの登録頻度は 1 コマごとであり，学年ごとに出欠情報が登録される．図 3.6-1 にテーブルの構造及び例について示す.

ID	21	33
学籍番号	y21501	y21513
科目ID	5040	5040
授業日	2023-01-13	2023-01-13
何コマ目	1	1
欠席開始時刻	00:00:00	08:50:00
欠席終了時刻	00:00:00	09:30:00
欠席理由		渋滞
教員名	鈴木	鈴木
チェックフラグ	2	1
更新日時	08:53:35	10:21:51

図 3.6-1　テーブルの構造及び例

図 3.6-1 のテーブル構造部分について説明する.

科目 ID は，科目を判別するために使用する．時間割システムの syllabus テーブルと同様の科目 ID が登録されている.

チェックフラグは，0～2 の値で登録される．学生が出席か欠席かどうか一目で判別するために使用される． 0 は欠席， 1 は 45 分出席， 2 は 90 分すべて出席となっている．本校では授業時間の半分の 45 分出席することで出席点を半分取得できるため必要になっている.

更新日時は, テーブルのデータが更新されたときにその時の時間を登録するために使用する. 主に, 無断欠席の学生が欠席連絡をしたときにデータが変更されたのがわかるようにするため追加している. (例) ID21 は出席学生，ID33 は無断欠席した学生である.

出席学生の特徴としては，欠席開始時刻・欠席終了時刻がともに 00:00:00 であり欠席理由が空欄, チェックフラグが 2 で更新日時が授業開始時間に近いことである.

欠席学生の特徴としては，欠席開始時刻・欠席終了時刻が具体的な時間になっていて，欠席理由もあり，チェックフラグが 0 もしくは 1 となっていることである.

また，無断欠席した学生については更新日時が目安になる．欠席連絡が遅れるとデータの更新日時が授業開始時間から遠くなるため更新日時を参考にすることで無断欠席か普通の欠席か判断することができる.

次にプログラムの説明をする.

```
//欠席理由，欠席開始・終了時刻を送る
function Set_Value(count_student){
    Set_Teacher(); //教員名取得
    Check_Click(count_student); //チェックフラグ取得                    ①

    let absence_data = []; //欠席データ配列
    let absence_key = []; //連想配列のためのキー
    for(let i=0; i<count_student; i++){
        let element_start = document.getElementById("start" + i); //開始時刻
        let element_end = document.getElementById("end" + i); //終了時刻
        let element_reason = document.getElementById("reason" + i); //理由
        absence_data.push(element_start.value,
                        element_end.value, element_reason.value);
        absence_key.push('start'+i, 'end'+i, 'reason'+i);
    }                                                                    ②
    const postData = new FormData; // フォーム方式で送る場合
    for(let i=0; i<(count_student*3); i++){
        postData.set(absence_key[i], absence_data[i]); // set()で格納する
    }                                                                    ③

    const data = {
      method: 'POST',
      body: postData
    };

    fetch('register_post_time_reason.php', data)
      .then((res) => res.text())
      .then(console.log);                                                ④

    window.setTimeout(window.location.href = "check_data_register.php", 500);  ⑤
}
```

リスト 3.6-1　rollbook.js

リスト 3.6-1 は欠席理由・欠席期間のデータを postData にして php ファイルへ受け渡す[8]プログラムである．受け渡すファイルは「リスト 3.6-4　register_post_time_reason_.php」である．

①は教員名とチェックフラグ情報を送るための関数である．これは「リスト 3.6-2」,「リスト 3.6-3」で詳細を後述する．

②は欠席理由と欠席期間を出席確認ページから取得する処理である．JS の処理で value を取得することでリアルタイムに変更されたデータが取得できる．

③は出欠情報を postData に追加する処理である．postData にすることで php 側が連想配列で出欠情報を受け取ることができる．

④は出欠情報を postData として php ファイルに受け渡す処理である．

⑤は出欠情報テーブルに出欠情報を登録するためのファイル（check_data_register.php）に遷移するための処理である．setTimeout()関数を利用することで php と JS のデータのやり取りで課題であるデータの受け渡しのタイミングによるエラーを対策している．

```
//教員名を送る
function Set_Teacher(){
        let element_teacher = document.getElementById("teacher_select"); //教員名
        let teacher_data = element_teacher.value;
        const postData = new FormData; // フォーム方式で送る場合
        postData.set('teacher', teacher_data); // set()で格納する

    const data = {
      method: 'POST',
      body: postData
    };
    fetch('register_post_teacher.php', data)
      .then((res) => res.text())
      .then(console.log);
}
```

リスト 3.6-2　rollbook.js

リスト 3.6-2 は教員名のデータを postData にして php ファイルへ受け渡すプログラムである. 受け渡すファイルは「リスト 3.6-5　register_post_teacher_.php」である.

