# Product Requirements

**Project Name**: Mind\_Reader

**Team Name**: Single Semester Snobs (Original, 2021) | We Don’t Byte (Continued, Spring 2022)

| Five Guys (Continued, Fall 2022 )

| Revision # | Revision Date | Summary of Changes | Author(s) |
| --- | --- | --- | --- |
| 0.1 | 9/21/2021 | Initial Requirement Document | Mason Bone, Sophia Drewfs, Jake Grossman, Josiah Moses, Cal Wooten |
| 0.2 | 10/19/2021 | Added New Key features and a NF Requirement | Mason Bone, Sophia Drewfs, Jake Grossman, Josiah Moses, Cal Wooten |
| 0.3 | 12/1/2021 | Updated Final Requirements | Mason Bone, Sophia Drewfs, Jake Grossman, Josiah Moses, Cal Wooten |
| 0.4 | 3/26/2022 | Add functional requirement H9: | John Breaux, Kendrick Johnson, Pedro Alvarez, Ryan Tolbert, Thomas Lane |
| 0.5 | 4/9/2022 | Add functional requirement H10: | John Breaux, Kendrick Johnson, Pedro Alvarez, Ryan Tolbert, Thomas Lane |
| 0.6 | 5/7/2022 | Add functional requirement R10: | John Breaux, Kendrick Johnson, Pedro Alvarez, Ryan Tolbert, Thomas Lane |
| 0.7 | 9/22/2022 | Add functional requirement R11, R12, H11, and H12: | Zachary Chenausky, Jigme Rinji Sherpa, Clay Lewis, Haris Javed, Saad Javed |

### 

### Brief problem statement

The current editor options available lack the level of accessibility that is required to allow students who are visually impaired to adequately edit, write, and debug code. This tool would extend Visual Studio Code’s existing accessibility options to allow everyone, primarily students K-12, to learn programming with Lego Mindstorms. Our goal is to provide an enhanced experience for students who are visually impaired that is transparent to sighted students. This allows for everyone to use the same software solution, whether or not they are vision impaired.

### System requirements

● Microsoft Visual Studio Code

● OS X El Capitán (10.11+)

● Windows 7 (with .NET Framework 4.5.2), 8.0, 8.1 or 10 (32-bit and 64-bit)

● Mindstorm EV3, Spike Prime or Inventor

● For the EV3 a MicroSD card in required

● For Spike Prime a USB port is required

### Users profile

The system is intended for everyone, but primarily for students K-12 who are visually impaired. The user should have an interest in coding and robots and have a background in using computer keyboards. The user should have general knowledge.

### List of Features

| F1 | Accessibility Pane |
| --- | --- |
| F2 | Audio Alerts |
| F3 | Navigation Hotkeys |
| F4 | Settings Window. More advanced than the menu, global “on/off”, not just adjusting the value of the setting. |
| F5 | Document Parsing |
| F6 | Hub Connectivity |
| F7 | Voice-to-Functions |

### Functional Requirements (user stories)

| No. | User Story Name | Description | priority |
| --- | --- | --- | --- |
| R1 | Open Accessibility Menu | The user can use a mouse and scroll over the Accessibility menu icon which will open a drop down menu with useful tools. Or the user can use a hotkey to open the same menu moving the cursor to the first item in the drop down menu. | 1 |
| R2 | Change Theme | The user will open the accessibility menu and move the cursor down to a change theme button which will then open another menu with options for themes for the instance of visual studio code. | 1 |
| R3 | Change Font Size | Allow users to increase or decrease font size for better visibility. | 1 |
| R4 | Connect to Hub | The user can select from 1 or more connected LEGO Hubs to connect to. | 2 |
| R5 | Disconnect from hub | The user can disconnect from the hub | 1 |
| R6 | Save Files to Hub | User can save/upload Python files to the Hub | 1 |
| R7 | Run Files from Hub | User can run Python files that are on the Hub | 1 |
| R8 | Delete Files from Hub | User can delete Python files currently on the Hub | 1 |
| R9 | Stop Execution | Users can stop a program currently running on the Hub. | 1 |
| R10 | Line Highlighter | The line containing the cursor is automatically highlighted. With options to select multiple lines, customize the highlighter, and disable the highlighter. | 1 |
| R11 | Speech-to-Function | The user will have predefined speech to written function options available for voice accessibility. | 2 |
| R12 | Voice Hotkey | The user will be able to say specific keywords to trigger any specific hotkey functionality. | 1 |

| Hotkeys | | | |
| --- | --- | --- | --- |
| H1 | Navigate Interface (not text editor) | - Arrows for small movements  - Tab for large movements  - Enter for accept & continue  - Spacebar for focus in menu, but don’t click (:focus in CSS) | 1 |
| H2 | Navigate Text Editor | Hotkeys to go to lines, functions, and definitions. | 1 |
| H3 | Get Input Help | Enter a help mode where pressing a key/key combination will read the keypresses to the user, along with the name of the function it maps to.  This could also possibly help with the learning curve when first using the editor. | 2 |
| H4 | Listen to cursor  context | Read the text at the user cursor with context (a word or two on either side). This can be used when making small changes to avoid listening to the whole line, just reading the part the user is interested in. | 2 |
| H5 | Listen to line context | Read the user a summary of the current scope of the current line, i.e.: “function foo inside class bar”. | 1 |
| H6 | Get Indentation Level | Read the user the current indentation level (text description, along the lines of “Line {line\_number} has {depth} indentations”.) | 1 |
| H7 | Be alerted of syntax errors | When saving, if there is a syntax error, play a sound. | 2 |
| H8 | Goto syntax errors | Hotkey to cycle through & read errors. | 2 |
| H9 | Get number of spaces | Read the user the current indentation level (text description, along the lines of “Line {line\_number} has {depth} indentations”.) | 1 |
| H10 | Get line number | Read the line number of the line containing the cursor | 1 |
| H11 | Move cursor | The hotkey will move the cursor to the start of the program or to the end. | 2 |
| H12 | List Autocomplete Options | Using the hotkey after typing a partial function or variable will speak the Intellisense autocomplete options to the user, receiving either a keyboard or voice prompt for selection. | 1 |

### Non-Functional Requirements

|  | Non-Functional Requirement | Category | Category Description |
| --- | --- | --- | --- |
| NF1 | WAI-ARIA technical specifications | Usability | The system will satisfy the requirements specified by the WAI-ARIA specifications. |
| NF2 | Transparent use for students with sight. | Usability | Usability The system should not alter the typical workflow of VScode. |
| NF3 | VS Code  Cross-Platform | Cross-Platform Compatibility | The system will support macOS, Linux, and Windows. |
| NF4 | Text-to-Speech Accuracy | Accuracy | The text-to-speech function should be accurate for all language features of Python. |
| NF5 | Screen Reader | Usability | Usability The UI should be readable by Voice-over, JAWS, NVDA |
| NF6 | VS code Extension requirements | Usability | The extension should be able to be published as a public extension in the extension market. |
| NF7 | Speech-to-Text Accuracy | Accuracy | The speech-to-function feature should be able to understand users well to present the functions they intend. |

### Sponsor Requirements

I have read and approved the material in this document. If there is no external sponsor, the TA or the instructor will sign it for accuracy/scope.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_

Print Name Signature Date