

```

1  const calendar = document.getElementById('calendar');
2  const tooltip = document.getElementById('tooltip');
3  const btn = document.querySelector("#check");
4  const footer = document.getElementById('scrollingText');
5  const header = document.getElementById('header');
6  const body = document.body;
7  let clickCount = 1;
8
9
10 const now = new Date();
11 var year = now.getFullYear();
12 var month = (now.getMonth() + 1);
13 var day = now.getDate();
14 var temp_name;
15
16 const dutySchedule = {
17   "2024-5-1": "S: 黃榮國 A: 黃煜森 N: 王瑞發 C: 孫景泰 R: 張哲維 T: 羅應順",
18   "2024-5-2": "S: 黃經洲 A: 唐 茂 N: 陳建中 C: 張日曜 R: 林厚運 T: 方振彬",
19   "2024-5-3": "S: 詹文欽 A: 王金誠 N: 許敦智 C: 秦桔萬 R: 劉錦郎 T: 呂明峰",
20   "2024-5-4": "S: 林森發 A: 許世勳 N: 彭偉慎 C: 柯正和 R: 黃煜森 T: 方振彬",
21   "2024-5-5": "S: 范振宇 A: 洪柜峰 N: 王瑞發 C: 邱冠霖 R: 余金原 T: 羅應順",
22   "2024-5-6": "S: 柯正和 A: 黃煜森 N: 唐 茂 C: 孫景泰 R: 劉錦郎 T: 林宏儒",
23   "2024-5-7": "S: 黃經洲 A: 王金誠 N: 許敦智 C: 張日曜 R: 余金原 T: 呂明峰",
24   "2024-5-8": "S: 林森發 A: 許世勳 N: 劉暉丞 C: 秦桔萬 R: 張哲維 T: 洪柜峰",
25   "2024-5-9": "S: 黃榮國 A: 劉錦郎 N: 彭偉慎 C: 邱冠霖 R: 林厚運 T: 林宏儒",
26   "2024-5-10": "S: 黃經洲 A: 王金誠 N: 范振宇 C: 張日曜 R: 余金原 T: 呂明峰",
27
28 };
29
30 const holiday = {
31
32   "2024-5-1": "【廿三】",
33   "2024-5-2": "【廿四】",
34   "2024-5-3": "【廿五】",
35   "2024-5-4": "【廿六】【放假日】",
36   "2024-5-5": "【立夏】【放假日】",
37   "2024-5-6": "【廿八】",
38   "2024-5-7": "【廿九】",
39   "2024-5-8": "【四月小】",
40   "2024-5-9": "【初二】",
41   "2024-5-10": "【初三】",
42
43 };
44
45 function Zellercongruence(day, month, year)
46 {
47     if (month == 1)
48     {
49         month = 13;
50         year--;
51     }
52     if (month == 2)
53     {
54         month = 14;
55         year--;
56     }
57     let q = day;
58     let m = month;
59     let k = year % 100;
60     let j = parseInt(year / 100, 10);
61     let h = q + parseInt(13 * (m + 1) / 5, 10) + k + parseInt(k / 4, 10) +
62     parseInt(j / 4, 10) + 5 * j;
63     h = h % 7;
64     switch (h)
65     {
66         case 0:
67             return 6;
68             break;
69         case 1:
70             return 0;
71             break;
72         case 2:
73             return 1;

```

```

73         break;
74     case 3:
75         return 2;
76         break;
77     case 4:
78         return 3;
79         break;
80     case 5:
81         return 4;
82         break;
83     case 6:
84         return 5;
85         break;
86     }
87 }
88
89 function addEventListeners(dayElement, btn, day, month, year, date) {
90     dayElement.addEventListener('mouseover', () => {
91         let todays = document.querySelectorAll('.today');
92         todays.forEach(today => {
93             today.style.backgroundColor = '';
94         });
95
96         if (btn.checked) {
97             dayElement.style.color = 'purple'; // Change the color when button is checked
98             // and mouse is over
99         }
100         showTooltip(date);
101     });
102
103     dayElement.addEventListener('mouseout', () => {
104         if (btn.checked) {
105             dayElement.style.color = 'white'; // Reset the color when button is checked and
106             // mouse is out
107             let today = document.querySelectorAll('.today');
108             today.forEach(today => {
109                 today.style.color = 'MediumBlue';
110             });
111             if (Zellercongruence(day, month, year) === 0 || Zellercongruence(day, month,
112                 year) === 6) {
113                 dayElement.style.color = 'yellow';
114             }
115         }
116         hideTooltip();
117     });
118 }
119
120 function addEventListener_toHideToolTipandShowToday(headerCell) {
121     headerCell.addEventListener('click', () => {
122         const formattedDate = `${year}-${month}-${day}`
123         if (clickCount % 2 === 0) {
124             hideTooltip();
125             let todays = document.querySelectorAll('.today');
126             todays.forEach(today => {
127                 today.style.backgroundColor = '';
128             });
129             // Perform actions for hiding tooltip
130         } else {
131             let todays = document.querySelectorAll('.today');
132             todays.forEach(today => {
133                 today.style.backgroundColor = '#ffff99';
134             });
135             showTooltip(formattedDate);
136             if (btn.checked) {
137                 hideTooltip();
138             }
139         }
140         clickCount++;
141     });
142 }
143
