Software Requirements Specification (SRS)

VS-Saturn

Team: 3

Authors: Alex Bochman, Luann Dias, Michael Gonzales, Dominic Gordon, and Derek Lamoreaux

Customer: Software Engineers

Instructor: Professor James Daly

# Introduction

This document is intended to provide detailed information about the proposed Visual Studio Code extension. The extension will supplement software engineers using Visual Studio Code who desire to follow the pomodoro style of time management. This document will start with an overview of the project including the purpose, scope, and a general overview of the extension. Followed by specific and detailed descriptions on the functionality of the extension, along with its restrictions and constraints. We have included diagrams that help in visualizing what the project is intended to do, along with a prototype of the system. References and contacts may be found at the end of the document.

## Purpose

The purpose of this document is to give detail on how the extension was constructed, who it will benefit, and give detail on improvements and faults of the extension. The intended audience of this document is specifically investors looking into monetizing the product, software engineers wanting to develop similar products or understand VS-Saturn itself, and our professor.

## Scope

VS-Saturn is a plugin designed specifically for Visual Studio Code that creates a visual guide for following the pomodoro style of time management inside Visual Studio Code itself. The intent for the extension is to keep the user’s focus on their work and not become distracted by changing their focus on an exterior notebook or note taking device. The user will have access to a list where they can add or remove tasks, along with the ability to pause or skip a timer completely. Keeping the user immersed in their work is of high priority. When signifying switches between work and break time, VS-Saturn inverts the current Visual Studio Code theme. This way the extension is not intrusive with loud pings or visible notifications, distracting the user from their work. This way, the user is encouraged to keep their focus on Visual Studio Code unless they are on a break, maintaining focus on what tasks need to be done. This is all done to achieve the goal of allowing a visible time management software that does not pull the user’s attention away from their current task.

## Definitions, acronyms, and abbreviations

1. Pomodoro Technique: A format of time management where time is separated in two separate blocks that repeat. The first block is for productive work for 25-30 minutes. The second block is a 3-5 minute break. This cycle can continue as long as the user would like. After three breaks, the next break is 30 minutes long, as to reward the user for their hard work.
2. Pomodoro: The time in which the user works on productive tasks. This is where the Pomodoro Technique gets its name.
3. VSCode: Abbreviation for Visual Studio Code. VSCode is a free source code editor where developers can edit, build, and debug code with ease.
4. User: The human using VSCode.
5. Task: An objective that the user would like to complete.
6. Timer: A display that shows how much time the user has until their next break or work time in minutes and seconds.
7. GUI: Graphical User Interface

## Organization

The rest of the document is organized in the following way. Section 2 describes the software itself, including its construction, specifics on functionality, and constraints. Section 3 goes over the requirements for designing and developing the VS-Saturn. Section 4 includes design diagrams and outlines on how the VS-Saturn will work. Section 5 goes over a prototype of VS-Saturn, giving details as to where it can be found and how to operate it. Lastly, section 6 is a list of references and sources we have used during development.

# Overall Description

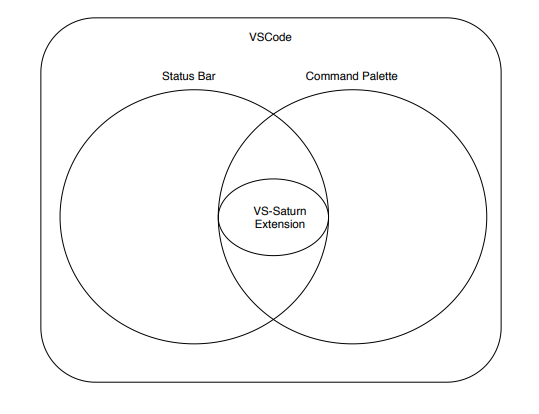
This section will cover the specifics of how VS-Saturn works. It covers constraints, interfaces, functionality, user expectations, assumptions of the software and user, and objectives beyond the scope of this project.

# Product Perspective

VS-Saturn aims to be an aid to programmers who struggle with time management or desire a structured time management technique. There are multiple different time management techniques such as Getting Things Done, the Important-Urgent Matrix, and Do It Now.[[1]](#footnote-1) However, Pomodoro is one of the more well known techniques.1 The philosophy behind the technique is the idea that frequent short breaks keep you more mentally clear and refreshed. It fits very well in today’s day in age as it relies on self control of the user and not checking your phone every ten seconds as we do today. The idea is that you focus for twenty to thirty minutes on productive work, entitled a pomodoro, and then take a three to five minute break. This forces the user to enact self-control, making checking social networks or playing a mindless phone game wait until your break. It also rewards the user for being consistent in sticking to the pattern by making every fourth break a twenty to thirty minute break. It also gives the mind a much larger break after working for two hours.

VS-Saturn implements these features by having a list for the tasks the user wants to get done, along with a visible timer to display the remaining time in the pomodoro. The task list is fully customizable so that the user can add and remove tasks as need be, as well as mark tasks complete when they are done to visually see their progress. The user can interact with the timer by pausing the timer if they need to step away momentarily, or reset it if they forgot to pause as they stepped away and want to reset it.

This is all available inside of VSCode using the status bar and command palette. The timers are manipulated through the status bar, and the task list is manipulated through the command palette. The relationship between VSCode and VS-Saturn can be seen more clearly through the following diagram.



