SENECA
POLYTECHNIC

BRAIN CAP

INSTRUCTOR: KAREN CRAIGS COURSE
CODE: TPJ655NCD

NAME WANPOSOP SURAYUT
STUDENT ID 168000180

ABSTRACT

TABLE OF CONTENT

TABLE OF FIGURES

Acknowledgements

I would like to express our heartfelt gratitude to our instructor, Karen Craigs, for her exceptional support, guidance, and encouragement throughout the duration of this project. Her insights and expertise were instrumental in helping me navigate both technical and creative challenges. I'm thankful for her patience, feedback, and the opportunities she gave me to push ourselves beyond my limits.

I would also like to thank our classmates and friends who helped us test the Braincap prototype and provided valuable feedback. Their honest opinions and suggestions helped me improve the system's design and functionality. Finally, I sincerely appreciate the continued motivation and support from our families. Their encouragement during late nights and tight deadlines meant a lot , and this project would not have been possible without them.

Project Summary

Braincap is a smart, wearable game system that transforms the classic guessing game of Charades into an interactive technological experience. Designed to bring people together through fun and teamwork, it uses modern components like an ESP32 microcontroller, sensors, displays, and feedback systems to create an immersive and hands-free gameplay experience.One player wears the helmet, while another enters a word using a web-based dashboard. That word is displayed on an OLED screen on the helmet for others to see — but not the wearer. The player wearing the helmet then guesses the word based on the other person's gestures. The system responds to inputs like voice commands, head movements, or manual controls and gives real-time feedback through lights, sounds, and vibrations. With features like a timer, RGB LEDs, a microphone, and accelerometer, Braincap is not just a game — it's a blend of nostalgia and innovation. It aims to modernize traditional games while keeping the spirit of togetherness and fun alive for all age groups.

INTRODUCTION

In today's time, most people spend a lot of hours playing games alone on mobile phones or gaming consoles. But earlier, people used to enjoy playing simple group games with friends and family. One such game is Charades, where players guess a word or movie name based on actions. This game always brings joy, laughter, and bonding among people. So, we thought — why not bring this old game into the modern world using technology?

My project, Braincap, is a smart helmet that makes this guessing game more fun and interactive. Instead of using paper or sticky notes, we use an OLED screen on the helmet to show the word. The person wearing the helmet cannot see the word, but others can — just like in the original game. Using voice input, head movements, lights, and sounds, the system helps players guess the word in a fun, hands-free way.

I used many components like the ESP32 microcontroller, ultrasonic sensor, microphone, rotary encoder, 7-segment display, and RGB LEDs to build this system. It is fully wireless and works with a web-based dashboard to add words, start the game, and set the timer.

The aim of Braincap is to bring people together using a mix of traditional games and modern tech. It is simple to use, suitable for all age groups, and adds excitement to any party, classroom, or family gathering. This project shows how technology can help make old games even more enjoyable for today's generation.

BACKGROUND RESEARCH & DEVELOPMENT

HISTORY OF THE GAME

This game is inspired by a traditional party game where people write a movie name on a sticky note and paste it on another person's forehead. The person wearing the note has to guess the movie based on the actions of others. This game has been played for years at parties and gatherings because it is simple, fun, and brings people together. But no one has ever created technology for this game. We looked for background research, but there was nothing similar, so we decided to develop our own tech-integrated version.

HOW IT'S USED TODAY

This traditional game is still played manually, using sticky notes or paper. However, with advancements in wearable technology, we saw an opportunity to make it more interactive and exciting. Instead of writing on paper, we created a smart helmet where one player wears the device, and another player types the word on a website. The word appears on an OLED display on the helmet, and the player has to guess it based on actions of the other person.

PAST RESEARCH & FINDINGS

Even though no one had created technology for this game, studies show that interactive games improve social engagement and cognitive skills. Research on haptic feedback (vibrations) suggests that it makes games more immersive and enjoyable. Other studies on voice recognition prove that hands-free interaction can make gameplay smoother and more accessible for all age groups.