144 // Function to create calendar days

```

```

143 function createCalendar(year, month) {
144     const daysInMonth = new Date(year, month, 0).getDate();
145     const weekdays = ['一', '二', '三', '四', '五', '六', '日'];
146     var counter = 0;
147     var counterN = 0;
148     var now = new Date();
149     var year_now = now.getFullYear();
150     var month_now = (now.getMonth() + 1); // Months are zero-based
151     var day_now = now.getDate();
152     var yearN;
153     var nextMonth = (now.getMonth() + 1) + 1;
154     if (nextMonth > 12) {
155         nextMonth = 1;
156         yearN = year + 1;
157     }
158     yearN=year;
159     const daysInMonthN = new Date(yearN, nextMonth, 0).getDate();
160     dayOfWeekN = Zellercongruence(1, nextMonth, yearN);
161     if (dayOfWeekN === 1) {
162         counterN=0;
163     } else if (dayOfWeekN === 2) {
164         counterN=1;
165     } else if (dayOfWeekN === 3) {
166         counterN=2;
167     } else if (dayOfWeekN === 4) {
168         counterN=3;
169     } else if (dayOfWeekN === 5) {
170         counterN=4;
171     } else if (dayOfWeekN === 6) {
172         counterN=5;
173     } else {
174         counterN=6;
175     }
176     const headerCell = document.createElement('div');
177     headerCell.classList.add('header-cell');
178     headerCell.textContent = ` ${year} 年IDSP${month} 月 `;
179     header.appendChild(headerCell);
180     addEventListener_toHideToolTipandShowToday(headerCell);
181
182
183     const date = `${year}-${month}-${day}`;
184
185     for (let i=0; i < 7; i++){
186         const weekdayElement = document.createElement('div');
187         weekdayElement.classList.add('weekday');
188         weekdayElement.textContent = weekdays[i];
189         calendar.appendChild(weekdayElement);
190     }
191
192     dayOfWeek = Zellercongruence(1, month, year);
193
194     if (dayOfWeek === 1) {
195         counter=0;
196     } else if (dayOfWeek === 2) {
197         counter=1;
198         for (let i = 0; i < counter; i++) {
199             const dayElement = document.createElement('div');
200             dayElement.classList.add('day');
201             dayElement.textContent = "";
202             calendar.appendChild(dayElement);
203         }
204     } else if (dayOfWeek === 3) {
205         counter=2;
206         for (let i = 0; i < counter; i++) {
207             const dayElement = document.createElement('div');
208             dayElement.classList.add('day');
209             dayElement.textContent = "";
210             calendar.appendChild(dayElement);
211         }
212     } else if (dayOfWeek === 4) {
213         counter=3;
214         for (let i = 0; i < counter; i++) {
215             const dayElement = document.createElement('div');

```

```

216     dayElement.classList.add('day');
217     dayElement.textContent = "";
218     calendar.appendChild(dayElement);
219 }
220 } else if (dayOfWeek === 5) {
221     counter=4;
222     for (let i = 0; i < counter; i++) {
223         const dayElement = document.createElement('div');
224         dayElement.classList.add('day');
225         dayElement.textContent = "";
226         calendar.appendChild(dayElement);
227     }
228 } else if (dayOfWeek === 6) {
229     counter=5;
230     for (let i = 0; i < counter; i++) {
231         const dayElement = document.createElement('div');
232         dayElement.classList.add('day');
233         dayElement.textContent = "";
234         calendar.appendChild(dayElement);
235     }
236 } else {
237     counter=6;
238     for (let i = 0; i < counter; i++) {
239         const dayElement = document.createElement('div');
240         dayElement.classList.add('day');
241         dayElement.textContent = "";
242         calendar.appendChild(dayElement);
243     }
244 }
245
246 for (let day = 1; day <= daysInMonth; day++) {
247     const date = `${year}-${month}-${day}`;
248     const dayElement = document.createElement('div');
249     dayElement.classList.add('day');
250     dayElement.textContent = day;
251     calendar.appendChild(dayElement);
252
253     if (Zellercongruence(day, month, year) === 0 || Zellercongruence(day, month, year)
254         === 6) {
255         dayElement.classList.add('weekend');
256     }
257     if (year === year_now && month === month_now && day === day_now) {
258         dayElement.classList.add('today');
259     }
260     addEventListeners(dayElement, btn, day, month, year, date);
261 }
