Flo/Pomodoro Timer App

Pomodoro

- Break a task into a series of 25 minutes of pomodoro (fully focused on work), and have 5 minutes break among each pomodoro
- However, you can adjust the duration of each pomodoro as long as it works with you

Flomodoro

- Put a timer on your side, the timer will keep adding up the time, indicating the duration of you focused on work
- When you are tired and want to take a break, stop the timer
- The total time you focused on work, divided by 5, will be the duration of your break
- E.g. 25 minutes / 5 = 5 minutes of break

Plans for Plomodoro

- 25 minutes pomodoro, 5 minutes break
- 35 minutes pomodoro, 5 minutes break
- 50 minutes pomodoro, 10 minutes break
- 60 minutes pomodoro, 15 minutes break

Control Buttons

- Reset = reset the pomodoro
- Return to main menu = exit the pomodoro, return to main menu

Plans for Plomodoro

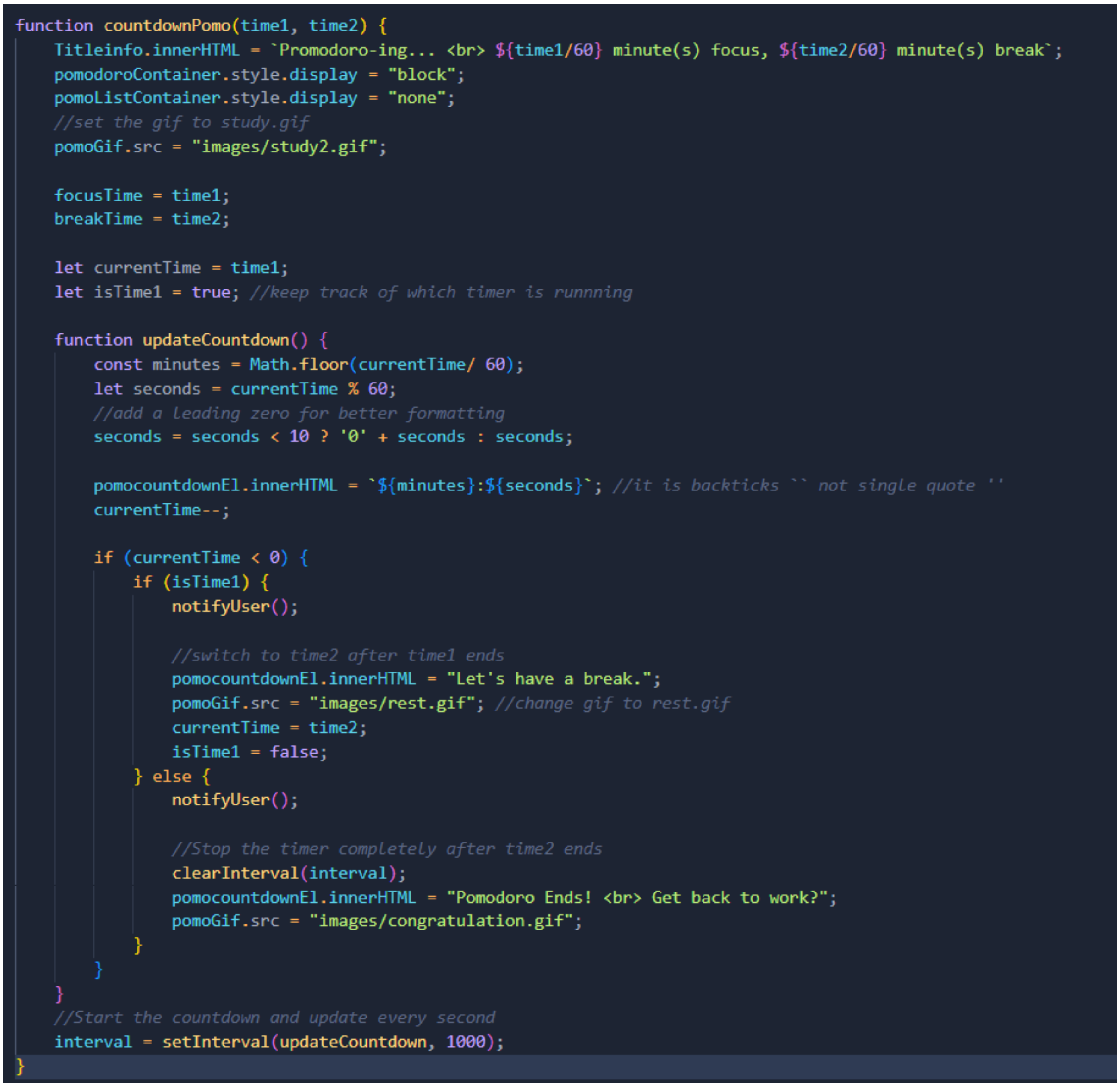
- Put a timer to record total time focused on work
- When the user stops the timer, the app calculates the duration of break available for the flomodoro
- Display the break duration, break timer starts
- Notify user once the break ends

