茨城工業高等専門学校

# 研　究　彙　報

第　58　号

令和５年４月

RESEARCH REPORTS

OF

NATIONAL INSTITUTE OF TECHNOLOGY(KOSEN),

IBARAKI COLLEGE

NO. 58

APRIL 2023

茨城工業高等専門学校

# 茨城工業高等専門学校研究彙報　第５８号

## 目　　次

# GAS(Google Apps Script)を用いた寮生活支援 Web アプリ開発
## ―検温報告入力支援と部屋換え状況表示に関して―

吉成　偉久

# Development of KOSEN Dormitory Life Support Web Application using GAS (Google Apps Script)
## ― Regarding Body Temperature Input Support and Room Movement Status Display―

YOSHINARI Takehisa

**Abstract：** Using GAS (Google Apps Script), it is possible to support body temperature input for KOSEN dormitory students and display room movement status. The dormitory students will be authenticated with KOSEN Gmail account, which they will use to log in to their Gmail account. Since the system uses authenticated information sharing, only dormitory students can connect to the system, and the system can be operated with a certain degree of information security. In temperature entry, the status for the past two weeks is listed, allowing the user to check items that have not been entered and complete them at the same time. The room movement status display allows the user to check my order and to easily see how far the room movement has been completed.

## 1. はじめに

日本において，新型コロナウイルス感染症(COVID-19)に対して，東京・神奈川・大阪・福岡など 7 都府県を対象に 2020 年 4 月 7 日，初の緊急事態宣言が出された。その後，4 月 16 日には対象は全国に拡大されている。これに伴い文部科学省から，2020 年 5 月 22 日付で，幼小中高・特別支援学校向けに，学校の衛生管理の観点から『学校における新型コロナウイルス感染症に関する衛生管理マニュアル～「学校の新しい生活様式」～』(以下，学校の新しい生活様式)が公表されている。COVID-19 の最新の知見に基づき改訂が続けられ，2022 年 4 月 1 日には Ver.8[1]が公表されている。

全国の学校では，この学校の新しい生活様式[1]を参考にした運営を行ってきている。「第１章の５．設置者及び学校の役割」では，「朝の検温」をはじめとする保健管理体制の維持を目的とした健康観察などが示されている。

### 1.1. 本校学寮での健康観察

本校学寮では，在寮生の健康観察の一環として，2020 年 4 月から Google フォーム[4]による毎朝の検温報告を義務化している。毎日の検温報告を通して，寮生自らに，自分の体調を把握する重要性を認識させている。COVID-19 の蔓延を防ぐため，37.5℃以上の体温になったら，保護者お迎えにより，自宅への帰宅を促してきた。

小中高校では学級担任などが対面で行う健康観察を実施している。一方，本校では，朝の会を新学期初日や新年初日に行うのみである。このため，担任が行う健康観察は体調不良者のみを対象者としている。本校学寮での健康観察は，寮生数が 200 名前後となるため，Google フォームによる検温報告により行っている。通常時は対面での確認（学寮での対面での検温報告の状況確認は，開寮日だけ実施）が行えない。残念ながら未報告者が多数発生し，学校管理者が行う検温未報告者への個別フォローが大きな負担となっていた。

この問題を解決するために，Google フォームによる検温報告に GAS(Google Apps Script)[2]を活用した機能を導入する。GAS を導入することで，寮生に対しては検温入力支援を，学校管理者には，検温未報告者へのフォロー実現を目的とした。

### 1.2. GAS(Google Apps Script)とは

GAS とは，Gmail や Google スプレッドシートと連係して動作する JavaScript をベースとしたプログラミング言語である。GAS を利用することで，Google 上で提供されているサービスの連携や自動化が可能となる。GAS の基本的な文法は，JavaScript に準じている。JavaScript を学習済みなら，ほとんどそのまま利用できる。通常の JavaScript は Web ページに動的な変化を起こさせるために，クライアントサイドで動作する。一方，GAS で利用される JavaScript は，Google のサーバ上で動作するサーバサイドな言語である。

基本的に無料で利用できる上に，以下に示すセキュリティ対策が可能となる。

(1) Gmail アカウントで認証された状態でのみ利用可能で，本校の場合全学生に独自の Gmail アカウント（以下，高専 Gmail）を付与している（2.1 節参照）。
(2) GAS の実行時の設定で，アクセスできるユーザーをさらに制限できる。本校の場合，高専 Gmail で認証済みの場合に制限できる（図 6 参照）。
(3) Google クラウド上のサービスが利用でき，本クラウドサービスは，世界中の数十億のユーザー向けにセキュリティを最優先[3]として運営されている。

### 1.3. その他の COVID-19 対応

学校の新しい生活様式[1]の「第6章 寮や寄宿舎における感染症対策」では，学寮における基本的な感染対策の方針が示されている。学寮は，集団生活を営む場であるため，三密対策などが示されている。本校学寮でも，同様の対策を行っている。

様々なできうる限りの対策を行っている。そのなかで，新たな対策が必要となったものに，学期末や学年末に実施する部屋換え（使用していた部屋を空にして，新しい部屋に私物を移動）である。部屋換えの実施時には，人流が大きくなるため，対策が重要である。COVID-19 以前の部屋換えは，全寮生が一斉に荷物の移動（並列方式）をしていた。

COVID-19 以降は，なるべく人流を減らすため，移動先の部屋が空き室になったら私物の移動を開始する（直列方式）方式に改めた。

COVID-19 対応のために対応が必要となった新方式の部屋換えについても，寮生と学校管理者の双方にとって，負担軽減となることを目的として，GAS を活用した Web アプリを開発する。

## 2. 毎朝の検温報告について

### 2.1. 検温報告用 Google フォーム

本校における寮生の検温報告は，Google フォームを用いて実施してきている。本校学生は，茨城高専 Gmail アカウント[5]の発行を受け，これを用いて Google クラスルーム[6]などによる情報共有が行われている。また，日頃から Google クラスルームや Google フォームを利用している。

図 1 に，寮生向けの検温報告 Google フォームの入力画面を示す。


**図 1 今日の体温(Google フォーム)**

Google フォームでは，図 1 に示すように，設定した設問に応じて情報の入力が行える。図 1 では，最後の設問で，今朝の体温を数値で入力する。入力した情報は，Google スプレッドシートと連携することもできる。Google フォームで入力した情報は，連係させた Google スプレッドシートの「フォームの回答 1」(シート名は変更可能)シートに展開される（図 2 参照)。

なお，本 Google フォームにおいては，「設定タブ」－「回答」において

- 「メールアドレスを収集する」が ON
- 「茨城工業高等専門学校と信頼できる組織のユーザーに限定する」が ON

とされていることを条件とする。

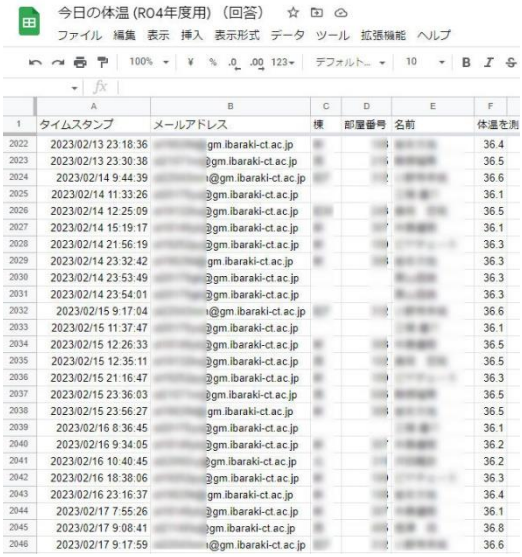



**図 2 Google フォームと連携したスプレッドシート**

### 2.2. 寮棟毎の検温データ入力状況集計シート

Google スプレッドシートは，Google フォームと連携したシート以外に，新規シートが追加できる。追加したシートに，スプレッドシート用関数を組み合わせて，図 3 に示す集計表が構築できる。

さらに，Google スプレッドシートを利用する利点として，Microsoft Excel には実装されていない関数である「QUERY 関数[7]」が利用できることである。図 3 に示したシートの A 列から G 列の内容は，「A2 セル」に入力された，次に示すスプレッドシート QUERY 関数により実現されている。

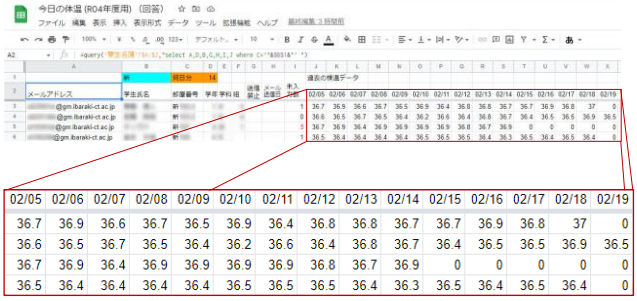
```
=QUERY('寮生名簿'!$A:$J,"select A,D,B,G,H,I,J
where C='"&$B$1&"' ")
```