262
263 //below is created for the next month. If the days of next month exceeds this month
264 for (let i = 0; i < (daysInMonthN + counterN - daysInMonth - counter) ; i++) {
265     const dayElement = document.createElement('div');
266     dayElement.classList.add('day');
267     dayElement.textContent = "";
268     calendar.appendChild(dayElement);
269 }
270 }
271
272 function showTooltip(date) {
273     const calendarRect = calendar.getBoundingClientRect();
274
275     tooltip.style.left = `${calendarRect.left-102}px`;
276     tooltip.style.top = `${calendarRect.top+106}px`;
277
278     tooltip.textContent = dutySchedule[date] || "None";
279     tooltip.style.display = 'block';
280
281     document.title = holiday[date] || "None";
282 }
283
284
285
286 function hideTooltip() {
287     tooltip.style.display = 'none';

```

```

288     document.title = 'On Duty Calendar';
289 }
290
291 function change() {
292
293     var day_now = now.getDate();
294     if (btn.checked) {
295         updateSelection();
296         clearSelectedClass();
297         hideTooltip();
298
299         let todays = document.querySelectorAll('.today');
300         todays.forEach(today => {
301             today.style.backgroundColor = '';
302         });
303
304         let previousToday = document.querySelectorAll('.today');
305         previousToday.forEach(today => {
306             today.classList.remove('today');
307         });
308
309         body.classList.add("dark");
310         body.style.backgroundImage = "url('TIAC.png')";
311
312         let days = document.querySelectorAll('.day');
313         days.forEach(day => {
314             day.style.color = 'white';
315         });
316
317         let weekendDays = document.querySelectorAll('.weekend');
318         weekendDays.forEach(day => {
319             day.style.color = 'yellow';
320         });
321
322         let header = document.getElementById('header');
323         header.style.color = 'white';
324         footer.style.color = 'MediumBlue';
325
326
327
328         month = (now.getMonth() + 1) + 1;
329         if (month > 12) {
330             month = 1;
331             year++;
332         }
333         var daysInNextMonth = new Date(year, month, 0).getDate();
334         days.forEach((day, index) => {
335             day.textContent = "";
336             if (index >= Zellercongruence(1, month, year) - 1 && index < daysInNextMonth +
337                 Zellercongruence(1, month, year) - 1) {
338                 const date = `${year}-${month}-${index - Zellercongruence(1, month, year) +
339                     2}`;
340                 day.textContent = index - Zellercongruence(1, month, year) + 2;
341                 addEventListeners(day, btn, index - Zellercongruence(1, month, year) + 2,
342                     month, year, date);
343             } else {
344                 addEventListeners(day, btn, index - Zellercongruence(1, month, year) + 2,
345                     month, year, 999);
346             }
347         });
348         const headerCell = document.querySelector('.header-cell');
349         headerCell.textContent = ` ${year} 年 10 ${month} 月 `;
350         highlightAdditionalHoliday();
351     } else {
352         updateSelection();
353         clearSelectedClass();
354
355         body.classList.remove("dark");
356         body.style.backgroundImage = "url('tower.png')";
357
358         // Reset the color of '.day' elements to their default
359         let days = document.querySelectorAll('.day');
360         days.forEach(day => {

```

```

357     day.style.color = ''; // This will remove the inline 'color' style, allowing
358     the CSS rule to take effect
359 });
360 // Reset the color of 'header' element to its default
361 let header = document.getElementById('header');
362 header.style.color = ''; // This will remove the inline 'color' style, allowing
363 the CSS rule to take effect
364 footer.style.color = 'black';
365
366 month = (now.getMonth() + 1);
367 if (month === 12) {
368     year--;
369 }
370 var daysInCurrentMonth = new Date(year, month, 0).getDate();
371 days.forEach((day, index) => {
372     day.textContent = "";
373     if (index >= Zellercongruence(1, month, year) - 1 && index < daysInCurrentMonth
374         + Zellercongruence(1, month, year) - 1) {
375         const date = `${year}-${month}-${index - Zellercongruence(1, month, year) +
376             2}`;