処理の流れとしてはリスト 3.6-1 と同様であるため詳細は省略する.

```
//チェックボックスの状態で送る値が変化
function Check_Click(count_check){
    let start_half = []; //1つ目のチェック　0が欠席，1が半分出席
    let end_half = []; //2つ目のチェック　0が欠席，1が半分出席
    let check_state = []; //チェックフラグ　0が欠席，1が半分出席，2が出席
    const postData = new FormData;
    for(let i=0; i<count_check; i++){
        if(document.getElementById('check1_'+i).checked){ //一つ目のチェック
            start_half.push(1);
        }else{
            start_half.push(0); }
        if(document.getElementById('check2_'+i).checked){ //2つ目のチェック
            end_half.push(1);
        }else{
            end_half.push(0); }
        check_state[i] = start_half[i] + end_half[i]; //チェックフラグに変換
        postData.set('check_flag'+i, check_state[i]);
    }
    const data = {
      method: 'POST',
      body: postData   };
    fetch('register_post_checkbox.php', data)
      .then((res) => res.text())
      .then(console.log);
}
```

リスト 3.6-3　rollbook.js

リスト 3.6-3 はチェックフラグ情報のデータを postData にして php ファイルへ受け渡すプログラムである．受け渡すファイルは「リスト 3.6-6　register_post_checkbox_.php」である．

処理の流れとしてはリスト 3.6-1 と同様であるため詳細は省略する．チェックフラグ情報の決定方法のみは説明する．チェックフラグ情報は出欠確認でのチェックの数をカウントした 0～2 のいずれかの値が入るようになっている．

```
session_start();

$post_count = $_POST;
//post で理由，スタート・終了時間を取得
for($i=0; $i<(count($post_count)/3); $i++){
    $_SESSION['register_start'][$i] = $_POST['start'.$i]; //欠席開始時刻
    $_SESSION['register_end'][$i] = $_POST['end'.$i]; //欠席終了時刻
    $_SESSION['register_reason'][$i] = $_POST['reason'.$i]; //欠席理由
}
```

リスト 3.6-4　register_post_time_reason.php

リスト 3.6-4 は「リスト 3.6-1」から受けとった値を SESSION に追加するプログラムである．

この時 post されてきたデータは連想配列で渡されてくるのでクラス人数の 3 倍のデータが入っている．そのためループの回数に気を付ける必要がある．

```
session_start();
//post で教員名を取得
if(isset($_POST['teacher'])){
    $_SESSION['register_teacher'] = $_POST['teacher']; //教員名
}
```

リスト 3.6-5　register_post_teacher.php

リスト 3.6-5 は「リスト 3.6-2」から受けとった値を SESSION に追加するプログラムである．

```
session_start();
$post_count = $_POST;
//post で理由，スタート・終了時間を取得
for($i=0; $i<(count($post_count)); $i++){
    $_SESSION['check_flag'][$i] = $_POST['check_flag'.$i]; //チェックフラグ
}
```

リスト 3.6-6　register_post_checkbox.php

リスト 3.6-6 は「リスト 3.6-3」から受けとった値を SESSION に追加するプログラムである．

```
session_start();

if(isset($_SESSION['check_data'])){
    $check_data = $_SESSION['check_data'];
    $hours = $check_data['hours_class']; //コマ目
}else{
    echo "<script>Error_Check('false');</script>";
}

if(isset($_SESSION['date_data'])){
    $date_data = $_SESSION['date_data']; //日付
    $date = $date_data['date'];
}else{
    echo "<script>Error_Check('false');</script>";
}

if(isset($_SESSION['register_number'])){
    //学籍番号が配列にある
    $student_number = $_SESSION['register_number']; //学籍番号
}else{
    echo "<script>Error_Check('false');</script>";
}

if(isset($_SESSION['check_register_data'])){
    //科目 ID
    $check_register_data = $_SESSION['check_register_data'];
    $subject_id = $check_register_data['subject_id']; //科目 ID
}else{
    echo "<script>Error_Check('false');</script>";
}
```

```
//教員名，チェックフラグ，欠席理由，欠席開始・終了時刻
if(isset($_SESSION['register_teacher'])){
    $teacher = $_SESSION['register_teacher']; //教員名
}else{
    echo "<script>Error_Check('false');</script>";
}
if(isset($_SESSION['check_flag'])){
    $check_flag = $_SESSION['check_flag']; //チェックフラグ
}else{
    echo "<script>Error_Check('false');</script>";
}
if(isset($_SESSION['register_start'])){
    $absence_time_start = $_SESSION['register_start']; //欠席開始時刻
}else{
    echo "<script>Error_Check('false');</script>";
}
if(isset($_SESSION['register_end'])){
    $absence_time_end = $_SESSION['register_end']; //欠席終了時刻
}else{
    echo "<script>Error_Check('false');</script>";
}
if(isset($_SESSION['register_reason'])){
    $absence_reason = $_SESSION['register_reason']; //欠席理由
}else{
    echo "<script>Error_Check('false');</script>";
}
```