Overall System Diagram: VS-Saturn integration with VSCode

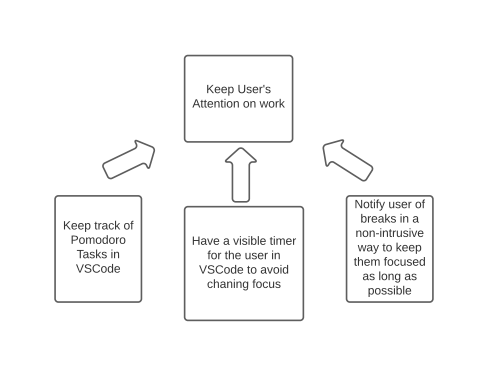
This way, the user can see all their tasks and how much time they have left inside of the VSCode itself and not have to move their attention to a separate notebook, phone, or different application. This helps avoid distractions by keeping work and pomodoro related information all in one application. VSCode is used as it is easy to create extensions in it, and it is compatible with a hundreds of programming languages such as HTML, Java, Python, C#, C++, and many more. [[2]](#footnote-2) It is readily available for hundreds of languages, and extensions are easily accessible through the VSCode marketplace, VS-Saturn is created for VSCode.

# Product Functions

The major functions of VS-Saturn are as follows:

* Capable of keeping track of tasks the user creates
* Clear tasks that have been marked complete
* Count down time for every break and pomodoro
* Notify the user when a break or pomodoro starts

The following is a high-level goal diagram for development of the VS-Saturn extension. The boxes represent individual goals, and the arrows show dependency on other goals.



High-Level Goal Diagram: Shows how our overall goals relate to each other

# User Characteristics

As VS-Saturn is designed for VSCode, it is expected that the user has at least beginner level knowledge of VSCode. They should be able to know how to access the extension marketplace and maneuver throughout VSCode’s different functionalities. As it goes along with using VSCode and software engineering, typing and mouse expertise is expected as well. The ability to tell time on a digital clock is also expected of the user. Knowledge of converting minutes to hours is beneficial to the user, but not necessary. The user is also expected to have a desire to either improve or cement their time management skills. Knowledge of the pomodoro technique is beneficial, but not necessary as there will be a general overview of how the extension works in the extension marketplace.

# Constraints

**Software**

VS-Saturn is specifically built for VSCode. The user must be willing to program in VSCode as VS-Saturn will not be available or usable outside of it. VSCode is only available on Windows, Mac OS, and Debian and Red Hat versions of Linux, the user is expected to be running one of these operating systems. VS-Saturn is only available via the built in extension market, thus the user must have an internet connection to initialize the process. The task list is constrained to one instance of VSCode, as the tasks do not save when you close VSCode. Other than the tasks the user writes themselves, VS-Saturn’s UI elements are limited to english only.

**Hardware**

VSCode is a small download (< 100 MB) and has a disk footprint of 200 MB. VSCode is lightweight and should easily run on today's hardware. Thus the user is expected to have a modern computer that fits these criteria. Microsoft recommends a 1.6 GHz or faster processor and 1 GB of RAM to run VSCode.

# Assumptions and Dependencies

As stated previously, developers and users must have experience using VSCode as a free source code editor and have at least beginner knowledge on its features and extension marketplace. It is also assumed that the hardware on the user’s computer is capable of running VSCode. Thus the user will be running on Windows, Mac OS, or Linux. The computer also needs a 1.6 GHz or faster processor and 1 GB of RAM in order to run VSCode. Along with the computer, the user is also assumed to have access to the internet to use the extension marketplace to download VS-Saturn. Lastly, it is assumed that the user has a desire to keep track of how they are using their time and keep track of what they have accomplished in said time.

# Apportioning of Requirements

VS-Saturn is built to help build time management skills. It does not, however, guarantee that the user will become an instant master of time management.

VS-Saturn is designed to notify the user in a non-intrusive way when pomodoros or breaks are up. This is done via inverting the theme while it is open. However, if the user minimizes VSCode, the user may not notice. Notifying the user in other non-intrusive ways when VSCode is minimized is beyond the current scope of the project, however, it is something that may be addressed in the future in the form of using taskbar notifications such as lighting up green on Windows or bouncing the icon on Mac OS.

Allowing for multiple users to interact with the same list and timers will be very beneficial for group work. However, VS-Saturn is designed to be used by one user on one device. Multi-user functionality is beyond the scope of the current project, however, it is something that may be added in future versions.

# Specific Requirements

**Overall Requirements**

Hardware Requirements:

1. A 1.6 GHz or faster processor
2. At least 1 GB of RAM

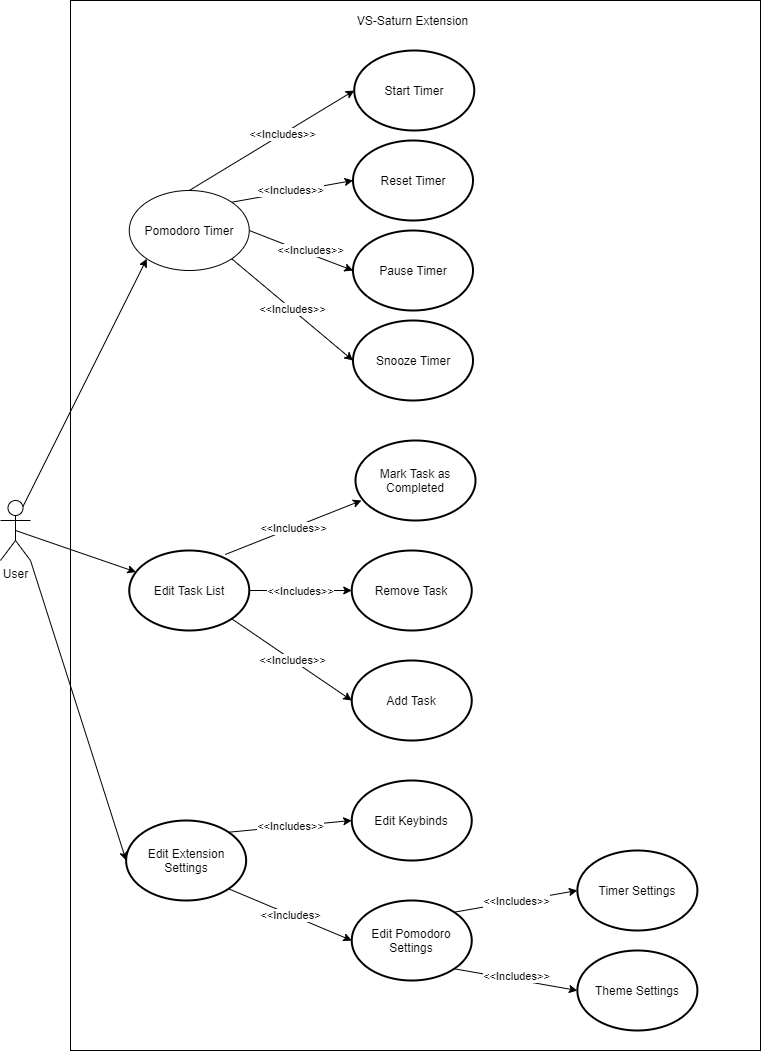
Software Requirements:

1. Task list
   1. Must be able to add tasks, remove tasks, and mark tasks as complete.
2. Clock.
   1. Must display when the user is on a pomodoro or on break.
   2. Display the remaining time of a pomodoro or break.
   3. Notify the user when it has reached the end of a timer.
   4. Must be able to track how many breaks have been taken.
      1. After each third short break is taken, the next break timer will be a long break, which is default 30 minutes.
      2. Any time before the third break ends, the next break timer should be set to 5 minutes be default.
   5. Allow long break and short break time to be changed by the user
   6. Allow the number of short breaks until a long break to be changed by the user
3. GUI
   1. Task Menu
      1. Task Menu buttons
         1. Can manipulate the task list in such was as adding a task, removing a task, and marking a task complete
   2. Clock
      1. Clock Control Buttons
         1. Can manipulate the clock in such ways as, pausing, starting, skipping, and resetting it.
   3. Options
      1. Options List
         1. Can manipulate aspects of the clock such as long break time, short break time, and pomodoro time.
   4. Break Completion Visualization
      1. Must be capable of showing the user how many breaks have been taken and show how many are left before a long break.
      2. Notifications to the user when a break is over or a pomodoro is starting must be non-intrusive
   5. Intuitive Layout
      1. All buttons pertaining to the clock and task list should be near each other for clarity
      2. Must aesthetically pleasing and understandable
      3. Button names should be short, clear, and concise for maximum use of space as the status bar is small and thin.
      4. Prompts included when necessary, such as adding or removing tasks from task list

# Modeling Requirements

**Use Case Diagram:**

Use case diagram to show the relationships between user and the system in a broad fashion.

****

|  |  |
| --- | --- |
| Use Case Name: | Pomodoro Timer |
| Actors: | User (initiator) |
| Description: | Initial step for timer manipulation, houses buttons and timer backend |
| Type: | Primary |
| Includes: | Start Timer, Reset Timer, Pause Timer, Snooze Timer |
| Extends: | N/A |
| Cross-refs: | Requirement 2 and 3b |
| Uses cases: | None |

|  |  |
| --- | --- |
| Use Case Name: | Start Timer |
| Actors: | User |
| Description: | The user starts pomodoro or break timer |
| Type: | Primary |
| Includes: | N/A |
| Extends: | N/A |
| Cross-refs: | Requirement 3b |
| Uses cases: | Pomodoro Timer |

|  |  |
| --- | --- |
| Use Case Name: | Reset Timer |
| Actors: | User (initiator) |
| Description: | User resets pomodoro or break timer |
| Type: | Primary |
| Includes: | N/A |
| Extends: | N/A |
| Cross-refs: | Requirement 3b |
| Uses cases: | Pomodoro Timer |

|  |  |
| --- | --- |
| Use Case Name: | Pause Timer |
| Actors: | User (initiator) |
| Description: | User pauses pomodoro or break timer |
| Type: | Primary |
| Includes: | N/A |
| Extends: | N/A |
| Cross-refs: | Requirement 3b |
| Uses cases: | Pomodoro Timer |

|  |  |
| --- | --- |
| Use Case Name: | Snooze Timer |
| Actors: | User (initiator) |
| Description: | Add five minutes to pomodoro or break timer |
| Type: | Primary |
| Includes: | N/A |
| Extends: | N/A |
| Cross-refs: | Requirement 3b |
| Uses cases: | Pomodoro Timer |

|  |  |
| --- | --- |
| Use Case Name: | Edit Task List |
| Actors: | User (initiator) |
| Description: | Initial step for editing task list. Starts when the user opens the task list |
| Type: | Primary |
| Includes: | Mark Task as Completed, Remove Task, Add Task |
| Extends: | N/A |
| Cross-refs: | Requirements 1a and 3a |
| Uses cases: | None |

|  |  |
| --- | --- |
| Use Case Name: | Mark Task as Completed |
| Actors: | User |
| Description: | User marks tasks complete or not complete |
| Type: | Secondary |
| Includes: | N/A |
| Extends: | N/A |
| Cross-refs: | Requirements 1a and 3a |
| Uses cases: | Edit Task List |

|  |  |
| --- | --- |
| Use Case Name: | Remove Task |
| Actors: | User |
| Description: | User removes task from task list |
| Type: | Secondary |
| Includes: | N/A |
| Extends: | N/A |
| Cross-refs: | Requirements 1a and 3a |
| Uses cases: | Edit Task List |

|  |  |
| --- | --- |
| Use Case Name: | Add Task |
| Actors: | User (initiator) |
| Description: | User adds task to task list |
| Type: | Secondary |
| Includes: | N/A |
| Extends: | N/A |
| Cross-refs: | Requirements 1a and 3a |
| Uses cases: | Edit Task List |

|  |  |
| --- | --- |
| Use Case Name: | Edit Extension Settings |
| Actors: | User (initiator) |
| Description: | Settings Menu that houses Keybind Settings and Pomodoro Settings |
| Type: | Primary |
| Includes: | Edit Keybinds, Edit Pomodoro Settings |
| Extends: | N/A |
| Cross-refs: | Requirement 3c |
| Uses cases: | N/A |

|  |  |
| --- | --- |
| Use Case Name: | Edit Keybinds |
| Actors: | User |
| Description: | Menu for Keybind settings |
| Type: | Secondary |
| Includes: | N/A |
| Extends: | N/A |
| Cross-refs: | Requirement 3c |
| Uses cases: | Edit Extension Settings |