HOW BRAINCAP IMPROVES EXISTING TECHNOLOGY

Braincap takes a simple, old game and adds modern technology to it, making it more fun and exciting. Instead of using sticky notes, we use an online system and a smart helmet. Instead of guessing randomly, the player gets haptic and sound feedback to make the experience better. This game can be played by young and old people, bringing everyone together for a fun and entertaining time.
This is not just a project it’s a new way of playing an old favorite. We believe Braincap can make group games more exciting, more interactive, and way more fun!

SYSTEM FUNCTIONAL FEAUTURES & SPECIFICATIONS

This system is designed to provide an interactive and fun experience by combining various components such as sensors, processors, and output devices. It works by integrating multiple inputs and outputs to enhance the gameplay, making it engaging and dynamic. Below is a description of the system's inputs and outputs, which are crucial in achieving the desired interactive experience.

Inputs:

The following are the inputs used in the system:

1. Microphone
2. Rotary Encoder
3. Ultrasonic Sensor
4. Accelerometer

These inputs allow the system to capture a variety of actions, from voice commands to physical gestures, making the game more immersive and interactive.

Outputs:

The system uses the following output devices to provide feedback and enhance user interaction:

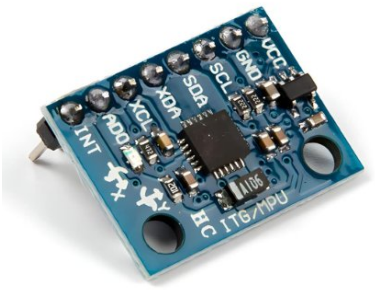
1. OLED Display
2. Vibration Motor / Buzzer
3. 7-Segment Display
4. RGB LED

INPUTS:

Ultrasonic Sensor:

The ultrasonic sensor is a very important part of our project. It helps to check if the helmet is properly worn on the user's head. This sensor sends out sound waves that travel fast and bounce back after hitting something. The sensor measures the time taken for the wave to return and calculates the distance. If the distance is short, it means someone is wearing the helmet, and then only the game will start. This helps in avoiding false starts when the helmet is just lying on a table or somewhere else. It also helps to avoid cheating—because if the helmet is not on the head, the game won't run. So the guessing person must wear it properly to play. Ultrasonic sensors are also commonly used in cars, robots, and many smart machines to measure distance or avoid obstacles. In our game, this sensor improves both fairness and user experience.

Accelerometer:



The accelerometer is one of the most important sensors used in our Braincap system. It is a small electronic component that can sense movement, direction, and tilt. In simple words, it helps the system understand how the helmet is moving.

In our project, the accelerometer is used to detect head movements of the person who is wearing the Braincap. For example, if the person nods their head, the system can understand it as a "yes". If they shake their head, it can be taken as a "no". These movements can be used in future updates to select options or give answers without using hands—making the game more interactive and fun.

This sensor works on three axes (X, Y, and Z), so it can understand movement in all directions. It makes the system more smart and hands-free. It also adds a layer of safety—for example, if the person wearing the helmet suddenly falls or moves fast, the system can react accordingly.

In future versions of Braincap, we plan to use the accelerometer for even more features, like menu navigation using head tilts, or starting/stopping the game just by moving your head. It's especially useful for kids and elders who may not want to press buttons or touch screens.

In short, the accelerometer makes Braincap more responsive, smart, and easy to use, by turning natural head movements into real game actions.

Microphone:

In Braincap, the microphone plays the role of a voice listener. It is the part that helps the system hear what the user is saying. This feature allows the person wearing the helmet to speak their guesses instead of only depending on actions or body language.

We added this to make the game more engaging and user-friendly, especially for people who are comfortable speaking instead of making gestures. It also helps in situations where hand movement is limited or not possible.

During testing, we found that the microphone works best in quiet surroundings. So we planned the game in such a way that it's played in places with less noise, and we also wrote clean code to filter unwanted sounds.

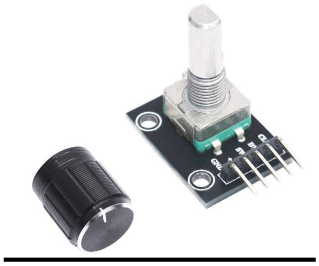
By using voice-to-text conversion, the spoken word can be changed into text and checked against the correct answer. This brings in a new way of interaction where the helmet understands what the user says.