QUERY 関数の優れている点は，データベース用 SQL 言語に準じた操作が実現できることである。上記の QUERY 関数内の「select, where」文によって，条件に合致した内容がすべてシート上に展開される。このため，図 3 の A 列から G 列の内容は，手入力することなくそれぞれのセルに適切な内容が QUERY 関数によって「寮生名簿」シートから選択的にコピーされる。

なお，汎用性を持たせるために，図 3 に示したシートの「B1」セルに建物名を配置した。これを参照する形で，B1 セルに示した建物に入居している寮生のデータだけを QUERY 関数を用いて表示させている。

さらに，図 3 に示したシートの「J～X」列に示された検温データは，次に示すスプレッドシート関数で表示させている。

```
=MAX(MAXIFS('フォームの回答'!$F$2:$F,'フォームの回答'!$B$2:$B,$A3,'フォームの回答'!$A$2:$A,">="&J$2,'フォームの回答'!$A$2:$A,"<"&(J$2+1)),MAXIFS('追加用'!$C$2:$C,'追加用'!$B$2:$B,$A3,'追加用'!$D$2:$D,">="&J$2,'追加用'!$D$2:$D,"<"&(J$2+1)))
```



| 02/05 | 02/06 | 02/07 | 02/08 | 02/09 | 02/10 | 02/11 | 02/12 | 02/13 | 02/14 | 02/15 | 02/16 | 02/17 | 02/18 | 02/19 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 36.7 | 36.9 | 36.6 | 36.7 | 36.5 | 36.9 | 36.4 | 36.8 | 36.8 | 36.7 | 36.7 | 36.9 | 36.8 | 37 | 0 |
| 36.6 | 36.5 | 36.7 | 36.5 | 36.4 | 36.2 | 36.6 | 36.4 | 36.8 | 36.7 | 36.4 | 36.5 | 36.5 | 36.9 | 36.5 |
| 36.7 | 36.9 | 36.4 | 36.9 | 36.9 | 36.9 | 36.9 | 36.8 | 36.7 | 36.9 | 0 | 0 | 0 | 0 | 0 |
| 36.5 | 36.4 | 36.4 | 36.4 | 36.4 | 36.5 | 36.5 | 36.5 | 36.4 | 36.3 | 36.5 | 36.4 | 36.5 | 36.4 | 0 |

**図 3 寮棟毎の検温データ状況表示（２週間分）**

上記のスプレッドシート関数は，代表で，「J3」セルに入力されている関数を示している。図 3 に示したシートの「J2～X2」セルに，スプレッドシートを開いている当日の日付から 2 週間前までの日付データを表示させ，これに該当する日付の検温データの最高値のみを表示させている。Google フォームと連係してある「フォームの回答」シート（図 2 参照）上から，A3 セル内の Gmail アカウントと一致した寮生の検温データを，J2 セルの日付に限定して，最大値を表示させる。さらに， 図 17 で詳細は説明するが，「追加用」シート内の検温データからも同様に J2 セルの日付限定で表示させ，入力忘れ集計表に反映できるようにしている。

### 2.3. Google フォームを利用する場合の問題点

前節の 2.2 節で，Google フォームとそれと連係した Google スプレッドシートを用いることで，各寮生の 2 週間分の検温データ一覧を確認することができる（図 3 参照）。しかし，これはあくまでも Google フォームを作成した学校管理者側の利点でしかない。寮生にとっては，

(1) 当日の体温しか入力できない
(2) 検温報告を忘れると，前日の体温でも入力できない
(3) 手元で体温の集計表を用意しないと，自分の体温の推移が不明
(4) どの日が未報告となっているか全く確認できない
(5) 寮生にとって義務だけの行為であるが，利点を感じにくい

との欠点を抱えることになる。

学校管理者側には，

(1) 検温未報告は「0」となり，未報告数のカウント表示のみ
(2) フォームで入力した寮生の高専 Gmail を収集しているが，シート間のデータ連係用途のみで，その他に活用できていない
(3) 未報告者へフォローのためのメール送信などは，手作業となる
(4) 負担ばかりが増えて，健康観察の遂行が難しくなる

との欠点を抱えることになる。

### 2.4. 問題点解決のための GAS 導入方法

前節の 2.3 節で説明した通り，Google フォームの問題点は，標準の機能を利用していただけでは解決できない。Google フォームと連係させた Google スプレッドシート上で GAS が利用できる。多くの問題点が解決できるように GAS によるスクリプトを作成する。GAS が利用できるようになるまでの手順を以下に示す。以下の手順は，Google フォームの「回答」タブで「スプレッドシートで表示」をクリックしていることを前提とする。

(1) 「拡張機能」－「Apps Script」とメニュー操作する
(2) 図 4 に示すような画面がブラウザの新しいタブとして開かれる
(3) 画面右上の「デプロイ▽」をクリック後，現れたメニューで「新しいデプロイ」を選びクリックする
(4) 図 5 に示すダイアログが表示されるので，左上の「種類の選択」にある「歯車」のアイコンをクリックして，「ウェブアプリ」を選択
(5) 図 6 に示す画面に切り替わる
(6) 図 6 で，「説明」に実行目的を表す名称を，「次のユーザーとして実行」に自分のアカウントを，「アクセスできるユーザー」に「茨城工業高等専門学校 内の全員」を選択し，「デプロイ」ボタンをクリックする
(7) 続く画面で，「アクセスを承認」ボタンをクリックして，自分のアカウントを選んでいく，「Allow」ボタンをクリックする

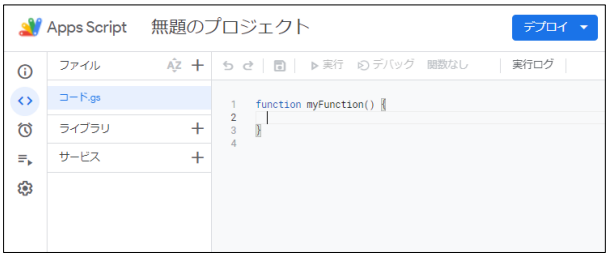



**図 4 無題のプロジェクト**


**図 5 新しいデプロイ，種類の選択**


**図 6 新しいデプロイ，種類の選択と設定**

図 4 に示す画面は，GAS のためのスクリプトを記述するエディタ画面としても機能している。初期状態は，関数の骨組みだけで何も中身のない自作関数である「myFunction」がある状態となる。myFunction は，function キーワードに続き記述される。

GAS を利用する上で，必要なのは Google Chrome ブラウザソフトだけである。一般的な開発言語に必要な統合開発ツール等は不用となる。myFunction 自体は，別の名前に変更することも可能である。

なお，利用する Google スプレッドシート内に閉じた動作となる自作関数の場合は，上記の手順で示したデプロイは不用である。デプロイは，本論文の主要な内容である Web アプリの動作を可能とするために必要な設定である。スプレッドシート内で閉じた動作を実現させる自作関数は，Microsoft Excel の VBA(Visual Basic Application)[8]と対比で比較できる。

## 2.5. GAS による自作関数と Web アプリ

GAS で自作関数を作成するには，以下に示す「function」キーワード（図 4 参照）を利用する。

```
function 自作関数名 {
    // 自作関数の中身
}
```

function キーワードを用いて自作関数を記述した内容は，標準では図 4 に示すように「コード.gs」ファイルに展開される。スクリプトの入力をした後は，必ず「Ctrl+S」キー操作をするか，画面上部 ボタンをクリックして保存しなければならない。Google スプレッドシートは，自動保存されるが，GAS で記述したスクリプトは自動保存されない。

GAS を用いた Web アプリは，2.4 節で紹介したデプロイの種類設定で，「ウェブアプリ」を設定（図 6 参照）していれば利用できる。具体的には，「デプロイ」ボタンをクリックし「デプロイを管理」をクリックすると，図 7 に準じたダイアログが表示される。


**図 7 デプロイを管理**

ここで，「ウェブアプリ」の下にある「URL」について説明する。

本 URL は，以下に示す形式に準じた URL（以下，Web アプリ URL）を Google Chrome ブラウザで開くと，Web アプリとして動作する。

アクセスできるユーザーを制限している場合

```
https://script.google.com/a/macros/gm.ibaraki-ct.ac.jp/s/abcdefghijklmnopqrstuvwxyz1234567890/exec
```

ウェブアプリ ID

上記で，「https://script.google.com/a/macros/」に続く「gm.ibaraki-ct.ac.jp/」の部分が，図 6 で指定した「アクセスできるユーザー」の部分である。今回は，本校寮生に限定したサービスであるため，セキュリティ向上のため設定した。この後の朱書きの部分が，Web アプリのためにデプロイで生成される部分（ウェブアプリ ID）である。朱書きの部分はあくまでもサンプルである。

