Second Semester Work Statement

Date: 2/6/2022
Team Name: Techblazers
Team Members: Slate Jordan, Chase Williams, Victor Siooh, Emmit Brandt, Chief Boateng

Introduction:

The Keyboard Finger Position Display is a product that enables the visually impaired to see which keyboard keys are being touched without them needing to look down at a physical keyboard. There are very few systems to assist the visually impaired type, and even fewer that provide feedback to the user. This product will allow the user to receive helpful feedback, and adjust it according to their needs.

This work statement will outline the scope, period of performance, location of work, tasks, deliverables, work performance, acceptance criteria, and requirements of the Keyboard Finger Position Detector and Display. The purpose of the statement of work is to provide clear goals the team needs to accomplish before we can donate our device and app to Envision, an organization in Wichita that assists those with low vision. This will serve as an accountability tool, letting each team member know what they need to get done and when.

Scope:

The Keyboard Finger Position Display senses when a keyboard key is touched, and it displays the key in an on-screen keyboard app within the user's computer. The user will be able to adjust different settings such as the font type, font size, font color, background color, the color of each key, and the onscreen keyboard's transparency for better visibility. The major activities on this project can be broken down into three main categories. The first will be purely physical getting all of the required hardware fitted and installed in the base keyboard including any custom enclosures necessary. The second will be the calibration and configuration of sensor hardware as well as the communication with the computer. Lastly, the design and coding of our overlay application to effectively and conveniently display data from the hardware sensors on screen.

Technical Assumptions

- Initially, we added aluminum to the top surface of each key and directly connected a wire from the aluminum to a capacitive sensing node. The work we will do this semester will confirm whether we are able to detect a touch on the outside surface of a nonmetallic keyboard key with a wire connected from a capacitive sensing node to the inside surface of the key.
- For the software application development, we will need to make sure the app is able to respond after the user interacts with the keyboard, within a duration below 70ms, which was an estimate that was calculated to help ensure a smooth display of the keys and to make the delay unnoticeable from users.

Ground Rules

- Each team member will be responsible for completing the tasks and deliverables assigned to them within this document.
- During the weekly team meeting on Teams, each team member will be required to share what they have contributed to the project and what their progress is on their assigned tasks.

Applicable Standards

- Restriction of Hazardous Substances Directive 2002/95/EC
- USB 2.0 Standard
- Windows app design guidelines
- ISO/IEC 9995-2

Standard Methodologies

- Iterative approach
- VS Code Python Programming

Period of Performance:

Week 1 - Week 2: We need to acquire the hardware components that were finalized such as the TCA9548A multiplexer, seven MPR121 12-key capacitive touch sensors, and copper. Without these materials, the team will not be able to build and test the hardware mentioned in the next section. Also, the hardware is needed, so we can test whether the microcontroller outputs a touch when the user touches the top of a keyboard key. At the beginning of the semester, the team had already obtained all the material listed above.

Week 3 - Week 10: We will integrate the TCA9548A Multiplexer, seven MPR121 12-key capacitive touch sensors, and attach the metal material underneath the keyboard keys. A compartment will be constructed to house all the hardware that cannot fit in the keyboard. Testing will be implemented over the ability for the microcontroller code to correctly output touches on the keys. We will also modify the keyboard display code in

Visual Studio to read the touches the microcontroller outputs from all the keys. Finally, we will add features to the keyboard display app such as customizable fonts, colors, and transparency.
Week 11 - Week 16: We will test and debug any errors within the code for the onscreen keyboard and its additional features. Durability testing of the hardware component will ensue. As the Keyboard Finger Position Display is completed, we will prepare to deliver the device to Envision.

Location of Work:

The primary location for the production of the Keyboard Finger Position Display will be the Project Innovation Hub in the John Bardo Center at Wichita State University. This facility provides the team members with numerous tools (such as soldering equipment) to utilize when working on the design of the hardware. Consequently, the integration of the hardware will take place there.