リスト 3.6-7　check_data_register.php

リスト 3.6-7 は出席確認ページや「リスト 3.6-1～3.6-6」までに SESSION に格納されたデータを参照するプログラムとその際に起きる可能性のあるエラー回避のプログラムである．

エラー回避の処理は JS で行っている．詳細は「リスト 3.6-8　rollbook.js」に後述する．

```
//session エラーの確認
function Error_Check(){
    history.back(); //データはそのままに前のページに戻る
    window.addEventListener('load', function(){
        alert('SESSION エラーが起きました．¥n もう一度完了ボタンを押すか¥n
              ページをリロードしてください．');
    });
}
```

リスト 3.6-8　rollbook.js

リスト 3.6-8 は出欠情報テーブルに登録・編集する際に起こる可能性のある SESSION エラーを回避するためのプログラムである．

このエラーは出欠確認ページをリロードすることで直ることがわかっているため，history.back()関数を用いて出欠確認ページに遷移させている．

```
//出欠席データを整理し，テーブルに登録 or 編集をする
$strage = []; //テーブルに登録・編集するデータを格納する配列
for($i=0; $i<count($student_number); $i++){
    //重複データを挿入しない
    if($subject_id == $absence_data[$i][0] && $date == $absence_data[$i][1] &&
      $hours == $absence_data[$i][2]){
        $strage[$i] = [$student_number[$i], $subject_id, $date, $hours,
                    $absence_time_start[$i],$absence_time_end[$i],
                    $absence_reason[$i], $teacher, $check_flag[$i]];
        Update_Data($i, $strage);
    }else{
        $strage[$i] = [$student_number[$i], $subject_id, $date,
                    $hours, $absence_time_start[$i], $absence_time_end[$i],
                    $absence_reason[$i], $teacher, $check_flag[$i]];
        Insert_Data($i, $strage);
    }
}
```

リスト 3.6-9　check_data_register.php

リスト 3.6-9 は出欠情報テーブルに登録するか編集するか判別するプログラムである．

科目 ID と日付とコマ目を用いて出欠データがあるかどうか判断する．出欠データがある場合は編集，ない場合は登録となる．

$strage に登録・編集するデータを格納している．

登録編集するデータは，学籍番号・科目 ID・日付・コマ目・欠席開始時刻・欠席終了時刻・欠席理由・教員名・チェックである．

```
//出欠席データをテーブルに登録
function Insert_Data($i, $strage){
                                        (省略)
        //欠席データを格納する
        $insert_sql = "insert into class_absence_data (student_number, class_id,
                  date, hours, absence_time_start, absence_time_end,
                  absence_reason, teacher, check_flag)
                  values (:number, :class_id, :date, :hours,
                  :start, :end, :reason, :teacher, :flag);";
        $stmt = $pdo->prepare($insert_sql);
        $stmt -> bindValue(":number", $strage[$i][0], PDO::PARAM_STR); //学生番号
        $stmt -> bindValue(":class_id", $strage[$i][1], PDO::PARAM_STR); //科目 ID
        $stmt -> bindValue(":date", $strage[$i][2], PDO::PARAM_STR); //日付
        $stmt -> bindValue(":hours", $strage[$i][3], PDO::PARAM_STR); //コマ目
        $stmt -> bindValue(":start", $strage[$i][4], PDO::PARAM_STR); //開始時刻
        $stmt -> bindValue(":end", $strage[$i][5], PDO::PARAM_STR); //終了時刻
        $stmt -> bindValue(":reason", $strage[$i][6], PDO::PARAM_STR); //理由
        $stmt -> bindValue(":teacher", $strage[$i][7], PDO::PARAM_STR); //教員名
        $stmt -> bindValue(":flag", $strage[$i][8], PDO::PARAM_STR);//チェックフラグ
        $stmt->execute();
                                        (省略)
}
```