Control Buttons

- Take a break = calculate the break duration and starts break timer
- Reset = Reset the flomodoro

- Return to main menu = exit the flomodoro, back to main menu

Pomodoro Mechanism



```
function countdownPomo(time1, time2) {
    Titleinfo.innerHTML = `Promodoro-ing... <br> ${time1/60} minute(s) focus, ${time2/60} minute(s) break`;
    pomodoroContainer.style.display = "block";
    pomoListContainer.style.display = "none";
    //set the gif to study.gif
    pomoGif.src = "images/study2.gif";

    focusTime = time1;
    breakTime = time2;

    let currentTime = time1;
    let isTime1 = true; //keep track of which timer is runnning

    function updateCountdown() {
        const minutes = Math.floor(currentTime/ 60);
        let seconds = currentTime % 60;
        //add a leading zero for better formatting
        seconds = seconds < 10 ? '0' + seconds : seconds;

        pomocountdownEl.innerHTML = `${minutes}:${seconds}`; //it is backticks `` not single quote ''
        currentTime--;

        if (currentTime < 0) {
            if (isTime1) {
                notifyUser();

                //switch to time2 after time1 ends
                pomocountdownEl.innerHTML = "Let's have a break.";
                pomoGif.src = "images/rest.gif"; //change gif to rest.gif
                currentTime = time2;
                isTime1 = false;
            } else {
                notifyUser();

                //Stop the timer completely after time2 ends
                clearInterval(interval);
                pomocountdownEl.innerHTML = "Pomodoro Ends! <br> Get back to work?";
                pomoGif.src = "images/congratulation.gif";
            }
        }
    }
    //Start the countdown and update every second
    interval = setInterval(updateCountdown, 1000);
}
```

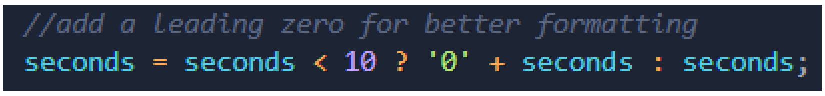
2 values are passed into the ***countdownPomo()*** function, ***time1 = focusTime***, and ***time2 = breakTime***. ***updateCountdown()*** will be the core function that explains the pomodoro mechanism. It calculates and updates how many minutes and seconds are left in the timer. ***currentTime*** will be the timer value, it will be decremented by 1 second on each interval.

Conditions:

1. If ***currentTime*** < 0 and ***isTime1*** is true, it switches to break time ***(time2)*** and updates the UI accordingly.

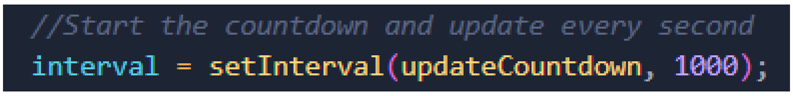
2. Else (after break time ends (***isTime1*** is false) and ***currentTime*** < 0), the timer stops. The interval (timer) is cleared to stop the countdown, and notify users that the Pomodoro session has ended.