さて，Web アプリ URL を Google Chrome ブラウザソフトで閲覧した場合の処理の流れを図 8 に示す。

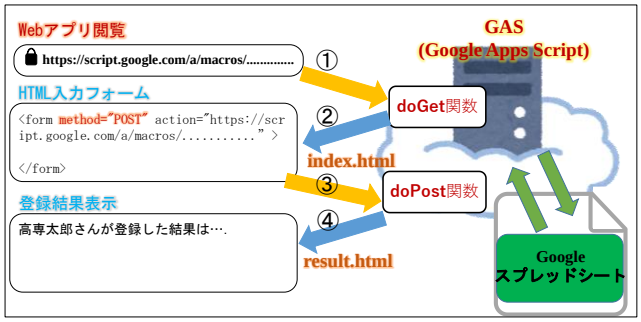



**図 8 GAS による Web アプリ処理の流れ**

図 8 に示した処理の手順は以下の通りである。

① Web アプリ URL で閲覧すると，GAS 側の「コード.gs」内で記述された「doGet 関数」が呼び出されて実行される。

② doGet 関数では，連携したスプレッドシートをデータベースのように扱い，必要な情報を読み取り，レスポンスとして「index.html」をブラウザに戻す。

③ これにより，ブラウザは，index.html を表示させて，その中に仕組まれている HTML 入力フォームを機能させる。HTML の form 文で，「method="POST"」として，Web アプリ URL を呼び出すと，GAS 側のコード.gs 内に記述された「doPost 関数」が呼び出されて実行される。

④ doPost 関数で受け取った HTML 入力フォーム内の情報を，スプレッドシート内に反映させて，登録結果として「result.html」をブラウザに返す

上記の流れにより，最終的にブラウザソフト上には，result.html による登録結果が表示されて，一連の手順が終了する。

なお，上記で示した②の index.html や④の result.html は，固定ではなく，スクリプトを作成する側で決められる。doGet 関数と doPost 関数は，GAS 側の仕様で固定となる。

## 2.6. GAS を用いた学校管理者側のための自作関数

学校管理者側にとっては，検温未報告者にフォローが必要となる。このフォローのために必要な情報をまとめたシートを追加する。図 3 に示したシートと同様に，追加したシートを図 9 に示す。

検温未報告者の収集は，図 9 上に 検温未達者抽出 ボタンを作成して，これをクリックすることで，本シート上に展開されるようにした。本ボタンを押すと図 10 に示す通りに，未入力日数が何日以上を対象者とするかを確認できるようにしてある。

図 9 上にある 検温未達者への通知 ボタンをクリックすることで，本シート上にリストアップされた対象者へメール通知するよ

うにしてある。通知文章は，GAS のスクリプト上にある定型文に，ある程度通知内容を差し込めるようにしてある。

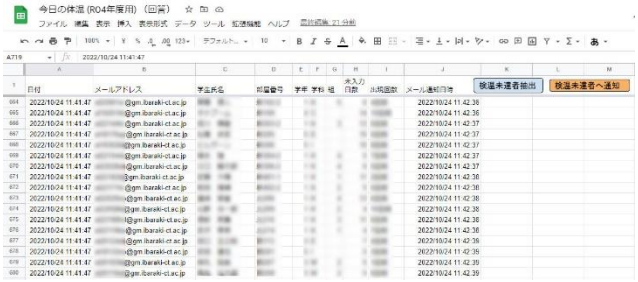


**図 9 検温未報告者へのフォロー**

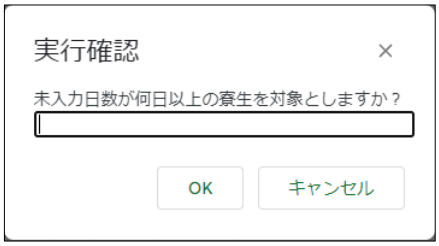



**図 10 未入力日数が何日以上**

Google スプレッドシート上へのボタン配置は，「挿入」－「図形描画」とメニュー操作し，図 11 に示す様に，［図形］アイコンをクリックして「角丸四角形」である▢を選択する。

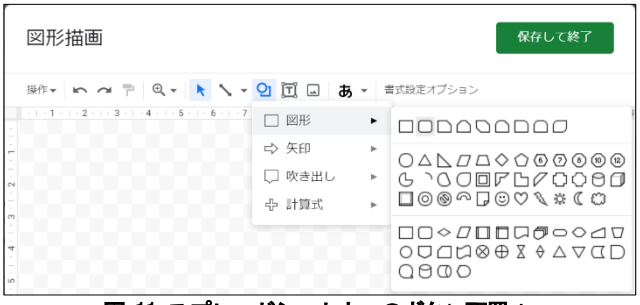



**図 11 スプレッドシート上へのボタン配置１**

その後，図 12 に示す様に，ドラッグでボタンを作成し，そのボタンにテキストを挿入する。テキストの挿入は，作成したボタン上で，ダブルクリックすると行える。最後に「保存して終了」ボタンをクリックして，スプレッドシート上にボタンが描かれれば終了である。

ボタンの位置は，ドラッグ操作で，変更できる。


**図 12 スプレッドシート上へのボタン配置２**

検温未報告者収集のための自作関数として「saveUninputedName」を用意した。この関数とシート上に配置した「検温未達者抽出」ボタンとを連携させるために，「スクリプトを割り当て」を行う。

割り当てたいボタンの上で，「右クリック」し，現れた「⋮」を左クリックして現れる「スクリプトの割り当て」を選択すると，図 13 が現れる。ここで，中央の入力欄に割り当てたい自作関数名(saveUninputedName)を入力して「確定」ボタンをクリックする。この操作をしておくことで，ボタンをクリックするだけで，自作関数が実行される。


**図 13 スクリプトを割り当て**

```
1  function saveUninputedName() {
2    const spreadsheet = SpreadsheetApp.getActiveSpreadsheet();
3    const sheet = spreadsheet.getSheets();
4    var NoDays = Browser.inputBox("実行確認", "未入力日数が何日以上の寮生を対象としますか？", Browser.Buttons.OK_CANCEL);
5    if (NoDays == "cancel" || NoDays > 15) return;
6    // 検温未達者のシート上の登録済み数を得る
7    const colAValues = sheet[4].getRange("A:A").getValues();
8    var nRow = colAValues.filter(String).length;
9    nRow++;//次の行から
10   // 現在の日時を得る
11   const date = new Date();
12   const strDate = Utilities.formatDate(date, 'Asia/Tokyo', 'yyyy/MM/dd HH:mm:ss');
13   // 新・北・西・虹M・紫・虹F を順に調べる
14   for(var i=5; i <= 10; i++) {
15     // シート上に記載されている最終行を得る
16     var colBValues = sheet[i].getRange("B:B").getValues();
17     var lr = colBValues.filter(String).length;
18     //各寮生のメールアドレスから未入力日数までを一括で得る
19     var dataValues = sheet[i].getRange(3, 1, lr-2, 9).getValues();
20     // dataValues から，条件に合った寮生のデータだけを ary 配列に転送する
21     var ary=[];
22     var n=0;
23     for(var j=0; j < lr-2; j++) {
24       if( dataValues[j][6] != 'X' && dataValues[j][8] >= NoDays ) {// メール送信許可者で，指定した日数以上であるか
25         ary[n] = [];
26         ary[n].push(strDate);//日付の登録
27         for(var k=0; k < 6; k++)//氏名から組まで 6 列分登録
28           ary[n].push(dataValues[j][k]);
29         ary[n].push(dataValues[j][8]);//未入力日数の登録
30         n++;
31       }
32     }
33     // 抽出した ary 配列を一括で登録
34     if( n > 0 ) {
35       sheet[4].getRange(nRow, 1, n, 8).setValues(ary);
36       nRow += n;
37     }
38   }
39 }
```

**図 14 saveUninputedName 自作関数**

図 9 で示した「検温未達者抽出」ボタンがシート上に配置しているため，2,3 行で簡単に GAS 処理上必要な sheet オブジェクトを取得できる。4 行が，図 10 で示したダイアログを処理する部分である。7,16 行の「sheet[]」は，簡単にスプレッドシート上のシートを特定する方法である。[ ]内の数値+1 番目にあるシートが特定される。19 行の「getValues()」はその前の getRange()で指定したセル範囲を一括で 2 次元配列として取得する。セル上の値の処理は，getValues を用いて一括に取得した方が，GAS の実行時間が短縮できる。35 行

の「setValues()」も実行時間短縮に効果的である。GAS の主な制限に，「スクリプトの実行時間は 1 回当たり 6 分まで」がある。この制限にかからないように，処理を軽くする。スプレッドシートのセル処理は特に時間がかかり，可能であれば，一括取得，一括登録が最も効果的である。