リスト 3.6-10　check_data_register.php

リスト 3.6-10 は出欠情報テーブルにデータを登録するプログラムである.

```
//出欠席データをテーブルに編集
function Update_Data($i, $strage){
                                        (省略)
        //欠席データを格納する
        $update_sql = "update class_absence_data set
                    absence_time_start = :start, absence_time_end = :end,
                    absence_reason = :reason, teacher = :teacher,
                    check_flag = :flag where student_number = :number and
                    date = :where_date and hours = :where_hours;";
        $stmt = $pdo->prepare($update_sql);
        $stmt -> bindValue(":start", $strage[$i][4], PDO::PARAM_STR); //開始時刻
        $stmt -> bindValue(":end", $strage[$i][5], PDO::PARAM_STR); //終了時刻
        $stmt -> bindValue(":reason", $strage[$i][6], PDO::PARAM_STR); //理由
        $stmt -> bindValue(":teacher", $strage[$i][7], PDO::PARAM_STR); //教員名
        $stmt -> bindValue(":flag", $strage[$i][8], PDO::PARAM_STR);//チェックフラグ
        $stmt -> bindValue(":number",$strage[$i][0],PDO::PARAM_STR); //学生番号
        $stmt -> bindValue(":where_date",$strage[$i][2],PDO::PARAM_STR); //日付
        $stmt -> bindValue(":where_hours",$strage[$i][3],PDO::PARAM_STR); //コマ目
        $stmt->execute();
                                        (省略)
}
```

リスト 3.6-11　check_data_register.php

リスト 3.6-11 は出欠情報テーブルのデータを編集するプログラムである．学籍番号と日付とコマ目で特定の学生だけを編集するようにしている．

3.7 確認メール

確認メールで作成したファイルを表 3.7-1 に示す.

表 3.7-1　確認メール機能ファイル一覧

ファイル名	役割
mail.php	メール送信

確認メールについて

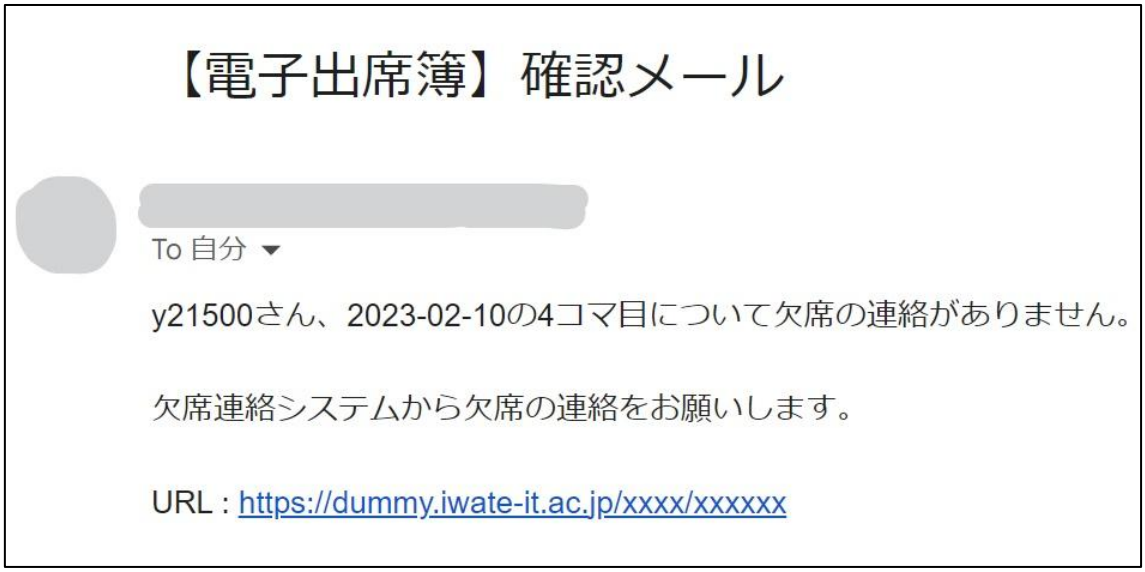


図 3.7-1

確認メールとは，出欠確認ページで無断欠席していると判断された学生に対して欠席連絡を促すメールである．学生一人一人に割り振られたメールアドレスを使用して送信する．欠席連絡を促すために欠席連絡システムの URL をメールの内容に含めている．また，同様の内容を教員にも送信する．