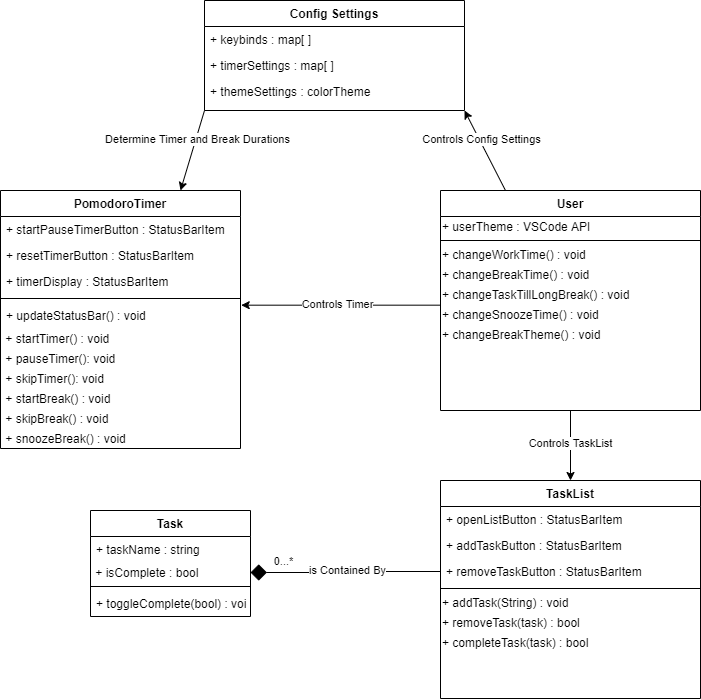
|  |  |
| --- | --- |
| Use Case Name: | Edit Pomodoro Settings |
| Actors: | User |
| Description: | Menu for Pomodoro settings including Timer and Theme |
| Type: | Secondary |
| Includes: | Timer Settings, Theme Settings |
| Extends: | N/A |
| Cross-refs: | Requirements 2e, 2f, and 3c |
| Uses cases: | Edit Extension Settings |

|  |  |
| --- | --- |
| Use Case Name: | Timer Settings |
| Actors: | User |
| Description: | Settings Menu to change break and pomodoro time |
| Type: | Tertiary |
| Includes: | N/A |
| Extends: | N/A |
| Cross-refs: | Requirements 2e, 2f, and 3c |
| Uses cases: | Edit Pomodoro Settings |

|  |  |
| --- | --- |
| Use Case Name: | Theme Settings |
| Actors: | User |
| Description: | Get access to VSCode theme settings |
| Type: | Tertiary |
| Includes: | N/A |
| Extends: | N/A |
| Cross-refs: | Requirements 3c and 3d |
| Uses cases: | Edit Pomodoro Settings |

**Class Diagram:**

A class diagram in order to show the associations between the different classes and the user.

****

|  |  |  |
| --- | --- | --- |
| **Element Name** | | **Description** |
| **Config Settings** | | Menu class for settings of extension such as pomodoro timer and key binds |
| **Attributes** |  |  |
|  | keybinds:map | Array of strings that holds all keybinds for specific commands |
|  | timerSettings:map[] | Array of integers for timer settings such as break time and pomodoro time |
|  | themeSettings : colorTheme | Container for current theme and break theme |
| **Operations** |  |  |
|  | **None** |  |
| **Relationships** | Config Settings has relationships with User and PomodoroTimer. The user can edit the timer and key bind setting inside config settings. The Config Settings determine the length of short and long breaks for PomodoroTimer. | |
| **UML Extensions** | **None** | |

|  |  |  |
| --- | --- | --- |
| **Element Name** | | **Description** |
| **PomodoroTimer** | | Class that contains all the functionality of pomodoro timer. This includes break notification, starting, pausing, and snoozing timers. |
| **Attributes** |  |  |
|  | startPauseTimerButton : StatusBarItem | A button that starts timers, the icon turns to a pause icon and will pause the timer |
|  | resetTimerButton : statusBarItem | A button that resets the current timer to the initial time. |
|  | timerDisplay : StatusBarItem | Container displaying the current amount of time left in the current timer. |
| **Operations** |  |  |
|  | updateStatusBar() : void | Update button icons and time |
|  | startTimer() : void | Start current timer |
|  | pauseTimer() : void | Pause current timer |
|  | skiptimer() : void | Skip current timer and go to the next timer |
|  | startBreak() : void | Start break timer |
|  | skipBreak() : void | Skip break and go to next pomodoro |
|  | snoozeBreak() : void | Extend break by five minutes |
| **Relationships** | The PomodoroTimer has relationships with User and ConfigSettings. The user can manipulate both break and pomodoro timers in PomodoroTimer. ConfigSettings determine the length of short and long breaks. | |
| **UML Extensions** | **None** | |

|  |  |  |
| --- | --- | --- |
| **Element Name** | | **Description** |
| **Task** | | Object class for tasks that the user will create. |
| **Attributes** |  |  |
|  | taskName : string | Name that appears in task list |
|  | isComplete : bool | Bool that signifies whether or not the task is complete |
| **Operations** |  |  |
|  | toggleComplete() : void | Mark task as the opposite of isComplete |
| **Relationships** | The TaskList contains an array of 0 or more tasks | |
| **UML Extensions** | The relationship between Task and TaskList is symbolized by an aggregation. | |