上記の「saveUninputedName」関数を利用することで，指定した未入力日数以上の寮生がシート上にリストアップされる。これらの者に対するフォローとしてメール配信をする。このための処理が，「検温未達者への通知」ボタンである。本ボタンは，図 13 と同様の方法でスクリプトを「sendEmailTemperature」自作関数に割り当てている。本自作関数の概要を図 15 に示す。

```
1  function sendEmailTemperature() {
2    // 今日の体温のスプレッドシート
3    const spreadsheet = SpreadsheetApp.getActiveSpreadsheet();
4    const sheet = spreadsheet.getActiveSheet();
5    // シート上に記載されている最終行を得る
6    const colAValues = sheet.getRange("A:A").getValues();
7    var nRow = colAValues.filter(String).length;
8    // すべての日付データを得る
9    const sDates = sheet.getRange(2,1,nRow-1,1).getDisplayValues
     ();
10   const sLastDate = sDates[nRow-2][0];
11   var nStartRow = -1;
12   for(var r=nRow-3; r >=0; r--){
13     if( sLastDate === sDates[r][0] ) {
14       nStartRow = r;
15     } else {
16       break;
17     }
18   }
19   if( nStartRow == -1) {
20     Browser.msgBox('登録データが異常です!!');
21     return;
22   }
23   // 該当の未達者情報を得る
24   const n = nRow - nStartRow -1;  // データの行数を得る
25   nonReportDatas = sheet.getRange(nStartRow+2, 2, n, 8).getDis
     playValues();
26   //
27   // 検温報告未達者にその状況をメールで送信する
28   const subject = "【重要】検温報告を行って下さい";
29   var message = '';
30   var sSendDates = [];
31   for(var i=0; i < n; i++){
32     var name = nonReportDatas[i][1];  // 氏名を得る
33     var room = nonReportDatas[i][2];  // 部屋番号を得る
34     var nCount = nonReportDatas[i][7];  // 通知回数
35     message = room + ' ' + name + "さん¥n¥n茨城高専の吉成です
     。¥n";
36     message += "（ここに，検温報告を促すためのメッセージ）";
37     var mail = nonReportDatas[i][0];// メールアドレスを得る
38     // 現在の日時を得る
39     var nDate = new Date();
40     var strDate = Utilities.formatDate(nDate, 'Asia/Tokyo', 'y
     yyy/MM/dd HH:mm:ss');
41     MailApp.sendEmail(mail, subject, message);
42     // 送信日時を残す
43     sSendDates[i] = [];
44     sSendDates[i][0] = strDate;
45   }
46   sheet.getRange(nStartRow+2, 10, n, 1).setValues(sSendDates);
     // メール送信日時の内容を一括で登録する
47 }
48
```

**図 15 sendEmailTemperature 自作関数**

図 14 の説明と重複する内容は省略する。

9 行の「getDisplayValues」は，getValues とは異なり，セルに設定された表示形式に成形された文字を取得する。10～18 行で，最後にシートに追記されたフォローが必要な者の先頭行数を得る。36 行は，概要のみとなっている。ここに，体温未入力の状況を示し，入力を促すメッセージを用意する。41 行は，個々の未入力者ごとにメール配信している。この 1 行のみでメール送信できる。

## 2.7. GAS を用いた寮生側のための Web アプリ

寮生にとって，検温報告を通しての自分の体調確認ができることが大切である。自分の体調確認のために体温の推移が確認できれば，検温報告への協力がしやすい。そのために，未報告も含め過去２週間の体温推移が確認取れるように，図 16 に示す検温報告の状況表示と追加入力ができる Web アプリを用意した。

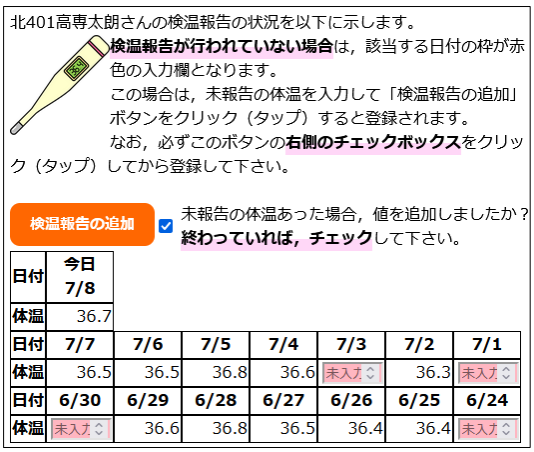



**図 16 検温報告の状況表示と追加入力**

寮生は，スマホ等から指定された URL に接続することで，図 16 に準じた表示が行われる。上記は，あくまでもサンプル表示であり，実際には利用する寮生の部屋番号，氏名，体温が表示される。

未入力部分について，追記できるようにするため，図 17 に示す「追記用」シートを追加した。

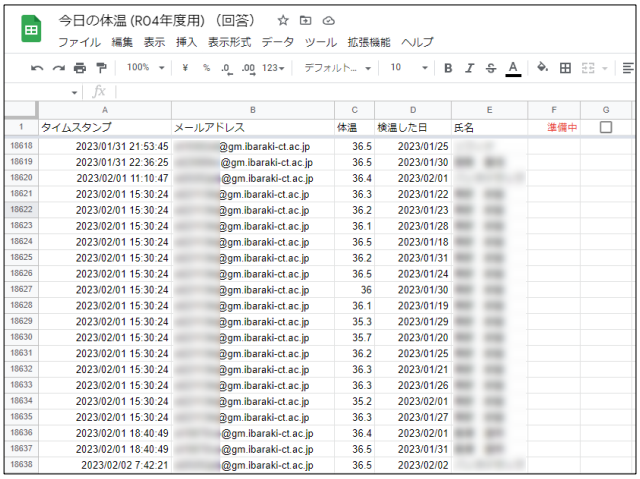



**図 17 追加用シート（検温データ追記用）**

Google フォームと連携させた Google スプレッドシートの初期状態は，図 2 で示した「フォームの回答」シートのみである。フォームの回答シートには，Google フォームで入力された内容が追記されていく。フォームの回答シートは，そのままにして，検温データの未入力分に対応させた追加用シートを準備する。これが，図 3 に示

した集計表にも反映されるように,「追加用」シート用の「MAXIFS」スプレッドシート関数を利用している。

また，図 17 の右上にある「準備中」については，本スプレッドシートのメンテナンス用に用意した。ここに，レ点が入ると「現在準備中です。しばらくお待ち下さい。」の表示が行われ，サービスを停止させられるようにしている。

今回，図 16 を実現するために開発した，「doGet 関数」，「doPost 関数」，「index.html」，「result.html」の概要を次に示す。この４つの関係性については，図 8 と 2.4 節を参照のこと。

```
1  function doGet() {
2   // ログインユーザ ID を得る
3    var objUsers = Session.getActiveUser();
4    var sUID = objUsers.getUserLoginId();
5    // スプレッドシートを ID で取得
6    var sheets = SpreadsheetApp.openById(spreadSheetID);
7    //
8    // 準備中であれば，その表示をして終了する
9    //
10   var bPreparing = sheets.getRangeByName(preparingFlag).getVal
     ue();  // 期間超過であるかを得る
11   if (bPreparing == true) { // 準備中であれば
12     return ContentService.createTextOutput('現在準備中です。し
     ばらくお待ち下さい。');
13   }
14   // 寮生名簿のシートを取得
15   var dormStudentSheet = sheets.getSheetByName(dormStudentList
     );
16   var dormSheetLastRow = dormStudentSheet.getLastRow();
17   // ユーザ ID から該当する寮生の情報を得る
18   var dormList = dormStudentSheet.getRange(2, 1, dormSheetLast
     Row-1, nMaxCol).getValues();
19   var bFind = false;
20   var sFullName = '';
21   var sRoom;
22   var sBldShortName;
23   // dormList は，A 列(配列では 0)がメールアドレスであることが前
     提
24   dormList.forEach(function(IDs){
25     if (IDs[0] == sUID) {
26       sRoom = IDs[nRoomCol -1];  // 部屋番号
27       sFullName = IDs[nFullNameCol - 1];
28       sBldShortName = IDs[nBldShortNameCol - 1];// 棟名の略称
29       bFind = true;
30     }
31   });
32   if( bFind == false) {
33     return ContentService.createTextOutput('あなたの '+sUID+'
     は寮生としての登録にありません。');
34   }
35   // 棟名の略称から，２週間の検温結果が集積されているシート名（
     棟名）を得る
36   var i;
37   var sBldSheetName = '';
38   for(i = 0; i < sBldShortNames.length; i++) {
39     if( sBldShortName == sBldShortNames[i] ) { // 一致する配
     列内の位置が分かれば
40       sBldSheetName = sBldSheetNames[i];
41     }
42   }
43   if( sBldSheetName == '') {
44     return ContentService.createTextOutput(sFullName + 'さんの
     寮生名簿のデータが正しくありません。');
45   }
46   // 検温結果の集計シートを取得
47   var bldSheet = sheets.getSheetByName(sBldSheetName);
48   // 下からアップして最終行を得る
49   var bldSheetLastRow = bldSheet.getRange(bldSheet.getMaxRows(
     ),nStartCol).getNextDataCell(SpreadsheetApp.Direction.UP).getR
     owIndex();
50   // 集計した検温データを得る
51   var dormTempList = bldSheet.getRange(3, 1, bldSheetLastRow-2
     , nStartCol+nMaxDay-1).getValues();
52   bFind = false;
53   var nInputedTemps = [];
54   // dormTempList は，A 列(配列では 0)がメールアドレスであること
     が前提
55   dormTempList.forEach(function(temps){
56     if( sUID == temps[0]) {
57       for(i = 0; i< nMaxDay; i++) {
58         nInputedTemps.push( temps[i + nStartCol - 1] );
59       }
60       bFind = true;
61     }
62   });
63   if( bFind == false) {
64     return ContentService.createTextOutput(sFullName +'さんの
     集計されている検温データがありません。');
65   }
66   // 上記で生成した内容を index.html に反映させる
67   var t = HtmlService.createTemplateFromFile("index");
68   t.email = sUID;
69   t.sroom = sRoom;
70   t.fullName = sFullName;
71   t.today = new Date();
72   t.nInputedTemps = nInputedTemps;
73   var html = t.evaluate().setTitle("検温報告の状況確認");
74   return html;
75 }
```