1 行目に学籍番号・日付・コマ目でいつだれが休んでいるかがあり，3 行目に欠席連絡システムの URL を挿入している．

次にプログラムを説明する．

```
                                    (省略)
$today = Date("Y-m-d-H:i:s");
$text_s = "";    //メール本文

for($i=0; $i<count($mail_user); $i++){
    $to_mail[] = $mail_user[$i][0]."@iwate-it.ac.jp";    //宛先
    $from_mail[] = $to_mail[$i];  //送信元
    $from[] = $mail_user[$i][0];     //送信者名
}
try{
    //無断欠席した学生本文
    for($i=0; $i<count($mail_user); $i++){ //無断欠席学生の人数ループ
        $text_s .= $mail_user[$i][0]."さん，".$mail_user[$i][1]."の"
                .$mail_user[$i][2]."コマ目について欠席の連絡がありません
                . ¥r¥n¥r¥n";
        $text_s .= "欠席連絡システムから欠席の連絡をお願いします．¥r¥n¥r¥n";
        $text_s .= "URL :
                 https://dummy.iwate-it.ac.jp/xxxxx/xxxxxx.php ¥r¥n";
```

①

②

```
        //学生用メール
        $chack = send_mail("【電子出席簿】確認メール", $text_s, $from[$i],
                $to_mail[$i], $from_mail[$i]);
        if(!$chack){
            print "メール送信エラー:". $today . "¥r¥n" .
                 "送信元:" . $from_mail[$i] ."¥r¥n" .
                 "送信先:" . $to_mail[$i] . "¥r¥n";
        }
        //教員用メール                                                    ③
        $chack_teacher = send_mail("【電子出席簿】確認メール - "
                        .$from[$i], $text_s, $from[$i],
                        "cis@ml.iwate-it.ac.jp", $from_mail[$i]);
        if(!$chack_teacher){
           print "メール送信エラー:". $today . "¥r¥n" .
                "送信元:" . $from_mail[$i] ."¥r¥n" .
                "送信先:"cis@xx.iwate-it.ac.jp ¥r¥n";
        }
        $text_s = "";
    }
                                (省略)
```

リスト 3.7-1　mail.php

リスト 3.7-1 はメールの文章とメールの送信元・宛先を作成するプログラムである.

①は送信元・宛先を取得する処理である．電子出席簿の出欠情報テーブルから無断欠席した学生のみを抽出して学籍番号を取得している.

②はメール本文を作成する処理である．$text_s がメール本文の変数となっている.

③はメールを送信する処理である．send_mail()関数を用いることでメール送信を行っている.

このメール送信の一連の流れや関数に関しては参考文献[1]があるため，そちらを参考した.

3.8 エクスポート

図 3.8-1，図 3.8-2 はファイルエクスポート画面である．

図 3.8-1　ファイルエクスポートページ(左)　図 3.8-2　レスポンシブ対応(右)

この画面では，年度ごとの欠席情報を CSV ファイルに出力することができる．年度選択ボックスは，現在の年度からアクセスした日付の年度を選択することができる．そして，「ファイル出力」を押下することで選択した年度の欠席情報の CSV ファイルが作成される[9]．すでに同じ年度の CSV ファイルが存在する場合，「ファイル出力」を押下すると上書き確認画面に遷移する．

次にプログラムを説明する．

```
                                        (省略)
sql = "SELECT * FROM class_absence_data" ;
$stmt = $pdo->prepare($sql);
$stmt->execute();  ①

$stmt2 = $pdo->prepare("select id, subject, flag from syllabus;");
$stmt2->execute();
$syllabus = $stmt2->fetchAll(PDO::FETCH_BOTH);  ②
                                        (省略)
$filename = './csv/欠席データ_'.$_SESSION['year'].'年度.csv';
                                        (省略)
if(file_exists($filename)) {
      echo "<!DOCTYPE html>";
      echo "<html lang = ¥"ja¥">";
      echo "<head>";
      echo "<meta charset=¥"UTF-8¥">";
      echo "<title>TOP</title>";
      echo "<link rel=¥"stylesheet¥" href=¥"rollbook.css¥" />";
      echo "<meta name=¥"viewport¥"
           content=¥"width=device-width, initial-scale=1.0¥">";
      echo"</head>";
      echo"<body>";
      echo"<header action=¥"¥" method=¥"POST¥" class=¥"TOP-header¥">";
 ③    echo"<h1 class=¥"title¥">上書き確認</h1>";
      echo"</header>";

      echo"<div class=¥"Create_csv¥">";
      echo"<form method=¥"POST¥" action=¥"create_csv_system.php¥"
           class=¥"csv_select¥">";
      echo "<h1>既にファイルが存在します<br>上書きしますか？</h1>";
      echo "<button type=¥"submit¥" class=¥"create_button¥" name=¥"write¥" >
           上書き</button>";
      echo"<button type=¥"submit¥" class=¥"create_button¥" name=¥"return¥" >
           TOP に戻る</button>"
```

```
    echo"</from>";
    echo"</div>";
③   echo"</body>";
    echo"</html>";
```