In the future, we plan to upgrade this feature by using AI-powered speech recognition that will understand more words clearly and maybe even different languages.

The microphone helps us achieve a hands-free experience, making the game smoother, especially for children and elderly people, who may find speaking easier than other inputs.

This component is key in turning Braincap into a modern and smart gaming device that listens, understands, and responds—all from the power of voice.

Rotary Encoder Switch:



The rotary encoder is a small knob that you can turn or press to control the Braincap system. It helps the user to move through menus, select game options, and adjust settings in a simple way. This input device makes it easier for the user to navigate the system without using a touchscreen or complicated buttons. By rotating the knob, the user can scroll through different options, and by pressing it, they can choose their selection. This is especially useful for people who prefer physical interaction with the system, like kids, adults, and older users who might not be comfortable with touchscreens.

In the Braincap project, the rotary encoder is used to make the system easy to control. It gives a clicking sound and a physical feel when you turn it, which tells the user that their action has been registered. This makes the experience more interactive and satisfying. The rotary encoder is durable, so it will last longer and doesn't rely on fragile parts like touchscreens. It is a simple and effective way to control the system, especially in situations where users need a reliable, easy-to-use control. In the future, we plan to add more functions to the rotary encoder, like volume control or screen brightness, making the system even more versatile and user-friendly.

OUTPUTS

OLED Display:

The OLED display is one of the most important parts of the Braincap system. It shows the crucial information, like the word that needs to be guessed. Without this display, the game wouldn’t be able to function properly, even if other components were working. The display is small, clear, and mounted on the helmet so only the other players can see the word while the wearer tries to guess it. It provides real-time visual feedback and makes the game more interactive.

In the Braincap project, the OLED display plays a key role in ensuring the game runs smoothly. It shows the word the player needs to guess, and without it, the whole concept of the game falls apart. Even though the game could work without other outputs like sound or vibrations, it cannot work without the display. The sharp and bright visuals ensure that the user can clearly see the word, making it easy to play. This display is essential for creating an engaging experience and is one of the core components of the game.

The OLED display is not just another output device; it is the heart of the gameplay experience. Since the person wearing the helmet cannot see the word, it is very important that the display is clear and easy to read. The OLED display makes sure that everyone except the player can easily see the word. This helps the game run smoothly and quickly. Without this display, the game wouldn't work properly because there would be no way to show the word that needs to be guessed. This makes the OLED display the most important part of the system. It helps keep the game fun and **interactive**, and without it, the whole game idea would not be possible.

Buzzers:

Buzzers are small sound-making parts that help give audio feedback to the user. In this project, we use a buzzer to make sounds when something important happens in the game like when the game starts, when a player gives a correct answer, or when the time runs out. These sounds help players stay alert and understand what's happening even if they are not looking at the screen. It adds another way for the system to communicate with the user.

Using a buzzer makes the Braincap more fun and interactive. Just like how we hear beeps in a microwave or washing machine, here also the buzzer gives quick and simple sound alerts. In future versions, different types of sounds can be used to give hints, warnings, or game instructions. It is a small but useful part that helps make the whole system more exciting and easy to use, especially in noisy or busy places.

RGB Lights:

RGB lights are colourful lights that make the Braincap look more attractive and fun. They are used mainly for decoration, but they also help in showing what is happening in the game. For example, the lights can turn green when the answer is correct, red when it’s wrong, or blue when the game is running. These lights catch everyone’s attention and make the experience more enjoyable, especially in group settings like parties or classrooms.

Using RGB lights also helps people who may not hear the buzzer or see the screen properly. The changing colours give a clear idea of the game status. It makes the game feel more like a real tech gadget, just like fancy gaming systems or smart devices. In the future, these lights can be made to blink in patterns or match the mood of the game, adding even more fun to the experience.

7-Segment Display:

The 7-segment display is used to show the timer in the game. It helps players know how much time is left while guessing the word. The numbers are shown clearly, so everyone around can easily see them. The user can set the time limit for each round using the online dashboard or website connected to the Braincap system.

SYSTEM DESIGN & LAYOUT

Hardware components & specifications

Detailed List of Hardware Components
The Braincap system consists of various sensors, output devices, and a microcontroller that work together to make the game interactive and fun. Below is a list of the main components used in the project.