The ***notifyUser()*** function is called when the timer ends for both focus and break times to notify the user through sound.



```
//add a leading zero for better formatting
seconds = seconds < 10 ? '0' + seconds : seconds;
```

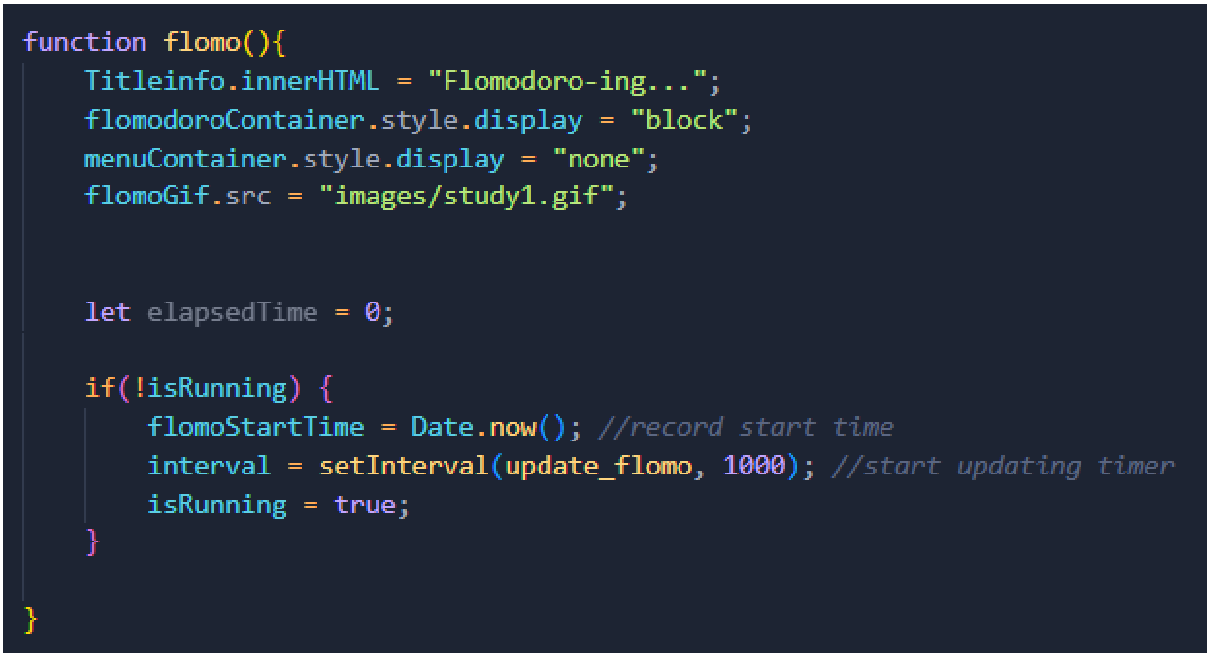
This line of code adds a leading zero for better formatting/padding. This checks if the number of ***seconds*** is less than 10 **(condition)**. If ***seconds*** < 10, it adds a leading zero to the number **(true case)**. Else if the ***seconds*** >= 10, the value remains unchanged **(false case)**.



```
//Start the countdown and update every second
interval = setInterval(updateCountdown, 1000);
```