Another work activity that will take place in the Project Innovation Hub is the testing of the microcontroller and keyboard app code. While test data can be generated and fed as input to the code for the app, it is ideal to test the entire system at once with the whole team present to resolve any bugs. While testing of the software will be done at the Project Innovation Hub, most of the software development can be done from each programmer's home.

Tasks and Deliverables

Project Phase	Team Member(s)	Work Activity
Project Planning	Slate/Victor	Create a list of all the hardware that still needs to be acquired.
Project Planning	Slate/Victor	Acquire all the hardware
Project Planning	Whole Team	Meet and confirm what tasks each team member is responsible for completing and when they should complete them.
Design	Chase/Slate	Review the documentation of the TCA9548A multiplexer to determine how to correctly integrate it with the touch sensors in our sensor circuit.

Design	Chase/Slate	Create an integration plan for the TCA9548A multiplexer and touch sensors.
Design	Victor/Slate/Chase	Research what the form of the conductive material underneath the key will be.
Design	Victor/Slate/Chase	Create a report detailing what the pros and cons are of each form option and report this to the team, so a decision can be made about which form will work best for our device.
Design	Victor/Slate/Chase	Research different methods of attaching the metal underneath each key.
Design	Victor/Slate	Determine how much space is available inside of the keyboard to fit all the hardware: the sensors, multiplexer, Arduino Nano, and wires.
Design	Victor/Slate	Report to the group whether there is enough space to fit all the hardware in the keyboard.
Design	Victor/Slate	If we determine there is enough space to fit all the hardware inside of the keyboard, this task will involve us figuring out where to place the sensors, multiplexer, and Arduino Nano.
Design	Victor/Slate	If there is not enough space to fit all the hardware inside the keyboard, we will design a compartment that will house the sensors, multiplexer, Arduino Nano, and some portions of the wires.
Design	Victor/Slate	Show the 3D model/drawing of the compartment design to the group for review
Design	Victor/Slate/Chase	Design how we will route the wires from the touch sensors to the inside

		surface of each keyboard key.
Build and Construction	Victor/Slate/Chase	Integrate the TCA9548A multiplexer with all the touch sensors in our capacitive sensor circuit.
Build and Construction	Victor/Slate	If we decide to build the hardware compartment, we will need to 3D print it.
Build and Construction	Chase	Adjust the Arduino code to read touches from the additional capacitive sensors that are connected to the multiplexer.
Build and Construction	Emmitt/Chief	Adjust the Python code in Visual Studio to display touches from any key on a 60% keyboard.
Build and Construction	Emmitt/Chief	Give the user the ability to customize the contrast of the keyboard.
Build and Construction	Emmitt/Chief	Give the user the ability to customize the transparency of the keyboard.
Build and Construction	Emmitt/Chief	Give the user the ability to customize the font types of the keyboard.
Build and Construction	Emmitt/Chief	Give the user the ability to customize the font sizes of the keyboard.
Build and Construction	Emmitt/Chief	Give the user the ability to customize the key sizes of the keyboard.
Build and Construction	Emmitt/Chief	Update the Python code to give users the option to hear which keys are being touched.
Build and Construction	Whole Team	Meet at least weekly to report each team member’s progress in their work.
Testing	Victor/Slate/Chase	Test the different methods of attaching the metal underneath the key to find the method that is most durable.

Testing	Chase/Slate	Add the raw ADC counts data from the capacitive touch sensors output to a spreadsheet.
Testing	Chase/Slate	Create a plan to test how accurate the touch detection is in the device.
Testing	Chase/Slate	Test the accuracy of the touch detection and to find the ADC counts threshold that yields the best results.
Testing	Chase/Slate	Test the integration of the MPR121 12-Key Capacitive Touch Sensors with the TCA9548A multiplexer to see if touches can be detected on all capacitive sensing nodes.
Testing	Chase/Slate	Create a table that shows which keys were pressed and whether the correct keys are shown in the on-screen keyboard.
Testing	Emmitt/Chief	Create a plan to determine how to test the customizable features of the onscreen keyboard.
Testing	Emmitt/Chief	Test the customizable features of the onscreen keyboard to ensure operability (i.e transparency, font color change, font type change, font size change, etc.)
Testing	Emmitt/Chief	Test the onscreen keyboard application to ensure that it will grant access to a user.
Demonstration	Whole Group	Send the final product to Envision and do a live demo.