1. ESP32 (Microcontroller)
· The main controller that manages all sensors and outputs.
· Handles wireless communication between the helmet and the website.
· Supports I2C, UART, and Wi-Fi for smooth data transfer.
2. Ultrasonic Sensor
· Detects if the helmet is worn by measuring the distance from the user's head.
· Ensures the system only activates when properly worn.
3. Accelerometer
· Tracks the head movement and tilt of the user.
· Helps in gesture-based interactions.
4. OLED Display
· Shows the game word and other instructions.
· Compact 128x64 resolution for clear visibility.
5. 7-Segment Display
· Displays the game timer for countdown.
· Ensures players stay aware of time limits.
6. Rotary Encoder
· Allows manual selection of options in the game.
· Provides tactile feedback for better control.
7. Microphone
· Captures voice input for interactive gameplay.
· Helps in speech-based guessing.
8. Buzzer
· Provide haptic feedback when needed.
· Enhances the game experience with sound alerts.
9. RGB Lights
· Used for decorative effects and to indicate game progress.
· Changes color based on game events.
10. Rechargeable Battery
· Provides portable power for the system.
· Ensures the game can be played anywhere.

Theory of Operation

The OLED Hat Project is a smart and entertaining wearable device that mixes electronics and creativity. It is made using different types of sensors and output components that all work together. These sensors collect data and send it to a small computer called a microcontroller, which controls everything. The main aim of this system is to give users an interactive and fun experience by using simple but smart technology.

The game is played between two people. One person wears the helmet (hat) that contains all the sensors and displays. The second person, who is not wearing the helmet, types a word into the system dashboard using a mobile or computer. That word is shown on a small OLED display screen on the helmet. The twist is that the person wearing the helmet cannot see the screen. Their job is to guess the word by observing the actions of the second person. The second person has to act out the word without speaking. Based on the actions, the helmet wearer guesses the word by saying it aloud.

The system uses an ultrasonic sensor to know when the helmet is worn. This sensor measures the distance between the head and the sensor. If the distance is correct, it knows someone is wearing the helmet. This helps the system turn on only when needed. It saves battery and makes sure the game does not start by accident. Along with this, the accelerometer checks if the helmet is being worn properly. It detects movement and tilt. Only when the hat is in the correct position does the game begin. These sensors make sure the system works only when everything is ready.

The accelerometer plays a special role during gameplay too. It detects head movements like nodding or shaking. If the helmet wearer nods their head, the system takes it as a "No" input. If they shake their head side to side, it is taken as a "Yes" input. This adds a natural and fun way for the player to give responses without using their hands.

Once the player is ready and the system is turned on, the game begins. A rotary encoder (a type of switch) is used by the helmet wearer to control the game. It helps in starting the game, skipping a word, or selecting an option. This gives the player manual control when needed, especially if the voice or motion input is not detected. The microphone is another key input part. It listens to the voice of the person wearing the helmet. When the player says

the guessed word, the system uses software to change voice to text. Then it checks if the guessed word is correct. If it is correct, it moves on to the next word. If not, it waits for the correct answer.

There is also a timer system in the game. The timer starts as soon as the game begins. The 7-segment display shows how much time is left. The user can set the game duration using the dashboard before starting. The player has to guess as many words as

possible before the time runs out. The game becomes exciting and challenging as time counts down. If the player guesses the word correctly, the next word is shown automatically, and the game continues until the timer ends.

The OLED display is the most important output device in this system. It shows all the important information like the word to be guessed, game start messages, or timer updates. This display is small but very clear. Without this display, the game will not work at all. Even if other components are not used, the OLED display is necessary for showing the game word. The clear and bright screen helps make the experience smooth and fun.

Other output devices like vibration motors, buzzers, and speakers are used to make the experience better. For example, if the player says the correct word, the vibration motor gives a small buzz to show it was correct. The buzzer or speaker also makes a sound to confirm the answer. If the answer is wrong or not heard clearly, different feedback is given to help guide the player.

The RGB lights are used to make the game more colorful and fun. These lights can change color to show different things. For example, green for correct answer, red for wrong answer, and blue when time is up. These lights are mainly for decoration but they also help the player understand what is happening in the game.