リスト 3.8-1　create_csv_system.php①

まず，$stmt で欠席情報テーブルのデータを取り出す準備(①)を，$syllabus で時間割システムのシラバステーブルから科目 ID，科目名，フラグを取得(②)する．

その後，欠席情報ファイルを保存しているフォルダ内に同じ年度(名前)ファイルが存在する場合には，上書き確認画面を表示(③)する．

```
if(isset($_POST['write'])){
    if(unlink('./csv/欠席データ_'.$_SESSION['year'].'年度.csv')){
       echo "成功";
    }
    if($fp = fopen($filename,'w+')) {
       if (is_array($absence)) {
           $count=count($absence);
       }
    while($absence = $stmt->fetch(PDO::FETCH_ASSOC)){
       if (is_array($syllabus)) {
           $co=count($syllabus);
       }
       for($i=0;$i<$co;$i++){
           if($absence['class_id']==$syllabus[$i]['id']){
               $class=$syllabus[$i]['id'];
           }
       }
       if($count==0){
           array_push($csv, "学籍番号");
           array_push($csv, "科目");
           array_push($csv, "出欠席日時");
```

①

```
        array_push($csv, "出欠席時限");
        array_push($csv, "出欠席情報");
        array_push($csv, "欠席開始時刻");
        array_push($csv, "欠席終了時刻");
        array_push($csv, "欠席理由");
        mb_convert_variables("SJIS", "UTF-8", $csv);
        fputcsv($fp, $csv);
        $csv = [];
        $count = $count+1;
    }

    if($year_start<=$absence['date']&& $year_end>=$absence['date']){
        array_push($csv, $absence['student_number']);
        array_push($csv, $absence['class_id']);
        array_push($csv, $absence['date']);
        array_push($csv, $absence['hours']);
        array_push($csv, $absence['check_flag']);
        array_push($csv, $absence['absence_time_start']);
        array_push($csv, $absence['absence_time_end']);
        array_push($csv, $absence['absence_reason']);
    }else{
        continue;
    }
    mb_convert_variables("SJIS", "UTF-8", $csv);
    fputcsv($fp, $csv);
    $csv = [];
}
header('Location: File_download.php?
        message=scv ファイルの出力に成功しました. ');
exit();
echo "<pre>";
echo print_r($_POST);
echo "</pre>";
}else{
```

①

②

```
            echo 'ファイルの書き込みに失敗しました';
            exit;
        }
    }
    else if(isset($_POST['return'])){
        header('Location: TOP.php');
        exit();
}
```

リスト 3.8-2　create_csv_system.php②

ファイル出力時に選択した年度のファイル存在しない場合か，上書き確認画面で上書きボタンを押下すると，CSV ファイル出力処理を実行する．ファイル出力処理として，まず 1 行目に CSV ファイルを Excel で閲覧する際，分類ごとに分かりやすいような文字列を挿入(①)する．２行目以降からは，選択された年度の欠席データを 1 行ずつ取り出して挿入(②)する．

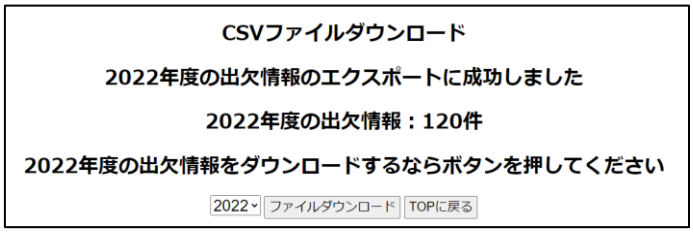
また，出欠情報がデータベースから CSV ファイルに挿入されるときに使用される言語は「SJIS」であるが，その言語だと Excel で CSV ファイルを開いた際に文字化けしてしまう．そのため，CSV ファイルに出欠データを挿入する前にそのデータの言語を「UTF-8」に変更する．