**図 18 doGet 関数**

3,4 行で，Web アプリに接続してきているユーザーID（Gmail アドレス）を取得する。6 行の「spreadSheetID」には，スプレッドシート ID を別途定義しておく。スプレッドシート ID とは，今回のスプレッドシートを ID 化した文字列である。Google スプレッドシートを開いたときに，「https://docs.google.com/spreadsheets/d/」に続くハッシュ値のような文字列である。これをコピーして，設定しておく。10～13 行で，準備中にできる機能を追加してある。15 行の「dormStudentList」には，寮生名簿用シート名を定義しておく。24～31 行の「forEach」文内では，寮生名簿から，接続してきているユーザーID と一致する，寮生の情報を取得している。この処理をすることで，32～34 行で，高専 Gmail で接続してきている通学生は，サービスを受けられないように制限できる。図 6 で示したアクセスできるユーザーで，茨城高専 Gmail を持つ学生に制限しているが，寮生以外の通学生でも Web アプリ URL を利用すると，接続できてしまうためである。36～45 行で，２週間分の検温データが集計されているシート名を取得している。47～65 行で，接続してきている寮生の２週間分の検温データを nInputedTemps 配列にセットしている。67～74 行で，収集したデータをレスポンスとしてブラウザ側に返すための index.html 生成を行っている。73 行で，このレスポンスを，指定した変数を含ませた形で動的なレスポンスを生成している。

```
1   <body>
2   <?= sroom ?><?= fullName ?>さんの検温報告の状況を以下に示し
    ます。<BR>
3   <span class="marker">検温報告が行われていない場合</span>は，
    該当する日付の枠が赤色の入力欄となります。<BR>
4   この場合は，未報告の体温を入力して「検温報告の追加」ボタンを
    クリック（タップ）すると登録されます。<BR>
5   なお，必ずこのボタンの<span class="marker">右側のチェックボ
    ックス</span>をクリック（タップ）してから登録して下さい。<BR>
6   <form method="POST" action="https://script.google.com/a/macr
    os/gm.ibaraki-ct.ac.jp/s/ (Web アプリ ID) /exec">
7     <input type="hidden" name="email" value="<?= email ?>">
8     <input type="hidden" name="name" value="<?= fullName ?>">
9     <div>
```

```
10        <button id="btn-submit" class="btn-submit" type="submit"
    disabled>検温報告の追加</button>
11        <span>
12        <input type="checkbox" id="fCheck" class="fCheck" requir
    ed>
13        <label for="fCheck" class="lblChkBox">未報告の体温あった
    場合，値を追加しましたか？<BR><span class="marker">終わってい
    れば，レ点を入れて</span>から<BR>ボタンをクリック（タップ）して
    ！</label>
14      </span>
15      </div>
16      <table>
17      <tr><th>日付</th><th>今日<br>
18      <? var strDate = Utilities.formatDate(today, 'JST', 'M/d')
    ; ?>
19      <?= strDate ?>
20      </th><td colspan="6"></td></tr>
21      <tr><th>体温</th>
22      <td>
23      <? var nTemp = nInputedTemps[14]; ?>
24      <? if( nTemp != 0) {?>
25          <?= nTemp; ?>
26        <?} else { ?>
27        <input type="number" name="<?= sTempDate(today); ?>" cla
    ss="noDATA" step="0.1" min="34" max="42" placeholder="未入力">
    <BR>
28        <? } ?>
29      </td><td colspan="6"></td></tr>
30      <tr><th>日付</th>
31      <? var nDate = new Date(today);
32          nDate.setDate(nDate.getDate() -  1);        // 1日減らす
33          var bakDate = new Date(nDate);     // 日付をコピーする
34        for(var i = 13; i > 6; i--){ ?>
35      <th>
36      <?= Utilities.formatDate(nDate, 'JST', 'M/d');
37          nDate.setDate(nDate.getDate() - 1);        // 1日減らす
    ?>
38      </th>
39      <?} ?>
40      </tr>
41      <tr><th>体温</th>
42      <? for(i = 13; i > 6; i--){ ?>
43      <td>
44      <? var nTemp = nInputedTemps[i]; ?>
45      <? if( nTemp != 0) {?>
46          <?= nTemp; ?>
47        <?} else { ?>
48        <input type="number" name="<?= sTempDate(bakDate); ?>" c
    lass="noDATA" step="0.1" min="34" max="42" placeholder="未入力
    "><BR>
49        <?} ?>
50      </td>
51      <? bakDate.setDate(bakDate.getDate() - 1);    // 1日減らす
52        } ?>
53      </tr>
54      <tr><th>日付</th>
55      <? bakDate = new Date(nDate);          // 日付をコピーする
56          for(var i = 6; i >= 0; i--){ ?>
57      <th>
58      <?= Utilities.formatDate(nDate, 'JST', 'M/d');
59          nDate.setDate(nDate.getDate() - 1);        // 1日減らす
    ?>
60      </th>
61      <?} ?>
62      </tr>
63      <tr><th>体温</th>
64      <? for(i = 6; i >= 0; i--){ ?>
65      <td>
66      <? nTemp = nInputedTemps[i]; ?>
67      <? if( nTemp != 0) {?>
68          <?= nTemp; ?>
69        <?} else { ?>
70        <input type="number" name="<?= sTempDate(bakDate); ?>" c
    lass="noDATA" step="0.1" min="34" max="42" placeholder="未入力
    "><BR>
71        <?} ?>
72      </td>
73      <? bakDate.setDate(bakDate.getDate() - 1);    // 1日減らす
74        } ?>
75      </tr>
76      </table>
77    </form>
78    </body>
```

**図 19 index.html**

HTML の form タグの部分のみを示す。HTML タグの説明は省略する。

index.html 内で，特徴的なのが「<? ?>」の部分である。この部分で挟まれた内部は，GAS のスクリプトとして動作する。図 18 の 68～72 行で指定した変数が，この内部で使用できる。「=変数名」のみの場合は，変数の内容が出力される。GAS で使用していた if 文，for 文が見づらくなってはいるが，分割される形で機能している。特に，34～39 行では，for 文により，TH タグの内容を動的に生成している。42～49 行では，TD タグの内容として，登録があれば体温を，登録がなければ入力欄を動的に生成している。

全体としては form タグで機能しているので，6 行での action オプションにより，Web アプリ URL が指定されており，「method="POST"」と指定があるので，「検温報告の追加」ボタンがタップされると，図 8 で示された「doPost 関数」が呼び出される。

```
1  function doPost(e){
2    var sUID = e.parameters.email.toString();   // ログイン ID とな
    る高専 Gmail アドレスを得る
3    var sFullName = e.parameters.name.toString();   // 寮生名を得
    る
4    var params = e.parameters;  // html form からの変数群を得る
5    var nAdditionalTemp = [];   // html form で「stemp 日付」で入力
    された追加体温データ用
6    const regDate = /^stemp¥d{8}/;     // stemp20220320 のようなも
    のにヒットする正規表現
7    var sTempDate;  // 検温した日付
8    var i=0;
9    var regTemp = new RegExp(/^3[4-9](¥.¥d)??$|^4[0-1](¥.¥d)??$|
    ^42$/); // 正しい検温データ
10   for(var key in params) {
11     if( regDate.test(key) ) { // 連想配列の key が stemp2022032
    0 のような形式に一致するか
12       sTempDate = new Date(key.slice(5,9),key.slice(9,11)-1,ke
    y.slice(11));   //key の名前内にある日付文字を日付データに変換，
    月は-1 する
13       if( regTemp.test(params[key]) ) { // 入力欄に正しい検温
    データが入力されていれば
14         nAdditionalTemp[i] = [];
15         nAdditionalTemp[i][0] = sTempDate;
16         nAdditionalTemp[i][1] = params[key]; // 配列に整理して
    日付と体温を格納
17         i++;
18       }
19     }
20   }
21   if( i == 0) {
22     return ContentService.createTextOutput(sFullName + 'さんの
    追加した検温データはありません。');
23   }
24 // form で追加入力された検温結果をシートに追記する
25   // スプレッドシートを ID で取得
26   var sheets = SpreadsheetApp.openById(spreadSheetID);
27   // 追加用のシートを取得
28   var tempAppendSheet = sheets.getSheetByName(tempAppendList);
29   var date = new Date();  // 現時刻を得て
30   var sDate = Utilities.formatDate(date, 'JST', 'yyyy/MM/dd HH
    :mm:ss');
31   var sAppendDate;
32   var nTemp;
33   var protection;
```

```
34    for(i=0; i<nAdditionalTemp.length; i++){
35      sAppendDate = Utilities.formatDate(nAdditionalTemp[i][0],
    'JST', 'yyyy/MM/dd'); // 追加した日付を得る
36      nTemp = String(nAdditionalTemp[i][1]); // その日付の検温結
    果,なぜかStringで処理しないとスプレッドシートに数値が入らない
37      // 一時的に編集権限を付与する
38      protection = tempAppendSheet.protect();
39      protection.addEditor(sUID);
40      // 追加用のスプレッドシートに追記する
41      tempAppendSheet.appendRow([sDate, sUID, nTemp, sAppendDate
    , sFullName]);
42      // 編集権限を削除する
43      protection.removeEditor(sUID);
44    }
45    // 結果を返す
46    var resultpage = HtmlService.createTemplateFromFile("result"
    );
47    resultpage.name = sFullName;
48    resultpage.stemps = nAdditionalTemp;
49    return resultpage.evaluate().setTitle("検温追加登録結果");
50  }
```