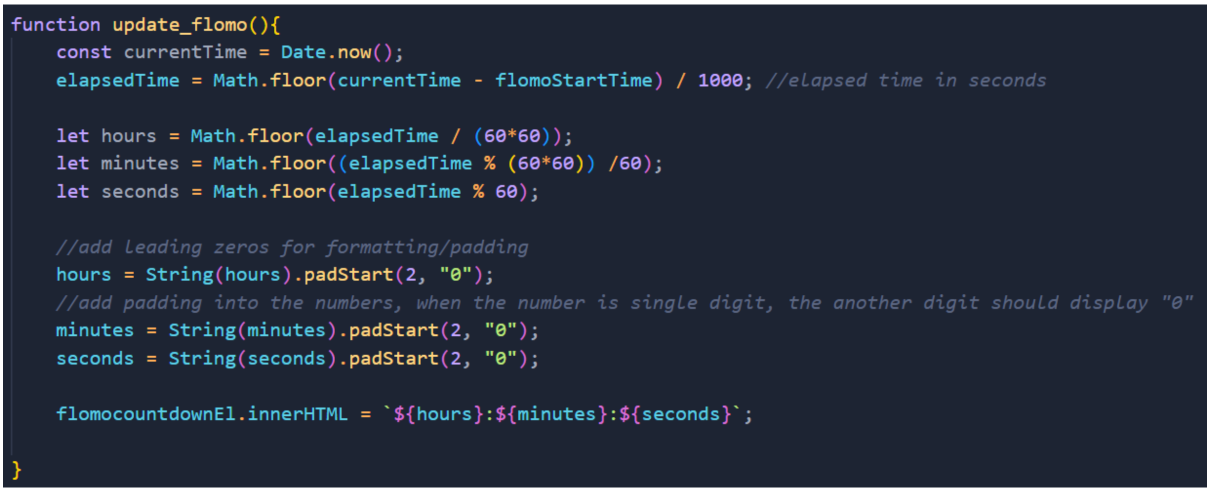
The countdown begins with the ***setInterval()*** function, which runs the ***updateCountdown()*** function every second (1000 milliseconds).

Flomodoro Mechanism



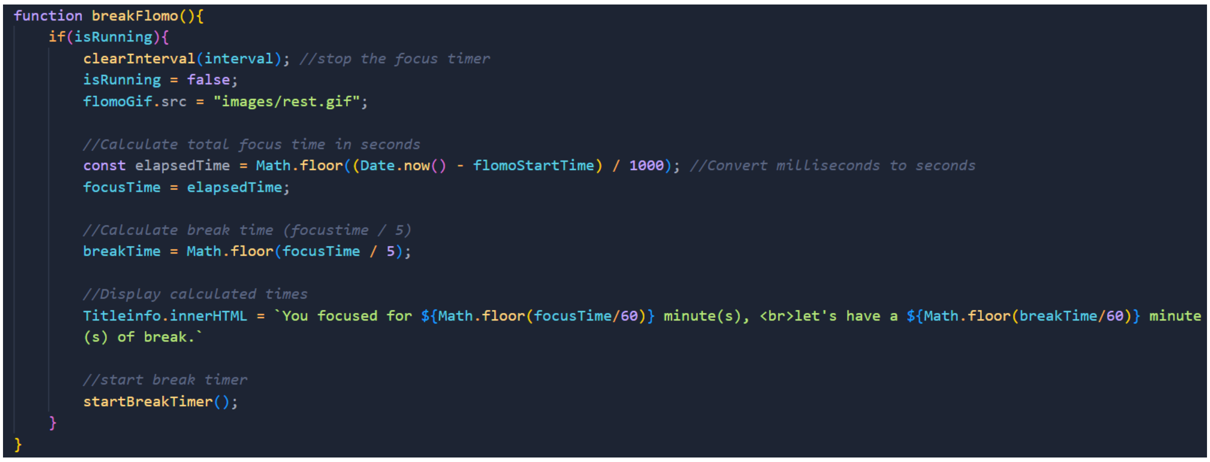
```
function flomo(){
    Titleinfo.innerHTML = "Flomodoro-ing...";
    flomodoroContainer.style.display = "block";
    menuContainer.style.display = "none";
    flomoGif.src = "images/study1.gif";


    let elapsedTime = 0;

    if(!isRunning) {
        flomoStartTime = Date.now(); //record start time
        interval = setInterval(update_flomo, 1000); //start updating timer
        isRunning = true;
    }

}
```

Flomo() function is responsible for starting the Flomodoro. It updates the UI and displays the Flomodoro screen. ***isRunning*** is a global boolean variable that identify whether the timer is running or not. If the timer is not running, then the timer starts. The current time is recorded using ***Date.now()***, which will be used later to calculate the elapsed time. The ***interval*** will be updated every seconds using ***update_flomo()***. ***isRunning*** is set to true to indicate that the timer is now running.