出力された CSV ファイルは図 3.8-3 の通りである．

1	学籍番号	科目	出欠席日時	出欠席時限	出欠席情報	欠席開始時刻	欠席終了時刻	欠席理由
2	y21501	5016	2023/1/17	1	0	9:30:00	10:20:00	通院・予防接種等
3	y21502	5016	2023/1/17	1	0	9:30:00	10:20:00	通院・予防接種等
4	y21503	5016	2023/1/17	1	0	8:50:00	10:20:00	その他
5	y21504	5016	2023/1/17	1	0	8:50:00	10:20:00	その他
6	y21505	5016	2023/1/17	1	0	8:50:00	10:20:00	電車遅延・渋滞等
7	y21506	5016	2023/1/17	1	1	8:50:00	9:10:00	電車遅延・渋滞等
8	y21507	5016	2023/1/17	1	2	0:00:00	0:00:00	
9	y21508	5016	2023/1/17	1	2	0:00:00	0:00:00	
10	y21509	5016	2023/1/17	1	2	0:00:00	0:00:00	
11	y21510	5016	2023/1/17	1	2	0:00:00	0:00:00	
12	y21511	5016	2023/1/17	1	2	0:00:00	0:00:00	
13	y21512	5016	2023/1/17	1	2	0:00:00	0:00:00	
14	y21513	5016	2023/1/17	1	2	0:00:00	0:00:00	
15	y21514	5016	2023/1/17	1	2	0:00:00	0:00:00	

図 3.8-3　エクスポートした CSV ファイルの例

次にファイルダウンロードについて説明する.



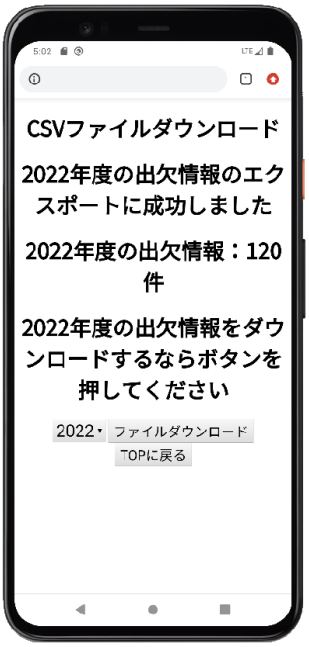


図 3.8-4　ファイルダウンロードページ(左)　図 3.8-5　レスポンシブ対応(右)

前述したファイル出力画面・ファイル上書き確認画面からファイル出力処理を行うと自動的にこの画面に遷移する.

```
$filename = ''.__DIR__.'/csv/欠席データ_'.$fileyear.'年度.csv';
download($filename);

function download($pPath, $pMimeType = null)
{
    //-- ファイルが読めない時はエラー
       (もっときちんと書いた方が良いが今回は割愛)
    if (!is_readable($pPath)) { die($pPath); }

    //-- Content-Type として送信する MIME タイプ
       (第 2 引数を渡さない場合は自動判定) ※詳細は後述
    $mimeType = (isset($pMimeType)) ? $pMimeType
                : (new finfo(FILEINFO_MIME_TYPE))->file($pPath);

    //-- 適切な MIME タイプが得られない時は,
        未知のファイルを示す application/octet-stream とする
    if (!preg_match('/¥A¥S+?¥/¥S+/', $mimeType)) {
        $mimeType = 'application/octet-stream';
    }

    //-- Content-Type
    header('Content-Type: ' . $mimeType);

    //-- ウェブブラウザが独自に MIME タイプを判断する処理を抑止する
    header('X-Content-Type-Options: nosniff');

    //-- ダウンロードファイルのサイズ
    header('Content-Length: ' . filesize($pPath));

    echo setlocale(LC_ALL, 0); // C

    setlocale(LC_ALL,'C.UTF-8');
    //-- ダウンロード時のファイル名
    header('Content-Disposition: attachment;
          filename="' . basename($pPath) . '"');
```

①

```
//-- keep-alive を無効にする
    header('Connection: close');

    //-- readfile()の前に出力バッファリングを無効化する ※詳細は後述
    while (ob_get_level()) { ob_end_clean(); }

    //-- 出力
    readfile($pPath);

    //-- 最後に終了させるのを忘れない
    exit;
}
```

①

リスト 3.8-3　file_download_system.php

このプログラムでは，$fileyear に選択された年度を挿入しその年度の CSV ファイルが存在する場合そのファイルをダウンロード(①)するプログラムである.

関数 download では，第２引数を渡すことで MIME タイプを指定することができるが今回は使用しない.