All these devices are controlled by the ESP32 microcontroller. It is like the brain of the system. It connects to all the sensors and output parts and tells them what to do. The ESP32 supports both 3.3V and 5V, so we can connect

many different types of components. The whole system is powered by a rechargeable power bank. This makes the helmet easy to carry and use without wires.

The parts in the system talk to each other using serial communication. This means one part sends information and another receives it. For example, the microphone sends data to the microcontroller, and the controller checks it and then sends commands to the buzzer or display. This type of communication keeps the system fast and smooth.

This version of the OLED Hat Project is just the beginning. It is like the first version of popular gaming consoles like PS1 or Xbox 1. In the future, we plan to add more games, better graphics, a touch screen, and even online multiplayer features. The idea is to create a fun and simple gaming helmet that can be used by many people.

We also want to improve battery life, make the helmet lighter, and add more traditional Indian games. We plan to build a big library of more than 100 games in the next few years. For now, this project shows how we can use small electronics and smart ideas to build a cool and fun wearable game.

This helmet brings together fun, learning, and technology. It encourages teamwork, improves guessing skills, and provides a new way to enjoy simple games. The smart use of sensors and feedback makes it more engaging. Whether it is used for gaming, parties, or events, this device adds excitement and energy to the room. The OLED Hat Project is a new step in combining tech and fun in a wearable format.

PRODUCT OPERATING INSTRUCTION

How to Use the Braincap

1. Power on the device using the switch on the side of the helmet.
2. The OLED display will show a welcome message.
3. One player wears the Braincap, while the other player enters a word and set the timer on the website.
4. The word appears only on the OLED screen, so the person wearing the helmet
cannot see it.
5. The second player acts out the word using gestures and movements.
6. The accelerometer tracks the movement and confirms the player is interacting.
7. The person wearing the Braincap guesses the word based on the other player's actions.
8. If correct, the system plays a victory sound and lights up the RGB LEDs.
9. If time runs out, the buzzer sounds, and the game ends.
10. To start a new round, press the reset button or re-enter a new word on the site.

Safety Instructions

· Do not drop the device, as the sensors inside are sensitive.
· Avoid exposing to water or extreme heat, which can damage the electronics.
· Charge the battery only with the provided charger to prevent damage.
· Do not shake the helmet aggressively, as it may affect the accelerometer.

MAINTAINANCE REQUIREMENTS

General Maintenance

· Keep the Braincap in a dry and cool place when not in use.
· Avoid dropping or exposing the device to strong impacts.
· Ensure the battery is charged regularly to prevent deep discharge.
· Store the Braincap safely to avoid accidental damage.

Troubleshooting Guide

· No Display on OLED

Cause: The device is powered off, low battery, or loose wiring.
Solution: Ensure the power switch is turned on. Charge the battery fully and restart the device. If the issue remains, check and secure the wiring connections inside the Braincap.

· Word Not Updating on Display

Cause: Connection issue between the website and the Braincap.
Solution: Refresh the website and enter the word again. If it still doesn't work, restart both the Braincap and your browser. Ensure the device is properly connected to the system.

· No Sound from Buzzer

If no sound is coming from the buzzer, it might be loose wiring or a problem with the speaker. Check the connections and restart the system. If it still doesn't work, test the buzzer separately to see if it is functional.

· 7-Segment Display Not Showing Timer

Cause: The timer is not activated or there is a temporary glitch. Solution: Restart the Braincap and check if the game mode is correctly selected. Ensure that the timer settings are enabled on the website.

FUTURE IMPROVEMENT

Limitations of Current Design

· The current version of Braincap is designed for a single game, which might reduce user interest over time.
· The display size is limited, which can make it difficult to show detailed information.
· The microphone works well, but in very noisy environments, voice recognition can be affected.
Suggested Upgrades
· To keep users engaged, we plan to introduce multiple games in the system, allowing players to choose from different challenges.
· Our long-term vision is to integrate 1,000+ games into a single helmet, making Braincap a versatile gaming device.
· A mobile app can be introduced to allow users to upload custom themes and to change the rules of the games.
· Future versions will include an option for multiplayer mode using wireless communication.

Better Hardware Options

- A larger and higher-resolution display can be used for better visibility.
- A more powerful processor will enhance game processing and allow smoother operation.
- A higher-capacity battery will be considered to extend gameplay duration without frequent recharging.