377         day.textContent = index - Zellercongruence(1, month, year) + 2;
378         addEventListeners(day, btn, index - Zellercongruence(1, month, year) + 2,
379             month, year, date);
380     } else {
381         addEventListeners(day, btn, index - Zellercongruence(1, month, year) + 2,
382             month, year, 999);
383     }
384     if (index - Zellercongruence(1, month, year) + 2 === day_now) {
385         day.classList.add('today');
386     }
387 });
388
389 const headerCell = document.querySelector('.header-cell');
390 headerCell.textContent = ` ${year} 年IDSE${month} 月 `;
391 fetchWeather();
392
393 }
394 //below is to highlight the name previously selected in the change function
395 highlightSelectedName(temp_name);
396 const items = document.querySelectorAll('.picker-item');
397 items.forEach((item) => {
398     if (temp_name === item.textContent && temp_name !== ". . ."){
399         item.style.transform = 'scale(1.5)';
400         item.style.backgroundColor = "turquoise";
401     }
402 });
403 highlightAdditionalHoliday();
404 } //change function ends here
405
406 function scroll(info) {
407     var result = '';
408     // var isSpaceBeforeUppercase = false;
409
410     for (var i = 0; i < info.length; i++) {
411         var currentChar = info[i];
412         var nextChar = info[i + 1];
413
414         if (currentChar === ' ' && (nextChar.match(/[A-Z]/) )) {
415             result += '&nbsp;&nbsp;&nbsp;';
416             // isSpaceBeforeUppercase = true;
417         } else if (currentChar === ' '){
418             result += '&nbsp;';
419         } else {
420             result += currentChar;
421             // isSpaceBeforeUppercase = false;
422         }
423     }
424     var marquee = document.getElementById("scrollingText");
425     marquee.innerHTML = '<p>' + result + '</p>';
426 }
427
428 const names = [
429     ". . .",
430     ". . .",
431     ". . .",

```

```

424     ". . .",
425     "詹文欽",
426     "黃榮國",
427     "范振宇",
428     "唐  茂",
429     "許敦智",
430     "王金誠",
431     "王瑞發",
432     "彭偉慎",
433     "陳建中",
434     "劉暉丞",
435     "柯正和",
436     "張日曜",
437     "孫景泰",
438     "秦桔萬",
439     "邱冠霖",
440     "林森發",
441     "黃煜森",
442     "劉錦郎",
443     "余金原",
444     "林厚運",
445     "張哲維",
446     "黃經洲",
447     "洪柜峰",
448     "林宏儒",
449     "呂明峰",
450     "周育稔",
451     "許世勳",
452     "羅應順",
453     "方振彬",
454     ". . .",
455     ". . .",
456 ];
457 const namePicker = document.getElementById("namePicker");
458 let currentIndex = 0;
459
460 const selectedClassName = 'selected';
461 const selectedClassName2 = 'selected2';
462
463 function highlightSelectedName(selectedName) {
464     const days = document.querySelectorAll('.day');
465     days.forEach(dayElement => {
466         const date = `${year}-${month}-${dayElement.textContent}`;
467         const scheduleForDay = dutySchedule[date] || '';
468
469         // Split the schedule for the day into parts based on spaces
470         const scheduleParts = scheduleForDay.split(' ');
471
472         // Initialize two arrays to store names for each category
473         const namesForSelectedClassName = [];
474         const namesForSelectedClassName2 = [];
475
476         // Initialize a flag to indicate whether to start categorizing into
477         // selectedClassName2
478         let shouldCategorizeToClassName2 = false;
479
480         // Loop through each part of the schedule
481         scheduleParts.forEach(part => {
482             // Check if the part starts with "S:" or "A:"
483             if (part.startsWith('S:') || part.startsWith('A:')) {
484                 // Set the flag to true to start categorizing into selectedClassName2
485                 shouldCategorizeToClassName2 = true;
486             } else if (shouldCategorizeToClassName2) {
487                 // If shouldCategorizeToClassName2 is true, categorize the part to
488                 // selectedClassName2
489                 const name = part.trim();
490                 namesForSelectedClassName2.push(name);
491                 shouldCategorizeToClassName2 = false;
492             } else {
493                 // If shouldCategorizeToClassName2 is false, categorize the part to
494                 // selectedClassName
495                 const name = part.trim();
496                 namesForSelectedClassName.push(name);

```

```

494     }
495   });
496
497   // Check if selectedName is in either category and apply appropriate class