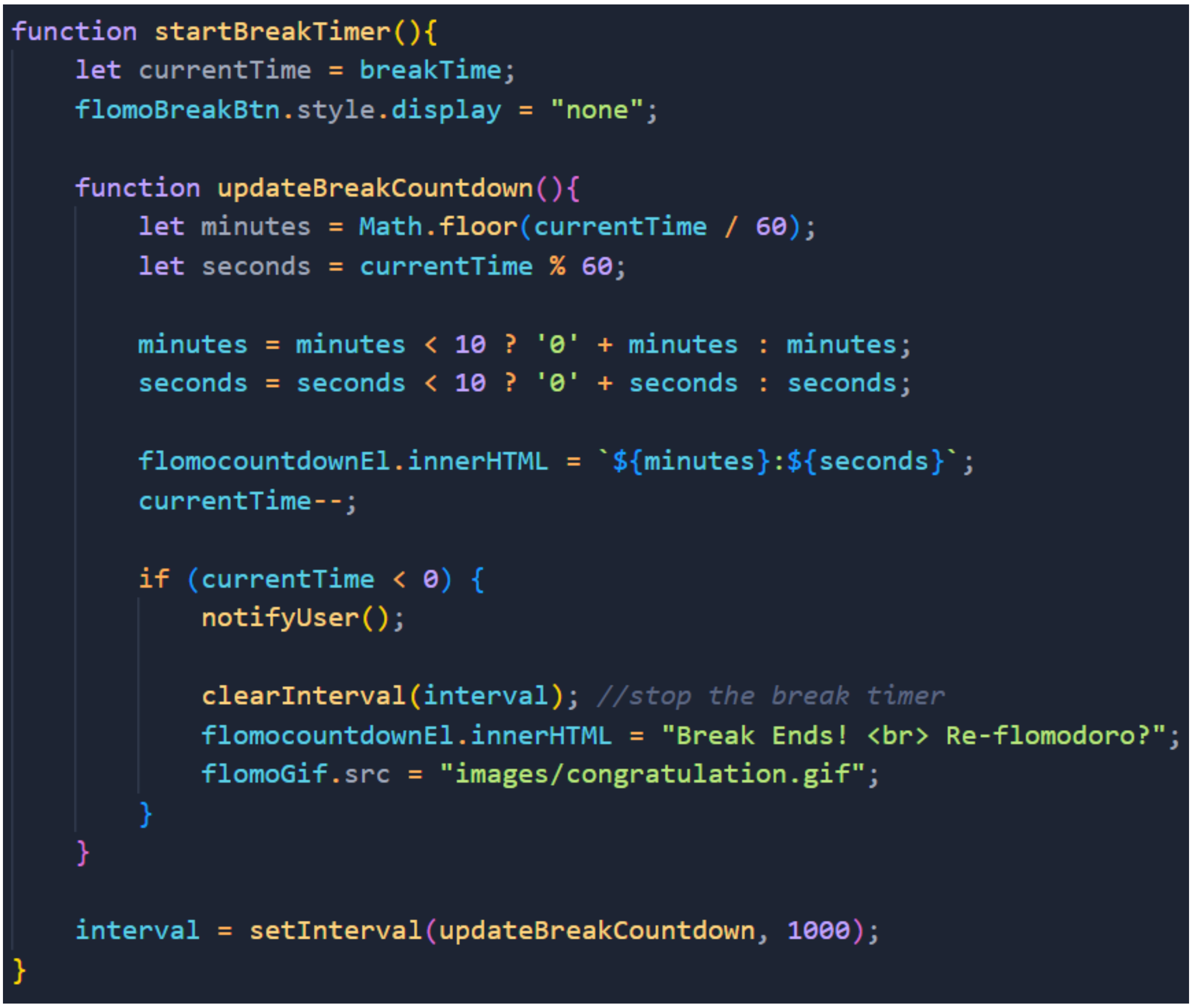
```
function update_flomo(){
    const currentTime = Date.now();
    elapsedTime = Math.floor(currentTime - flomoStartTime) / 1000; //elapsed time in seconds

    let hours = Math.floor(elapsedTime / (60*60));
    let minutes = Math.floor((elapsedTime % (60*60)) /60);
    let seconds = Math.floor(elapsedTime % 60);

    //add leading zeros for formatting/padding
    hours = String(hours).padStart(2, "0");
    //add padding into the numbers, when the number is single digit, the another digit should display "0"
    minutes = String(minutes).padStart(2, "0");
    seconds = String(seconds).padStart(2, "0");

    flomocountdownEl.innerHTML = `${hours}:${minutes}:${seconds}`;

}
```