**図 20 doPost 関数**

2～4 行で，doPost 関数に渡された引数「e」から必要な変数を受け取っている。図 19 の 27,48,70 行でしていた input タグでは，name オプションで「stemp20220320」のように，その input タグによる入力欄に日付に由来する名前を付与している。これを使う形で，5～23 行で，input タグで入力された検温データを日付情報が付与された状態で「nAdditionalTemp」配列変数にセットしている。34～44 行で，この配列変数の内容をスプレッドシートの「追加用」シートに追記している。38,39,43 行では，アクセス権のない本サービス利用の寮生アカウントに，書き込み権限を与えて，41 行で追記している。46～49 行では，登録結果をブラウザにレスポンスとして返すために，result.html を生成している。

```
1   <body>
2   <?= name ?>さんが追加した検温報告は，以下の通りです。<BR>
3   なお，登録できる体温は34℃～42℃の数値のみです。
4   <?
5     // stemps配列の日付の列で降順ソート
6     stemps.sort(function(a,b){
7       if( (new Date(a[0])) < (new Date(b[0])) ) {
8         return 1;
9       } else {
10        return -1;
11      }
12    });
13  ?>
14  <table border="0">
15    <tr>
16      <th>日付</th><th>体温(℃)</th>
17    </tr>
18    <? stemps.forEach(function(addTemp){ ?>
19    <tr>
20      <td><?= Utilities.formatDate(addTemp[0], 'JST', 'yyyy/MM
    /dd') ?></td>
21      <td><?= addTemp[1] ?></td>
22    </tr>
23    <? }); ?>
24  </table>
25  </body>
```

**図 21 result.html**

今回も，body タグの部分のみとする。

図 20 の 48 行で指定された，追加分の検温データを，4～13 行で，並べ替えをしている。「<? ?>」の説明は，図 19 の下の説明を確認下さい。18～23 行で，forEach 文を利用して動的に追加分の検温データを td タグの中にセットして，table タグを機能させている。

これらの内容が，図 22 に示すように，result.html としてブラウザに表示される。

高専太朗さんが追加した検温報告は，以下の通りです。
なお，登録できる体温は34℃～42℃の数値のみです。

| 日付 | 体温(℃) |
|---|---|
| 2023/02/11 | 36.5 |
| 2023/02/10 | 36.4 |

**図 22 result.html としての表示例**

## 2.8. 検温報告に GAS を機能追加したことによる導入効果

2.7 節で説明した「GAS を用いた寮生側のための Web アプリ」は，2022 年 4 月 1 日から，本格運用を開始した。

令和 4 年度前期開寮日である 4 月 4 日前後の検温入力数の推移を図 23 に示す。図中の「フォーム件数」とは，図 1 で示した検温報告用の Google フォームを利用した入力件数である。「追加件数」とは，本 Web アプリで追加入力された検温報告件数である。

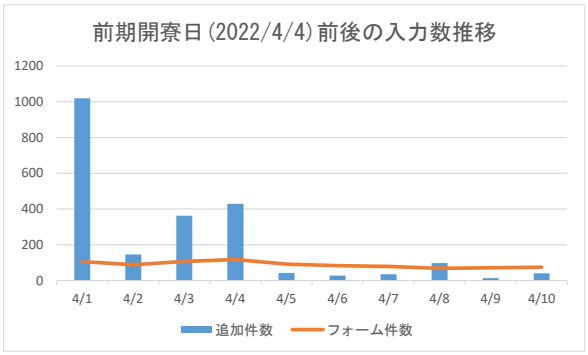



**図 23 前期開寮日(2022/4/4)前後の検温入力数の推移**

開寮日には，2 週間分の体温が入力済みであることを入寮の条件とした。このため，本 Web アプリによる追加入力では，初日に 1,000 件を超える報告となった。一方で，フォーム件数は，概ね 100 件前後で開始したが，開寮後の 4 月 5 日以降は，さらに減少し，70 件程度まで徐々に減少している。この方法による開寮日での入寮予定数 152 名に対して，半数以下の入力数である。本 Web アプリの本来の目的は，Google フォームによる検温報告を補助することである。しかし，図 16 で示したように，後日まとめて入力できるなど，より入力しやすい本 Web アプリを用いた方法を基本とする寮生が多くなっていると考えられる。

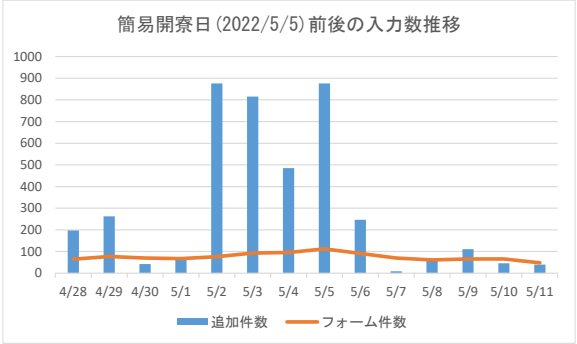



**図 24 簡易開寮日(2022/5/5)前後の検温入力数の推移**

5 月の大型連休に伴う連続 5 日以上の休校期間に実施する簡易閉寮に伴う入力件数の推移を図 24 に示す。簡易閉寮後の開寮日は

５月５日であった。前期開寮日と同様に，２週間の検温報告を入寮の条件としたため，「追加件数」による駆け込み入力の実態が分かる。一方で，図 23 と同様に，開寮日後はフォームによる件数が 70 件前後に減少し，多くの寮生が本 Web アプリを使って，検温報告を実施していることが分かる。本 Web アプリにより，検温報告の状況確認とまとめ入力がが定着していることが分かる。

また，５月２日時点で検温報告が未達の寮生 111 名に対して，2.6 節で説明した学校管理者側のための自作関数を用いてメール通知を行った。２週間の検温報告を入寮の条件とすることも，追記しているため，これ以降連日多くの入力件数となっている。ここから，本学校管理者側のための自作関数も効果を発揮していることが分かる。

開寮日とは異なる通常期の例を図 25 により説明する。検温報告未達寮生を対象に，本自作関数を利用して，週明けの毎週月曜日午前中などに，未達の状況と本 Web アプリによる入力を促している。

図 25 の「追加件数」で特徴的なピークは，この未達寮生へメール通知日と一致しており，学校代表者が本自作関数により入力を促さないと，入力に協力しない状況が見て取れる。しかし，本自作関数により，一定数の検温報告が維持できているとも考えられ，導入効果はあったと言える。

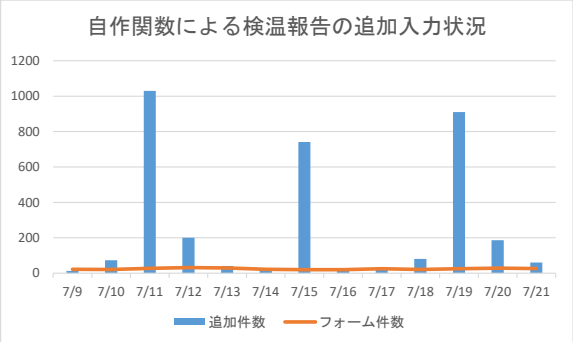



**図 25 自作関数による検温報告の追加入力状況**

## 3. 部屋替え状況表示

2 章で説明した通り，寮生の検温報告の効率化のために GAS による Web アプリを開発した。この経験を，1.3 節で説明した通りに，部屋替えの直列方式に応用した。ただし，検温報告では日々使用するが，今回は部屋替えを行う当日のみ意味のある Web アプリとなる。このため，Google フォームによる入力ではなく，最初から，この目的のために作成した Web アプリ URL に接続する方式とした。なお，寮生にとっては Google フォームを利用しないが，これと連携した Google スプレッドシートを作成するため，ダミーで図 26 に示すフォームを作成する。