Work Performance (integrated schedule):

Project Phase	Team Member(s)	Work Activity	Timeline

Project Planning	Slate/Victor	Create a list of all the hardware that still needs to be acquired.	01/18 - 01/26
Project Planning	Slate/Victor	Acquire all the hardware	01/18 - 01/21
Project Planning	Whole Team	Meet and confirm what tasks each team member is responsible for completing and when they should complete them.	01/18 - 05/06
Design	Chase/Slate	Review the documentation of the TCA9548A multiplexer to determine how to correctly integrate it with the touch sensors in our sensor circuit.	02/13 - 03/06
Design	Chase/Slate	Create an integration plan for the TCA9548A multiplexer and touch sensors.	02/13 - 03/06
Design	Victor/Slate/ Chase	Research what the form of the copper underneath the key will be.	01/25 - 02/25
Design	Victor/Slate/ Chase	Create a report detailing what the pros and cons are of the conductive material form options and report this to the team, so a decision can be made about which form will work best for our device.	02/06 - 02/25
Design	Victor/Slate/ Chase	Research different methods of attaching the metal underneath each key.	02/10 - 02/22
Design	Victor/Slate	Determine how much space is available inside	02/06 - 02/18

		of the keyboard to fit all the hardware: the sensors, multiplexer, Arduino Nano, and wires.	
Design	Victor/Slate	Report to the group whether there is enough space to fit all the hardware in the keyboard.	02/06 - 02/28
Design	Victor/Slate	If we determine there is enough space to fit all the hardware inside of the keyboard, this task will involve us figuring out where to place the sensors, multiplexer, and Arduino Nano.	02/06 - 02/28
Design	Victor/Slate	If there is not enough space to fit all the hardware inside the keyboard, we will design a compartment that will house the sensors, multiplexer, Arduino Nano, and some portions of the wires.	02/06 - 02/28
Design	Victor/Slate	Show the 3D model/drawing of the compartment design to the group for review	02/06 - 03/06
Design	Victor/Slate/ Chase	Design how we will route the wires from the touch sensors to the inside surface of each keyboard key.	02/26 - 03/14
Build and Construction	Victor/Slate/ Chase	Integrate the TCA9548A multiplexer with all the touch sensors in our capacitive sensor circuit.	02/13 - 02/27
Build and	Victor/Slate	If we decide to build the	02/27 - 03/13

Construction		hardware compartment, we will need to 3D print it.	
Build and Construction	Chase	Adjust the Arduino code to read touches from the additional capacitive sensors that are connected to the multiplexer.	03/14 - 03/25
Build and Construction	Emmitt/Chief	Adjust the Python code in Visual Studio to display touches from any key on a 60% keyboard.	2/25-3/04
Build and Construction	Emmitt/Chief	Give the user the ability to customize the contrast of the keyboard.	3/04-3/11
Build and Construction	Emmitt/Chief	Give the user the ability to customize the transparency of the keyboard.	3/04-3/11
Build and Construction	Emmitt/Chief	Give the user the ability to customize the font types of the keyboard.	2/04-2/11
Build and Construction	Emmitt/Chief	Give the user the ability to customize the font sizes of the keyboard.	2/011-2/18
Build and Construction	Emmitt/Chief	Give the user the ability to customize the key sizes of the keyboard.	2/18-2/25
Build and Construction	Emmitt/Chief	Update the python code to give users the option to hear which keys are being touched.	3/11-3/28
Build and Construction	Whole Team	Meet at least weekly to report each team member’s progress in their work.	01/27 - 05/01