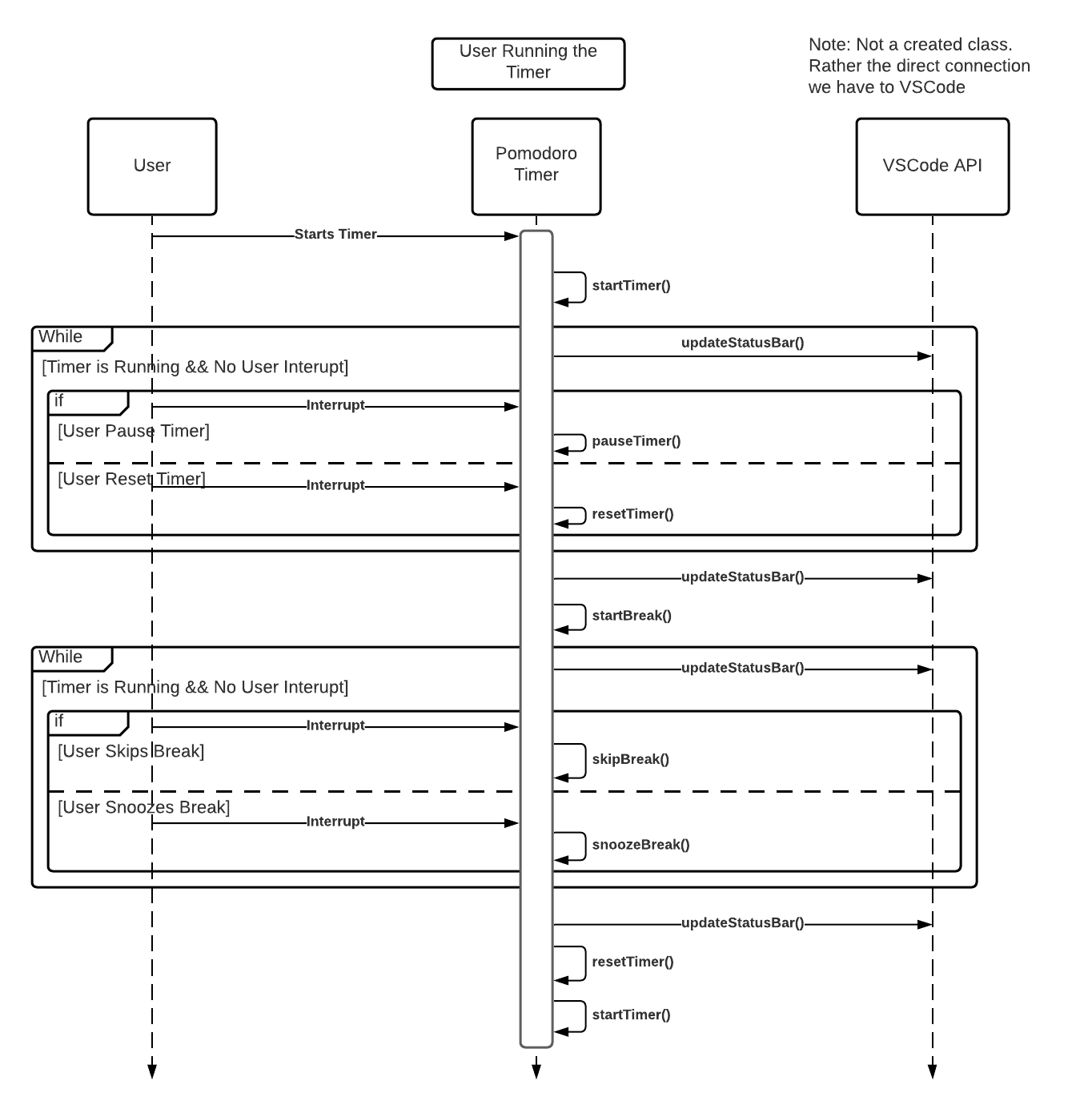
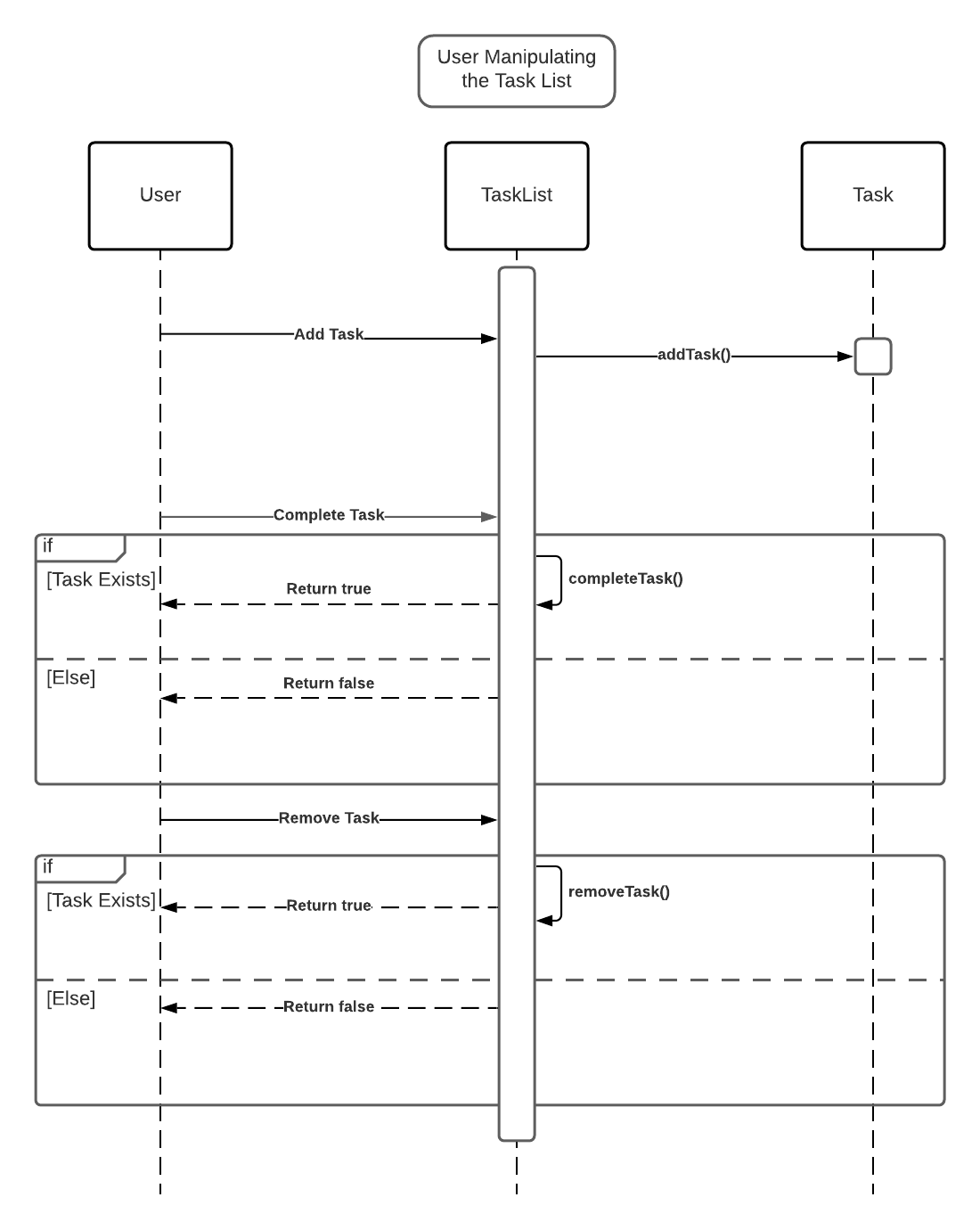
|  |  |  |
| --- | --- | --- |
| **Element Name** | | **Description** |
| **TaskList** | | A container of tasks that is displayed for the user. |
| **Attributes** |  |  |
|  | taskListArray : task[ ] | Array of tasks that are displayed to the user |
|  | openListButton : StatusBarItem | Button to open and close the task list |
|  | addTaskButton : StatusBarItem | Button to add a new task to the task list |
|  | removeTaskButton : StatusBarItem | Button to a remove task from task list |
| **Operations** |  |  |
|  | addTask(task t) : void | Function that adds task t to task list |
|  | removeTask(task t): bool | Function that removes task t from task list, returns true if successful. |
|  | completeTask(task t) : bool | Function that marks task t as complete, returns true if successful. |
| **Relationships** | The TaskList has a relationship with User. The user can add and remove tasks, or mark tasks as complete. This is all done through the task menu. | |
| **UML Extensions** | **None** | |