図 26 に示した Google フォームと連携したスプレッドシートを作成することで，図 20 の 39 行に示した一時的な編集権限の付与が実行できるようになった。このスプレッドシートには，利用する寮生への編集権限は与えずに運用しているが，Google フォームと連携したシートにすることで，GAS のスクリプトで一時的な編集権限を付与することができた。


**図 26 部屋替え用 Web アプリ向けダミーGoogle フォーム**

### 3.1. 部屋替え状況表示 Web アプリの構造

本 Web アプリの構造は，2.5 節や 2.7 節で説明した検温報告の状況表示 Web アプリと基本的な構造は同じである。検温報告の状況表示 Web アプリでは，表示させる内容は，日付等が異なるだけで画面への表示項目への変化はない。

一方で，部屋替え状況表示 Web アプリは，個々の寮生で部屋替えプランによって大きく異なる。このため，「部屋替えの状況」シートを追加して，部屋替えプランをデータ構造に置き換えて，展開した。本シート自体は，個々の寮生の個人情報が満載のため提示できない。基本的な考え方として，次の通りである。

(1) 部屋替えのプランが直列的に流れていくので，先頭の部屋替えに対応した固有の番号を系列番号として作成する
(2) この直列の部屋替えの流れに該当する部屋替えには，同じ系列番号を用いる
(3) 系列番号以外に，部屋替えの順番を示す時間帯番号を 1 から順に付与する
(4) 直列の部屋替えの流れが，長すぎて現実的な時間以内に終了しない場合は，一端空きスペースへの移動を含ませる
(5) (4)に該当する場合は，該当の寮生にもう一つの系列番号を生成する。
(6) (5)の新たな系列で，続きの部屋替えプランが必要であれば，(5)で生成した系列番号と時間帯番号を用いて，プランを表す

上記の系列番号と時間帯番号を用いて，部屋替えプランの表示を実現した。なお，Web アプリ側のコードの提示は省略する。

### 3.2. 学年末部屋替え時の部屋替え状況表示

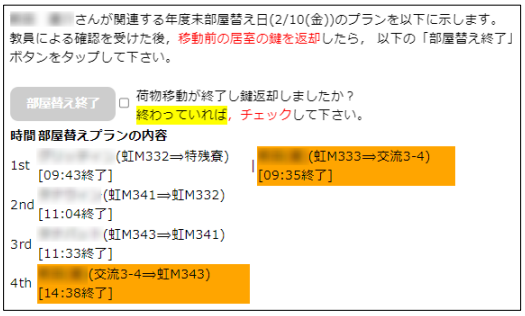



**図 27 ある寮生の部屋替え状況表示**

学年末部屋替のプランは，全寮生向けに Google クラスルームで一覧表の形式で提示している。一覧表の中から，自分はどの部屋か

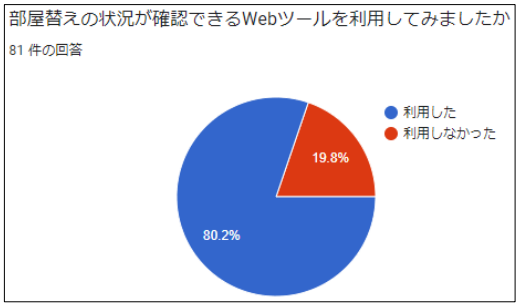
らどの部屋に移動するか確認できる。しかし，直列方式であり，自分の移動先の部屋が空くまでは待機するため，プランの全体的な進行が不明である。これを解消するために，図 27 に示す Web アプリを用意した。ここで使用する Web アプリ URL は，寮生のための Google クラスルームで周知しておく。

図 27 は，部屋替え当日のある寮生の部屋替え状況を示している。オレンジ色で色づけされている部分が，この寮生が関係する部屋替えの部分である。上図では，1st の時間帯で，一端空きスペース（交流 3-4）に荷物を移動し，4th の時間帯になるまで，待機するプランが示されている。なお，上図にはすべて終了時間が入っているが，朝の段階では，部屋替えプランの状況表示のみとなる。部屋替えが終了し，チェックボックスにレ点を入れて「部屋替え終了」ボタンをタップすると，終了した部屋替えに「[〇：〇終了]」と追記される。この終了追記で，どこまで部屋替えのプランが進行しているかが把握できる。特に，上図のプランとなった寮生においては，どこまで待てば良いかがある程度把握できることになる。

### 3. 3. 部屋替え状況表示 Web アプリへのアンケート調査

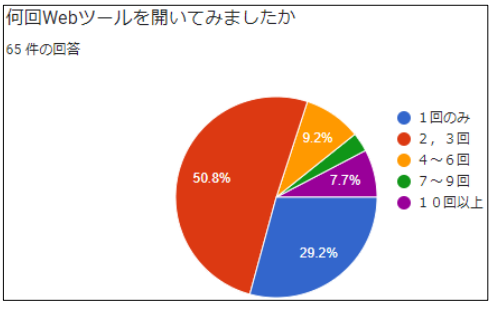
部屋替え対象の 171 名の寮生に対してアンケート調査を行った。回答者は 81 名で，47.4%の回答率であった。部屋替え当日は，別の寮棟への私物移動もあり，寮生にとっては相当の負荷である。学年末部屋替えは，学年末の授業終了日翌日に行うため，部屋替え後に自宅に帰宅する寮生もおり，アンケートへの協力がしにくい状況ではあった。以下，アンケートの調査項目と結果を示す。

**設問１：部屋替えの状況が確認できる Web ツールを利用してみましたか**



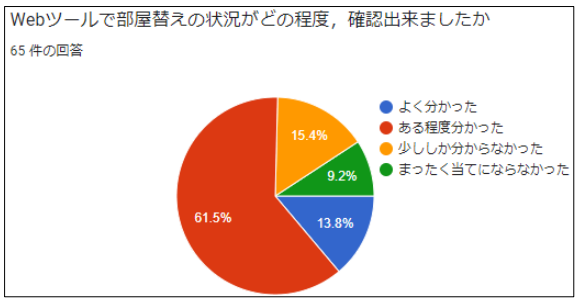

80.2%の寮生が，本 Web アプリを使ってくれたようである。使用した寮生は，次の設問 2 へ移動する。使用しなかった寮生は，設問 5 に移動する。

**設問２：何回 Web ツールを開いてみましたか**



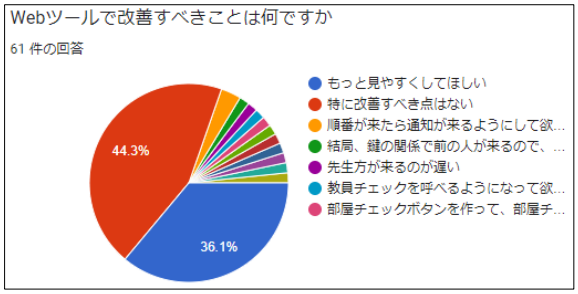

50.8%が 2,3 回の使用で，29.2%が 1 回のみであった。10 回以上使用したのが，7.7%となった。10 回以上使用した寮生は，部屋替えプランの最後の方の可能性がうかがえる。設問 3 に続く。

**設問３：Web ツールで部屋替えの状況がどの程度，確認出来ましたか**




61.5%が「ある程度分かった」と回答しているため，ある程度本 Web アプリは用意した利点を感じてもらえたと考えられる。一方で，9.2%も「まったく当てにはならなかった」と厳しい意見もあり，改善の余地は十分にあると考えられる。設問 4 に続く。

**設問４：Web ツールで改善すべきことは何ですか**



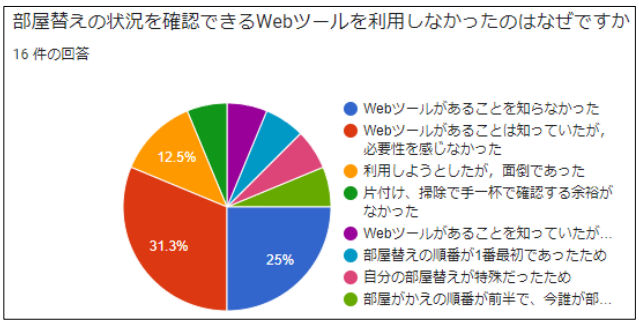

44.3%が「特に改善すべき点はない」であった。36.1%が「もっと見やすくしてほしい」とあった。その他，上図のように，設問 4 で表示する都合上，回答内容が省略されているものがある。これも含め少数意見を以下に示す。

| 回答割合 | 回答内容 |
|---|---|
| 3.3% | 順番が来たら通知が来るようにして欲しい |
| 1.6% | 結局、鍵の関係で前の人が来るので、システムが生かしにくいと思う |
| 1.6% | 先生が来るのが遅い |
| 1.6% | 教員チェックを呼べるようになって欲しい。教員をかなりの人が探し回っている状況であった。 |
| 1.6% | 部屋チェックボタンを作って、部屋チェックができる部屋を担当の教員が把握できるようにし、効率的に部屋チェックが行われるようにして欲しい。 |