Update_flomo() function is responsible for updating the Flomodoro focus timer. It tracks the elapsed time since the focus timer started. Elapsed time is the total amount of time that has passed between two events/points in time. It is used to measure the duration of an activity/process. In the code, ***elapsedTime*** refers to the time that has passed since the Flomodoro timer started. Then, ***update_flomo()*** splits the total elapsed time into hours, minutes, and seconds. The time is displayed in *hh:mm:ss* format.



```
function breakFlomo(){
    if(isRunning){
        clearInterval(interval); //stop the focus timer
        isRunning = false;
        flomoGif.src = "images/rest.gif";

        //Calculate total focus time in seconds
        const elapsedTime = Math.floor((Date.now() - flomoStartTime) / 1000); //Convert milliseconds to seconds
        focusTime = elapsedTime;

        //Calculate break time (focustime / 5)
        breakTime = Math.floor(focusTime / 5);

        //Display calculated times
        Titleinfo.innerHTML = `You focused for ${Math.floor(focusTime/60)} minute(s), <br>let's have a ${Math.floor(breakTime/60)} minute
        (s) of break.`

        //start break timer
        startBreakTimer();
    }
}
```

breakFlomo() function is triggered when the user needs to take a break after their focus session. It checks if the timer is running (***isRunning*** is true). If it is running, the function proceeds by ***clearInterval()***, which stops the current focus timer. Then, ***isRunning*** is set to false to indicate that the timer is stopped. The ***elapsedTime*** is calculated and re-declaring the ***focusTime*** to the calculated ***elapsedTime***, which is the total time spent in focus mode. Using the ***focusTime***, it calculates the ***breakTime***, which indicates the break duration available for the focus mode duration. The function ***startBreakTimer()*** is called to begin the break countdown.



```
function startBreakTimer(){
    let currentTime = breakTime;
    flomoBreakBtn.style.display = "none";

    function updateBreakCountdown(){
        let minutes = Math.floor(currentTime / 60);
        let seconds = currentTime % 60;

        minutes = minutes < 10 ? '0' + minutes : minutes;
        seconds = seconds < 10 ? '0' + seconds : seconds;

        flomocountdownEl.innerHTML = `${minutes}:${seconds}`;
        currentTime--;

        if (currentTime < 0) {
            notifyUser();

            clearInterval(interval); //stop the break timer
            flomocountdownEl.innerHTML = "Break Ends! <br> Re-flomodoro?";
            flomoGif.src = "images/congratulation.gif";
        }
    }

    interval = setInterval(updateBreakCountdown, 1000);
}
```

startBreakTimer() controls and updates the break countdown timer. It sets the ***currentTime*** to the calculated ***breakTime***. ***updateBreakCountdown()*** is called every second using ***setInterval()***. It calculates the remaining break time in minutes and seconds, ***currentTime*** is decremented by 1 second in each interval. When ***currentTime*** < 0, the timer stops and notify users about it.