This is just a prototype, and more improvements will be introduced in the next phase of upgrades. Braincap is just the beginning of a new way to enjoy traditional games with technology. With planned improvements, it will offer endless fun with a wide variety of games, making it suitable for all ages!

CONCLUSION

In today's world, most people are focused on the latest gaming consoles like PS4, Xbox, and other high-tech devices. While these devices are great, they can sometimes make people feel isolated. The trend of playing solo or against virtual opponents can lead to less social interaction. We rarely see innovations that bring traditional games into the modern age while focusing on social interaction and physical involvement. The Braincap project is an attempt to bridge this gap and bring back the joy of traditional games, but with a new, interactive twist.

No one has ever really planned to innovate or modernize traditional games in a way that encourages interaction between players. People have been more focused on advancing virtual experiences, but these don't offer the same personal connections that traditional games once did. Braincap brings a refreshing change by making the game interactive, physically engaging, and reliant on teamwork. This is something that sets it apart from the solo gaming experiences that dominate the market today.

The world today is moving fast towards high-tech gadgets, but people still crave simple, fun experiences that bring them closer to others. The idea behind Braincap is to take traditional games, like charades, and add a smart, wearable, and interactive twist. It's about bringing people together, encouraging teamwork, and making social interaction an essential part of the experience. This project is a step towards creating a new form of gaming that's not just about high-end graphics and screens but also about having fun with others in real life.

What sets Braincap apart from other gaming devices is that it provides a fully interactive experience. The traditional games that people have enjoyed for generations are brought to life through technology. The system allows players to guess words based on actions and voice input, making it fun and engaging for everyone involved. Whether it's a family gathering, a party, or a small event, Braincap turns every moment into an exciting gaming experience.

This innovation is not just about providing a game; it's about reimagining how people can connect and engage with one another. In a society where digital interactions are becoming more common, Braincap shows how we can use technology to bring people together in a more meaningful way. The goal is to keep the essence of traditional games while incorporating technology to make them more exciting, engaging, and fun.

Looking forward, the future of Braincap holds many possibilities. As we continue to develop it, we aim to add more games, enhance its features, and make it even more interactive. The vision is to create a gaming system that combines the best aspects of traditional and modern games, offering something for everyone, no matter their age or background. With the integration of more games, better connectivity, and improved user experience, Braincap has the potential to be a game-changer in the world of interactive gaming.

In conclusion, Braincap is more than just a wearable device or a game; it's a new way of thinking about gaming. It's a fusion of tradition and technology that brings people closer, encourages communication, and ensures that fun is experienced together. It's not just about gaming anymore—it's about connecting, sharing experiences, and having fun as a group. In a world that's increasingly driven by solo gaming and virtual realities, Braincap offers something refreshing and exciting that we've all been missing. The future is bright, and we look forward to expanding Braincap into a more interactive, multiplayer experience that will entertain and bring people together in ways never seen before.

Challenges Faced:

Developing the Braincap system presented various technical, design, and operational challenges that required significant effort to overcome. These challenges were key learning points in the project and gave us valuable insights into the complexities of integrating hardware, software, and design in wearable technology. Below, we've outlined the major challenges faced during the development process:

1. Power Management Issues:

One of the biggest challenges was managing the power requirements for different components. The system needed both 3.3V and 5V power for different parts of the setup, including the microcontroller, sensors, and display. This posed a significant issue as it was important to ensure that power did not fluctuate or exceed the required levels, as this could lead to component damage or malfunctions. Finding the right power regulation circuits that would provide both power levels efficiently without overloading any part of the system was time-consuming. It also involved testing the system under various load conditions to make sure everything ran smoothly.

2. Voice-to-Text Conversion:

Another challenge we faced was with the microphone-based voice recognition system. The system was designed to convert the spoken words of the player into text, which would then be checked against the correct word. However, in noisy environments, the microphone had difficulty capturing clear voice input, leading to inaccurate text conversion. This required us to refine the speech recognition software and optimize it for better accuracy in various environments. We had to tweak the microphone's sensitivity and implement noise reduction algorithms to improve the system's performance in real-world conditions.