上記の少数意見で最後の 3 例は，担当教員への不満である。部屋替え時に担当教員が直接部屋に出向き，清掃が行き届いているかなどの点検を実施する。この確認後に，次に利用する寮生に部屋を明け渡している。限られた教員数で，部屋確認を実施しているので，この不満を本 Web アプリだけで解決するのは相当に難しいが，参考にしていきたい。

GAS を用いた Web アプリは，万能ではないため，スマホのアプリになれた寮生にとっては，表示系が貧弱であったかもしれない。アプリの PR も含めて，改善していきたい。本 Web アプリを使用した寮生へのアンケートは，ここで終了となる。

**設問５：部屋替えの状況を確認できる Web ツールを利用しなかったのはなぜですか**




最も多い意見は，31.3%の「Web ツールがあることは知っていたが，必要性を感じなかった」である。25%が「Web ツールがあることを知らなかった」であった。事前に，寮生のための Google クラスルームで周知していたが，十分であったとは言えない。本アプリの概要などをよく説明するべきであった。

その他，%表示のない少数意見を次に示す。

| 回答割合 | 回答内容 |
|---|---|
| 6.3% | 片付け、掃除で手一杯で確認する余裕がなかった |
| 6.3% | Web ツールがあることを知っていたがバタバタしていて忘れていた |
| 6.3% | 部屋替えの順番が 1 番最初であったため |
| 6.3% | 自分の部屋替えが特殊だったため |
| 6.3% | 部屋がかえの順番が前半で、今誰が部屋替えをしているのか把握していたのと、自分の次に利用する寮生がいなかったため |

## 4. まとめ

GAS(Google Apps Script)を活用して，寮生向けの Web アプリ開発（図 16 図 27 参照）を行った。Web アプリについてアンケート調査も行った（3.3 節参照）。

Web アプリ開発においては，Google スプレッドシートの関数を用いてできる部分は，極力使用した（2.2 節参照）。GAS によるスクリプト記述でしか対応できない部分はこれを使用した（2.7 節参照）。スプレッドシート関数と GAS の連係で，処理を極力軽くし，1 回当り 6 分の動作時間制限に対応させた。今回の GAS スクリプト実行時間は，Google クラウド側の処理状況に左右されるため，数秒から数十秒まで差があるが，処理が軽いことは重要である。

2020 年度以降，日本全国の学寮を持つ学校においては， COVID-19 対策は避けて通れない重要な業務であった。2020 年度以降現在まで，茨城高専の学寮運営を携わってきた筆者にとっても，関係する教職員のさまざまなご協力を頂きながら，手探りで運営を行ってきた。今回，本論文で取り上げた内容は，筆者が開発したツールのほんの一部であったが，寮生と学校管理者双方にとって，有益なツールになるように日々改良を行ってきた。

検温報告は，単なる Google フォームによる入力からスタートした。入力に協力できない寮生への対応などを通して，少しでも寮生と学校管理者との間で，ともに有益なツールになるように改善をしてきた。この経験を元に，部屋替え状況表示アプリの開発も可能となり，実際に令和 3 年度と令和 4 年度末の部屋替え時に運用することができた。

ただ，学寮業務の合間を使った開発であったため，まだまだ改良の余地があるのは，致し方ないとも考えられる。しかし，2.8 節で示した検温入力件数の状況や 3.3 節で示したアンケート調査結果を参考に，残された学寮業務の期間中で改良を進めていきたい。さらに，その他の業務に応用できるものがあれば，発展していきたい。

## 参考文献


1) 文部科学省,”『学校における新型コロナウイルス感染症に関する衛生管理マニュアル～「学校の新しい生活様式」～』(2022.4.1 Ver.8),” https://www.mext.go.jp/a_menu/coronavirus/mext_00029.html
2) GAS (Google Apps Script), https://workspace.google.co.jp/intl/ja/products/apps-script/
3) Google セキュリティの概要, https://cloud.google.com/docs/security/overview/whitepaper?hl=ja
4) Google フォーム, https://www.google.com/intl/ja_jp/forms/about/
5) Google アカウント, https://www.google.com/intl/ja/account/about/
6) Google クラスルーム, http://classroom.google.com/
7) Google ドキュメントエディタヘルプ,”Google スプレッドシート>関数と数式の使用>QUERY” , https://support.google.com/docs/answer/3093343?hl=ja
8) Microsoft ドキュメント,“Excel VBA リファレンス,” https://learn.microsoft.com/ja-jp/office/vba/api/overview/excel

茨城工業高等専門学校
研究彙報校閲方針

**1)校閲の目的**

投稿原稿が，茨城工業高等専門学校研究彙報に掲載される原稿として，ふさわしいものであるかを判定するための資料を提供する校閲を行う．校閲に伴って見出された疑義や不明な事項について修正をお願いする.ただし,原稿の内容に対する責任は著者が負い,その価値は読者が判断する.

**２）校閲の方法**

**a)評価項目**

校閲に当たり，投稿原稿のその分野における位置づけ，新しい観点の内容を含んでいるか，研究・技術成果の貢献度はあるか等について以下の項目で評価する.

①新規性

内容が公知，既発表または既知のことから容易には導き得るものでないこと．以下の項目に該当する場合は新規性があると評価する.

- 主題，内容，手法に独創性がある.
- 学界，社会に重要な問題を提起している.
- 現象の解明に貢献している.
- 教育・人材の育成に新たな貢献をしている.
- 貴重な技術的検討，経験が提示されている.
- 時宜を得た主題について総合的に整理し,新しい知見と見解を提示している.
- その他

②有用性

内容が学問上，あるいは実用上何らかの意味で価値があること．以下の項目に該当する場合は有用性があると評価する.

- 主題，内容が時宜を得て有用である，もしくは，有用な問題提起を行っている.
- 研究・技術の成果の応用性，有用性，発展性がある.
- 研究・技術の成果は有用な情報を与えている.
- 当該分野での研究・技術の体系化を図り，将来の展望を与えている.
- ・研究・技術の成果は実務に取り入れられる価値を持っている.
- 今後の実験，調査，計画，設計，製造，品質管理等に取り入れる価値がある.
- 問題の提起,試論またはそれに対する意見として有用である.
- 実験，実測のデータで研究，開発，製造等の参考として寄与する.
- 教育企画・人材育成上への取り組みに対する有用な成果を含んでいる.
- その他

③論理性

内容が読者に理解できるように簡潔，明瞭，かつ，平易に記述されていること．以下の項目について評価する.

- 全体の構成が適切である.
- 目的と結果が明確である.
- 既往の研究・技術との関連性は明確である.
- 文章表現は適切である.
- 図・表はわかりやすく作られている.
- 全体的に冗長になっていない.
- 図・表等の数が適切である.
- その他

④信頼度

内容に重大な誤りがなく，また読者から見て信用のおけるものであること．信頼度の評価について，計算等の過程を逐一たどることはしないが，以下の項目について評価する.

- 重要な文献が落ちなく引用され,公平に評価されている.
- 従来からの技術や研究成果との比較や評価がなされ，適正な結論が導かれている.
- 実験や解析，あるいは，計画や設計などの条件が明確に記述されている.
- その他

**b）判定**

a)での各項目の評価を参考にして,水準以上であれば登載「可」とし, 掲載をすべきでない場合は「否」とする．なお，a)での各項の評価のうち １つでも問題がありと評価されても「否」とするものではない．多少の欠点があっても,学問や技術の発展に何らかの意味で，良い効果を及ぼす内容があるものは登載されるように配慮する．「否」とする場合は，下記の項目で該当するものが，校閲報告書に示される．

- 理論または考えのプロセスに客観的・本質的な誤りがある．
- 計算・データ整理に誤りがある．
- 都合のよいデータ・文献のみを利用して議論が進められ，公正でない記述により論文が構成されている．
- 修正を要する根本的な指摘事項をあまりにも多く含んでいる．
- 明らかに既発表とみなされる．
- 他人の研究・技術成果をあたかも本人の成果のごとく記述して論文の基本が構成されている．
- 通説が述べられているだけで新しい知見が全くない．
- 政策的な意図, あるいは宣伝の意図が極めて強い．
- 理論的または実証的な論文, あるいは事実に基づいた報告でなく, 単なる主観が述べられているに過ぎない．
- 私的な興味による色彩が極めて強く, 研究彙報に掲載するには問題が多い．
- 全く独断的記述であり，読者に益するとは考えられない．

c)登載の条件

登載可否の判定は，別に設置する校閲委員会で選任された２名の校閲結果に基づいて行う．校閲委員会では，校閲結果をまとめた校閲報告書を作成して審議する．校閲員２名が「可」であれば，原則としてこの投稿原稿は登載可とする．その際，校閲員からの修正意見があれば，校閲委員会で検討の上，修正依頼を行う．修正意見に対して著者が十分な回答を行ったかどうかは，校閲委員会で判断する．

令和５年４月発行

編集・発行　茨城工業高等専門学校
総務課研究協力･地域連携係

〒312-8508　茨城県ひたちなか市中根 866

TEL　029-271-2952