498   if (namesForSelectedClassName2.includes(selectedName)) {
499     dayElement.classList.add(selectedClassName2);
500   } else {
501     dayElement.classList.remove(selectedClassName2);
502   }
503
504   if (namesForSelectedClassName.includes(selectedName)) {
505     dayElement.classList.add(selectedClassName);
506   } else {
507     dayElement.classList.remove(selectedClassName);
508   }
509
510   });
511 }
512
513
514 function highlightAdditionalHoliday() {
515   const days = document.querySelectorAll('.day');
516   days.forEach(dayElement => {
517     const date = `${year}-${month}-${dayElement.textContent}`;
518     const namesForHoliday = (holiday[date] || '').split(' ');
519     // console.log('namesForHoliday:', namesForHoliday);
520     if (namesForHoliday.some(name => name.includes('放假日'))) {
521       if (btn.checked) {
522         dayElement.style.color = 'yellow';
523       } else {
524         dayElement.style.color = 'red';
525       }
526     }
527     if (namesForHoliday.some(name => !name.includes('放假日'))) {
528       if (btn.checked) {
529         dayElement.style.color = 'white';
530       } else {
531         dayElement.style.color = 'black';
532       }
533     }
534   });
535 }
536
537 // function AddLunar() {
538 //   const days = document.querySelectorAll('.day');
539 //   days.forEach(dayElement => {
540 //     const date = `${year}-${month}-${dayElement.textContent}`;
541 //     const lunarName = (holiday[date] || '').split(' ');
542 //     // console.log('namesForHoliday:', namesForHoliday);
543 //     dayElement.textContent += lunarName;
544 //   });
545 // }
546
547 // Populate the name picker with the list of names
548 names.forEach((name) => {
549   const selectedName = name;
550   const item = document.createElement("div");
551   item.className = "picker-item";
552   item.textContent = name;
553   item.style.color = "gray";
554
555   // Add click event listener to handle name selection
556   item.addEventListener("click", () => {
557     updateSelection();
558     const items = document.querySelectorAll('.picker-item');
559     items.forEach((item) => {
560       item.style.transform = 'scale(1)';
561       item.style.color = 'gray';
562     });
563     item.style.transform = 'scale(1.5)';
564     item.style.color = '';
565     item.style.backgroundColor = "turquoise";
566     temp_name=selectedName;

```



```

567     highlightSelectedName(selectedName);
568   });
569   namePicker.appendChild(item);
570 });
571
572 function clearSelectedClass() {
573   const days = document.querySelectorAll('.day');
574   days.forEach(dayElement => {
575     dayElement.classList.remove(selectedClassName);
576   });
577 }
578
579 function updateSelection() {
580   const items = document.querySelectorAll(".picker-item");
581   items.forEach((item) => {
582     item.style.backgroundColor = "";
583   });
584 }
585
586
587 function updateScale() {
588   const container = document.getElementById('namePicker');
589   const items = document.querySelectorAll('.picker-item');
590   const containerRect = container.getBoundingClientRect();
591   const containerCenterY = containerRect.top + containerRect.height / 2;
592
593   items.forEach((item) => {
594     const itemRect = item.getBoundingClientRect();
595     const itemCenterY = itemRect.top + itemRect.height / 2;
596     const distanceToCenter = Math.abs(containerCenterY - itemCenterY);
597     if (distanceToCenter < containerRect.height / 10 && item.textContent !== ". . .")
598       { // Adjust this threshold as needed
599         item.style.transform = 'scale(1.5)';
600         updateSelection();
601         item.style.backgroundColor = "turquoise";
602         item.style.color = '';
603         temp_name=item.textContent;
604         highlightSelectedName(item.textContent);
605       } else {
606         item.style.transform = 'scale(1)';
607         item.style.color = 'gray';
608       }
609   });
610 }
611
612 namePicker.addEventListener("scroll", () => {
613   const itemHeight = namePicker.querySelector(".picker-item").offsetHeight;
614   currentIndex = Math.floor(namePicker.scrollTop / itemHeight);
615   updateSelection();
616   clearSelectedClass();
617   updateScale();
618 });
619
620
621 const apiKey = '35af5c01f0d331eb99f5a42b0259c663';
622 const latitude = 25.07639;
623 const longitude = 121.22389;
624
625 // Clear: Indicates clear sky conditions.