第4章 評価

4.1 セキュリティ

セキュリティ部分では，ログイン機能のユーザーIDをファイル名に，パスワードをファイルの中身に挿入したテキストファイルを用意して，入力された情報がそれらに一致すればアクセスできる機能を作成した．テキストファイルを参照するので，ファイル名や中身を編集することでユーザーIDやパスワード変更でき，同じようなファイルを作成することでアクセスユーザーを増やすこともできる．

問題点としてこれらのテキストファイルは，プログラムファイルがある場所のフォルダ中に置かれているので，あまりセキュリティとして好ましくないという点がある．

4.2 出欠確認

スマホやタブレット端末で出欠確認を行うことを想定してレスポンシブ対応を工夫し，ユーザーにわかりやすく表示させることができた．具体的には，スマホでの表示になると出欠確認のテーブルを折りたたんで表示し，横方向へのスライドを行わないようにすることでユーザーに配慮した作りをすることができた．

また，試用をした際に発覚したもので，チェックボックスを一つだけチェックすると無断欠席ではない出席している学生にも欠席連絡の確認メールが送信されてしまうことがあった．これは，電子出席簿システムの仕様上，二つチェックをしないといけないことを教員に周知しできなかったことが原因であった．周知を行うことでメール送信は回避できるためしっかり周知は行いたい．また，このチェック部分はユーザー側からすると説明されないと分かりづらくなってしまっている部分である．しかし，本校の出欠確認と電子出席簿システムでの要求を満たすにはこの仕様でないといけないため仕方がない部分になっている．ここをユーザーに配慮した形にできないか来年度以降に模索してほしい．

4.3 メール確認

メール機能では，当初の目標としていた「無断欠席学生への欠席連絡の催促」についてはこの機能の実装で達成できた．学生はメールの確認を毎日行うようにしているため欠席連絡を自分がしているかどうかを分かりやすくすることができたと考えている．

しかし，メールの文面ではユーザーにあまりいい印象を持てない文面でメール送信を行っていたことは反省点である．これに関しては，文面中に「心当たりのある方は〜」などの柔らかい表現を行うべきであったと考えている．

4.4 エクスポート

CSV ファイル出力する範囲として，当初は先生方が管理しやすいように CSV ファイル出力を科目ごとに行おうとした．たが先生からは，科目ごとにファイルがあるとその管理に時間や負担がかかる，といわれこれでは負担を増やしてしまうことに気づいた．そこで，年度ごとの出欠情報を CSV ファイルに出力することでファイル管理をしやすくするようにした．そして CSV ファイルを Excel で回覧することでソートを行ってもらうようにした．

また，今回ファイル出力する際に年度ごとの出欠情報をデータベースから参照して出力しているがその際にリレーションにより，氏名や科目名を紐づけると良いと先生からアドバイスを頂いた．

第5章 終わりに

今回の電子出席簿作成に関しては以下の要件を実装できた.

⑴時間割から各出欠席データの挿入

⑵年度ごとに分けた出欠席データの CSV ファイル出力・ダウンロード

⑶無断欠席者への確認メール通知

出席簿の電子化，出欠席情報の管理の容易化，学生の欠席連絡促進これらのことに考慮し電子出席簿を作成できた.

しかし，本番環境でテスト及び試用を行ったところ，多くのバグが起きてしまっている．我々もバグの解消に時間の許す限り取り組み続けたが，完全な解消には至らなかった．そのため，来年度以降は電子出席簿を現在よりも安定した挙動をするように改良に努めてほしい.

第 6 章　参考資料

[1] PDF 自動生成とメール通知を利用した欠席連絡システムの開発(産業技術短期大学校矢巾校卒業研究報告書.令和 2 年度.河原烈土.佐々木郁貴.山本綺音)

[2] WEB データベースを用いた時間割アプリの作成(産業技術短期大学校矢巾校卒業　研究報告書.令和 3 年度.佐々木理帆.村井弘大.若松千波)

[3]【PHP】データベースを使わずにログイン機能を作ってみる

https://qiita.com/keisuke-okb/items/7e7f25bd78705a06d5e0

[4] JavaScript | チェックボックスを一括指定・解除する

https://1-notes.com/javascript-change-all-checkbox/

[5]【PHP 入門】徹底解説！PHP でセッションを使う方法〜基本から応用まで

https://www.sejuku.net/blog/25276

[6] PHP で日、月、年の加算と減算方法 – Qiita

https://qiita.com/thiagomatsui/items/619775a96dce38bc5060

[7]【PHP 入門】strtotime を使った UNIX タイムスタンプの取得方法まとめ

https://www.sejuku.net/blog/27815

[8] Fetch API で PHP にデータを送信する方法 – Qiita

https://qiita.com/te-k/items/cb22d0f49b941637cffb

[9] PHP で CSV ファイルを出力する方法

https://onl.sc/1PUqV6p