Testing	Victor/Slate/ Chase	Test the different methods of attaching the metal underneath the key to find the method that is most durable.	02/06 - 03/20
Testing	Chase/Slate	Create a plan to test how accurate the touch detection is in the device.	02/13 - 03/13
Testing	Chase/Slate	Test the accuracy of the touch detection and to find the ADC counts threshold that yields the best results.	02/13 - 03/27
Testing	Chase/Slate	Add the raw ADC counts data from the capacitive touch sensors output to a spreadsheet.	02/13 - 03/27
Testing	Chase/Slate	Test the integration of the MPR121 12-Key Capacitive Touch Sensors with the TCA9548A multiplexer to see if touches can be detected on all capacitive sensing nodes.	03/13 - 03/27
Testing	Chase/Slate	Create a table that shows which keys were pressed an whether the correct keys are shown in the on-screen keyboard.	03/20 - 04/10
Testing	Emmitt/Chief	Create a plan to determine how to test the customizable features of the onscreen keyboard.	1/28-2/11
Testing	Emmitt/Chief	Test the customizable features of the onscreen keyboard to ensure operability (i.e transparency, font color change, font type	3/04-3/11

		change, font size change, etc.)	
Testing	Emmitt/Chief	Test the onscreen keyboard application to ensure that it will grant access to a user.	3/04-3/11
Demonstration	Whole Group	Send the final product to Envision and do a live demo.	5/06

Acceptance Criteria

Deliverable	How deliverables are validated	Who is authorized to accept deliverables	Where deliverables will transfer ownership
Create an integration plan for the TCA9548A multiplexer and touch sensors.	The plan will be presented to the group in a Teams meeting to assess whether it has been completed. In addition, we will implement this plan to assess whether it allows the hardware to function properly.	Whole Team	Github
Create a report detailing what the pros and cons are of each form option and report this to the team, so a decision can be made about which form will work best for our device.	The report will be presented to the group in a Teams meeting to assess whether it has been completed.	Whole Team	Teams
Report to the group whether there is enough space to fit all the	Space availability will be validated from a report that details the	Whole Team	Teams

hardware in the keyboard.	reasons for a specified area to house hardware.		
Adjust the Arduino code to read touches from the additional capacitive sensors that are connected to the multiplexer.	The code will be uploaded to Github, so those who are testing the touch detection can verify whether current sensitivity value leads to accurate touch detection.	Slate/Chase	Github
Optimize the application enough that it does not lag behind the user's typing	A manual testing method could be used alongside regression testing. This is where we have a member test the new features of the app to ensure it works as intended	Emmitt/Chief	Github
Show the 3D model/drawing of the compartment design to the group for review	The model/drawing will be validated based on its dimensions which must adhere to the area of interest on the keyboard and be able to house the hardware.	Victor/Slate	Github
Create a plan to test how accurate the touch detection is in the device.	The plan will be presented to the group in a Teams meeting to assess whether it has been completed.	Whole Team	Github
Create a plan to determine how to test the customizable features of the onscreen keyboard.	A test plan will be created for each customizable feature, and it could be given to another member of the group to follow to	Emmitt/Chief	Github

	ensure those features work as intended		
Create a table that shows which keys were pressed and whether the correct keys are shown in the on-screen keyboard.	The whole team will inspect the data to confirm that we have 100% accuracy.	Whole Team	Github
Add the raw ADC counts data from the capacitive touch sensors output to a spreadsheet.	This data will be compared to the ADC counts sensitivity threshold in the program that yields the most accurate results to make sure the threshold aligns with what the raw data shows.	Chase/Slate	Github
Send the final product to Envision and do a live demo.	Stacy Fuller would like to see a demonstration of the device and inspect it to see if it should be added to Envision's suite of accessibility tools.	Stacy Fuller	Envision's facility in Wichita.

Special requirements:

Unique Skills

- Soldering
- Python programming
- C# programming

Data Handling

- Github
    - Used to store code and its associated data after debugging

Access Requirements

- The whole team will be granted access to view and upload code

Chase Williams 02/06/22

Victor Siooh 02/06/22

Slate Jordan 02/06/22

Emmitt Brandt 02/06/22

Chief Boateng 02/06/22