|  |  |  |
| --- | --- | --- |
| **Element Name** | | **Description** |
| **User** | | Class to represent the user |
| **Attributes** |  |  |
|  | userTheme : VSCode API | The current theme of the user |
| **Operations** |  |  |
|  | changeWorkTime(int t) : void | Change the length of the pomodoro timer to t. |
|  | changeBreakTime(map b, int t) : void | Change the length break b to time t. |
|  | changeTaskTillLongBreak(int t) : void | Change the amount breaks the user must take to achieve a long break to t tasks |
|  | changeSnoozeTime(int t) : void | Change length of break snooze to time t. |
|  | changeBreakTheme(VSCodeTheme t) : void | Change what theme VSCode changes to signify break time to theme t |
| **Relationships** |  | |
|  | The user has relationships with Config Settings, PomodoroTimer, and TaskList. The user can edit the timer and key bind setting inside config settings. The user can change the break and pomodoro timers, along with how many tasks it takes in order to get a long break. The user can also manipulate both break and pomodoro timers in PomodoroTimer. This is done through the start, reset, snooze and skip buttons. Lastly, the user can add and remove tasks, or mark tasks as complete. This is all done through the task menu. | |
| **UML Extensions** | **None** | |

**Sequence Diagram:**

The following sequence diagrams give an example of the user interacting with the task list, and interacting with the pomodoro timer.

**Sequence Diagram 1: TaskList**

When using the task list, if the user adds a task, it appears in the task list. If a user desires to remove a task or mark a task as complete, the system will first find out whether or not the task actually exists before deleting the task.****

**Sequence Diagram 2: Pomodoro Timer**

The pomodoro timer starts with the user starting a pomodoro. Then, while the timer is going, the user can pause or reset the timer at any given moment. While all this is happening, the status bar is being updated with the correct icons and remaining time. Then the break starts, changing the theme and starting a new timer. While this timer is running, the user can skip or snooze the break at any given moment. The status bar continues to update throughout the break as well. The cycle continues as long as the user desires.