626 // Clouds: Indicates cloudy weather.
627 // Rain: Indicates rainy weather.
628 // Drizzle: Indicates light rain.
629 // Thunderstorm: Indicates thunderstorm activity.
630 // Snow: Indicates snowy weather.
631 // Mist: Indicates misty or foggy conditions.
632 // Smoke: Indicates smoky conditions.
633 // Haze: Indicates hazy conditions.
634 // Dust: Indicates dusty or sandy conditions.
635 // Fog: Indicates foggy conditions.
636 // Sand: Indicates sandstorm conditions.
637 // Ash: Indicates volcanic ash in the air.
638 // Squall: Indicates sudden violent winds.

```

```

639 //Tornado
640 // Function to fetch weather data from the API
641 function fetchWeather() {
642     fetch(
        `https://api.openweathermap.org/data/2.5/weather?lat=${latitude}&lon=${longitude}&ap
        pid=${apiKey}`)
        .then(response => response.json())
        .then(data => {
            let temperatureKelvin = data.main.temp; // Temperature in Kelvin
            let temperatureCelsius = (temperatureKelvin - 273.15).toFixed(1);
            let ch_weather = '';
            let visibility = (data.visibility/1000).toFixed(1);
            let windSpeed = (data.wind.speed*1.943844).toFixed(2); // Wind Speed in
            meters per second
            // let windDirection = data.wind.deg; // Wind Direction in degrees

            humidity = data.main.humidity; // Humidity in percentage
            weatherCondition = data.weather[0].main;

            // Adjust background based on weather condition
            if (weatherCondition === 'Rain') {
                document.body.style.background = 'url(rain.png)';
                document.body.style.backgroundSize= 'cover';
                document.body.style.backgroundPosition= 'center';
                ch_weather='雨天';

            } else if (weatherCondition === 'Clouds' && humidity > 80) {
                document.body.style.background = 'url(clouds.png)';
                document.body.style.backgroundSize= 'cover';
                document.body.style.backgroundPosition= 'center';
                ch_weather='陰天';

            } else if (weatherCondition === 'Drizzle') {
                document.body.style.background = 'url(drizzle.png)';
                document.body.style.backgroundSize= 'cover';
                document.body.style.backgroundPosition= 'center';
                ch_weather='毛毛雨';

            } else if (weatherCondition === 'Thunderstorm') {
                document.body.style.background = 'url(thunderstorm.png)';
                document.body.style.backgroundSize= 'cover';
                document.body.style.backgroundPosition= 'center';
                ch_weather='雷雨';

            } else if (weatherCondition === 'Squall') {
                document.body.style.background = 'url(squall.png)';
                document.body.style.backgroundSize= 'cover';
                document.body.style.backgroundPosition= 'center';
                ch_weather='狂風暴雨';

            } else {
                // Default background for other weather conditions
                document.body.style.background = 'url(tower.png)';
                document.body.style.backgroundSize= 'cover';
                document.body.style.backgroundPosition= 'center';
                ch_weather='晴天';
            }
        })
        if(windSpeed > 63) {
            document.body.style.background = 'url(typhoon.png)';
            document.body.style.backgroundSize= 'cover';
            document.body.style.backgroundPosition= 'center';
            ch_weather='颱風天';

        }

        const date2 = `${year}-${month}-${day}`;
        let info = `${year}年${month}月${day}日` + (holiday[date2] || '') + `桃園機場
        温度:${temperatureCelsius}°C 濕度:${humidity}% 能見度:${visibility}km
        ${ch_weather} ${weatherCondition}` + " " + (dutySchedule[date2] || '');
        scroll(info);
        // Log temperature and humidity

    })
    .catch(error => console.error('Error fetching weather:', error));

```

```
707     }
708
709
710     // Call fetchWeather function initially
711     fetchWeather();
712
713     // Call fetchWeather function periodically (e.g., every 10 minutes)
714     setInterval(fetchWeather, 600000); // 600000 milliseconds = 10 minutes
715
716
717     btn.addEventListener('change', change);
718     createCalendar(year, month);
719     // AddLunar();
720     highlightAdditionalHoliday(); //this must be done finally.
```