# 5. Prototype

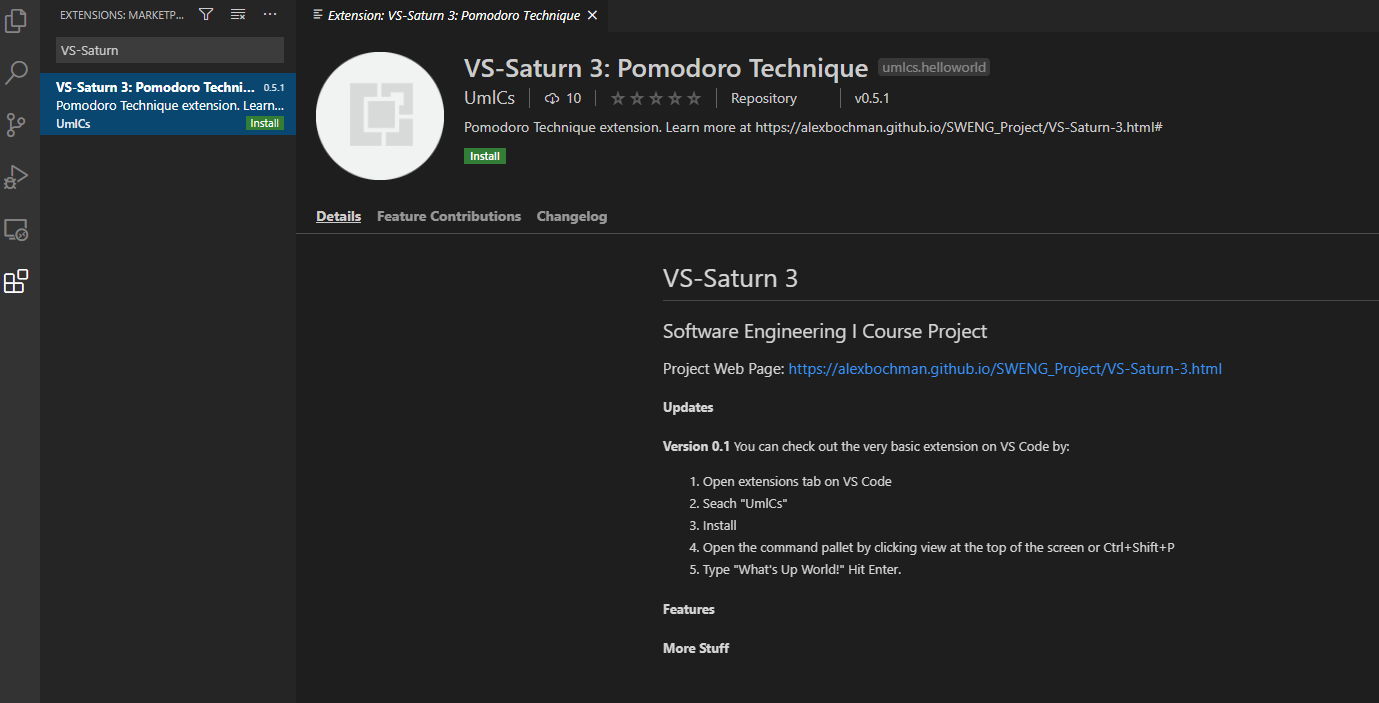
The following prototype was implemented to show what the final product will look like visually. It is essentially an empty shell with no functionality. Its sole purpose is to show what using the VS-Saturn will look like. This includes showing the time and list user interfaces.

# 5.1 How to Run Prototype

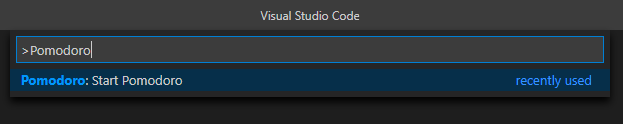
**How to Download and Run VS-Saturn:**

1. Download and Open Visual Studio Code
2. Open the Extension Marketplace
3. Search “VS-Saturn” into the search bar and click on “VS-Saturn 3: Pomodoro Technique”
4. Click “Install”
5. Press “CTRL + SHIFT + P” on Windows or Linux, or “CMD + SHIFT + P” on Mac OS to open the Command Palette.
6. Type in “Pomodoro” into the Command Palette
7. Press “ENTER” or click on “Pomodoro: Start Pomodoro”

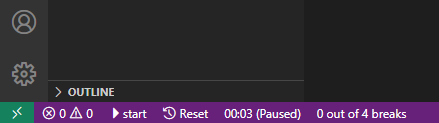
VS-Saturn in Visual Studio Marketplace:



Command Palette View:

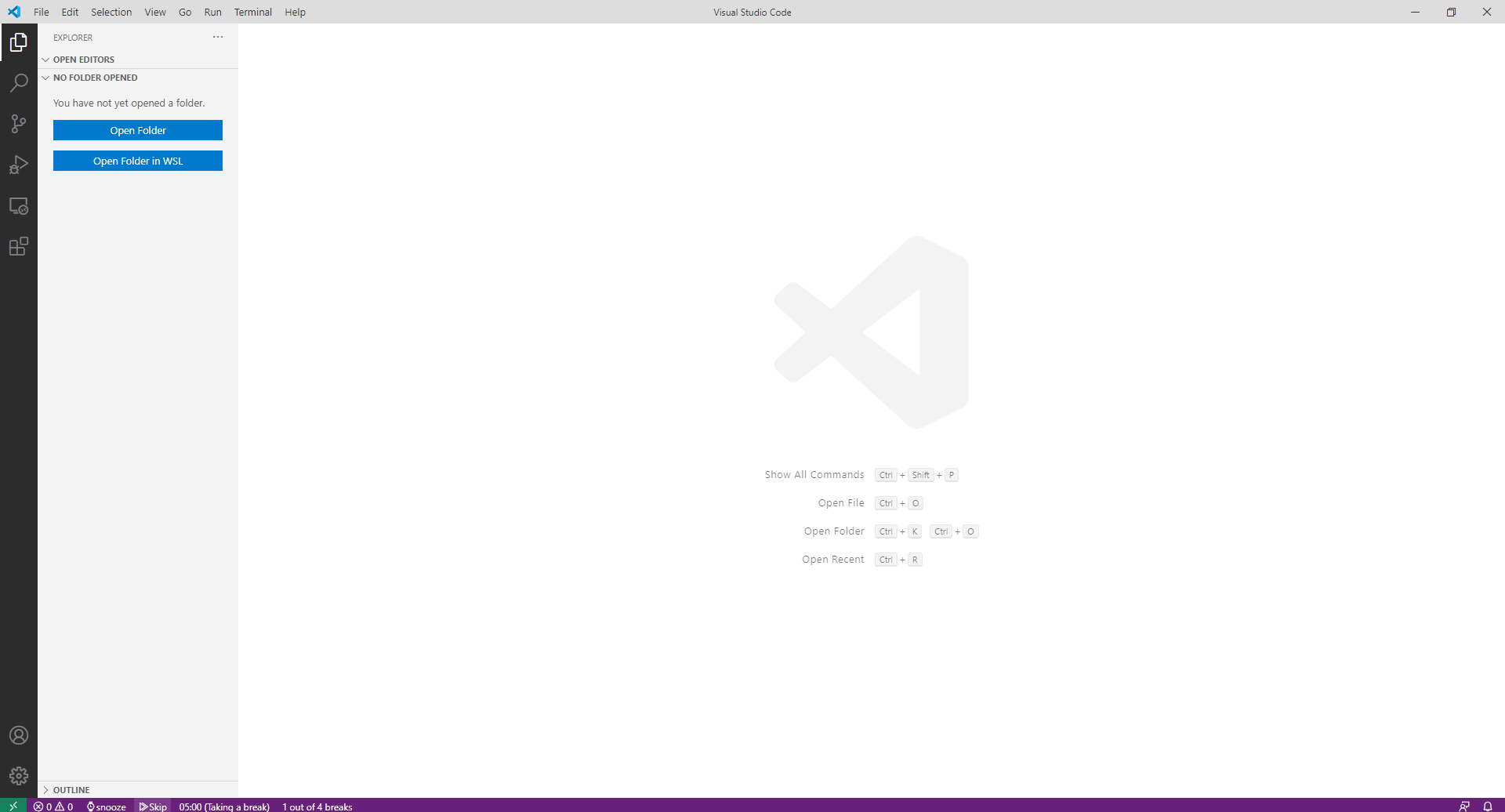


Status Bar View at Start of Pomodoro:

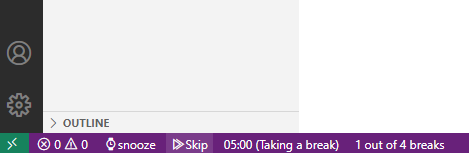


# 5.2 Sample Scenarios

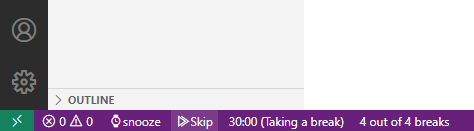
In the download and start guide, the initial startup of VS-Saturn was displayed. The following is what VS-Saturn looks like when the user’s pomodoro timer is up and the break starts. In this example, the user’s default theme is the dark theme, thus the break theme is the light theme.

VSCode’s theme is now the light theme

The status bar updates to show that the user is on a break, marking off one break. The buttons have also switched from Start and Reset to Snooze and kip respectively.



The following is how VSCode looks after the user has taken their set number of breaks to achieve a long break. In this example, the user must take four breaks in order to get a thirty minute break.

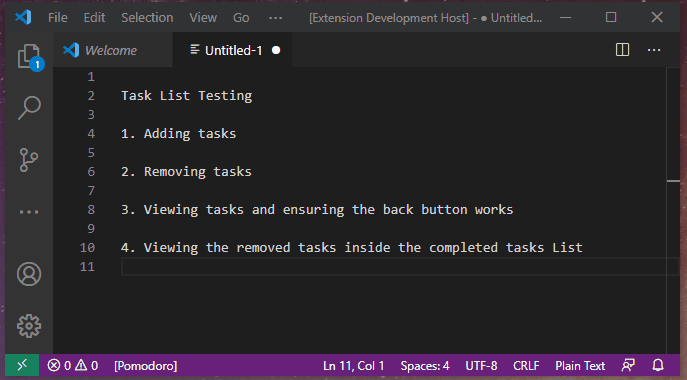


The long, short, and pomodoro timer can all be edited in the options tab. The following example shows a user editing the timer settings.

Text

Description automatically generated

The following is what the task list will look like. The user clicks on Tasks on the status bar and opens the list on the top of VSCode that displays their tasks. The user is able to add, remove, and view completed tasks in the task menu.



# References

[1] Code, V., 2020. Language Support In Visual Studio Code. [online] Code.visualstudio.com. Available at: <https://code.visualstudio.com/docs/languages/overview#:~:text=On%20this%20website%2C%20we%20have,%2D%20T%2DSQL%20%2D%20TypeScript.> [Accessed 18 November 2020].

[2] Tay, D., 2020. 10 Popular Time Management Techniques. [online] Brightpod.com. Available at: <https://www.brightpod.com/boost/10-popular-time-management-techniques> [Accessed 18 November 2020].

Project Website: [VS Saturn 3 Project Page (alexbochman.github.io)](https://alexbochman.github.io/SWENG_Project/VS-Saturn-3.html)

VSCode Marketplace Link: <https://marketplace.visualstudio.com/items?itemName=UmlCs.helloworld#review-details>

# Point of Contact

For further information regarding this document and project, please contact **Prof. Daly** at University of Massachusetts Lowell (james\_daly at uml.edu). All materials in this document have been sanitized for proprietary data. The students and the instructor gratefully acknowledge the participation of our industrial collaborators.

1. https://www.brightpod.com/boost/10-popular-time-management-techniques [↑](#footnote-ref-1)
2. https://code.visualstudio.com/docs/languages/overview#:~:text=On%20this%20website%2C%20we%20have,%2D%20T%2DSQL%20%2D%20TypeScript. [↑](#footnote-ref-2)