3. Accelerometer Calibration:

The accelerometer played a crucial role in detecting the tilt and movement of the helmet to ensure that the game only started when the helmet was properly worn. However, ensuring accurate motion detection was a challenge. The accelerometer required regular calibration to maintain precise orientation readings. Without this calibration, the system would register false motion detections, either causing the game to start prematurely or not start at all. We had to constantly recalibrate the sensor during the development phase and implement checks in the code to correct for any misreadings.

4. OLED Display Issues:

The OLED display was another component that posed difficulties. At times, the display would fail to show the intended text or images, causing confusion during testing. The screen would either flicker or fail to load certain content, making it hard to troubleshoot. This issue was particularly frustrating as the display was one of the most important output devices in the system, providing crucial game information. Debugging the display required us to adjust the code controlling the display and address compatibility issues between the microcontroller and the OLED screen. Multiple testing iterations were required before we could get the display working reliably.

5. Coding and Integration Challenges:

Integrating multiple hardware components, including sensors, displays, motors, and the microcontroller, into a seamless system was a major coding challenge. Ensuring smooth communication between all the parts required careful programming and frequent debugging. Different components communicated through serial communication, and the timing of their interactions had to be precise to avoid delays or failures. We encountered numerous software bugs and synchronization issues, which were especially challenging because of the large number of devices involved. Extensive testing, debugging, and optimization of the code were essential in overcoming these challenges.

6. Initial Design Modifications:

Our first version of the helmet design was functional, but it lacked the visual appeal we desired. Additionally, the design was not as comfortable as we had hoped. The helmet was bulky and heavy, making it less enjoyable to wear for extended periods. After feedback from early testers, we decided to make modifications to improve the design. We redesigned the helmet to make it lighter, more comfortable, and aesthetically pleasing. This required reworking the placement of components and ensuring that the design would still be functional while providing a better user experience.

7. Timing and Synchronization Problems:

Another issue we encountered was the synchronization of various components, particularly the vibration motors and OLED display. Initially, the vibration motors would not work in sync with the display or the gameplay, leading to delays in providing feedback to the player. The game experience was not as smooth as we intended, so we had to implement more complex coding to synchronize the components. This required fine-tuning the timing of the components and adjusting the interactions between sensors, outputs, and the game logic.

8. Game Logic Refinements:

While the basic game mechanics were functional, we had to adjust the rules and interactions to ensure smooth gameplay. The system needed to provide real-time feedback while also maintaining the traditional feel of the game. We had to ensure that players were not confused by any unexpected behavior of the system and that the transitions between words were seamless. Fine-tuning the game logic, including the flow of gameplay and the timing of actions, was a critical task to make the game enjoyable and engaging for players.

9. Battery Optimization:

Battery life was another area of concern. Given that the system was powered by a rechargeable battery, we had to make sure that the battery would last for an extended period without frequent recharging. This was especially challenging because of the power requirements of the sensors, displays, and motors. We implemented power-saving techniques to optimize battery usage, such as controlling when components were powered on or off based on the game state. The goal was to maximize playtime while keeping the system portable and lightweight.

10. Component Placement and Wiring:

Arranging all the components inside the helmet while maintaining comfort and usability was one of the most challenging design tasks. We had to ensure that the helmet was not too bulky or heavy and that all the wiring was neatly organized. The placement of components such as the sensors, battery, and wiring had to be done carefully to avoid interference or discomfort for the wearer. This required several iterations to find the right balance between functionality, comfort, and weight.

Project Success Story: A Proud Journey of Braincap

Working on the Braincap project has been an exciting and challenging journey for me. When I first started, I never imagined how much this project would teach me and how proud I would feel once it came together. Every step, from brainstorming ideas to testing the final product, was full of learning and hard work. As I saw the project progress, I realized how much potential Braincap had not just as a game, but as a mix of creativity and technology.

The idea for Braincap started simply — I wanted to create a wearable game system that would change how people play traditional games. Traditional games, like board games, have always been a good way to bring people together. But today, everyone is focused on PS4, Xbox, and other gaming consoles. I felt there was a need for a game that was more interactive and social. I wanted to create something that would bring people together, make them work as a team, and give them a more interactive experience. And that's how Braincap came into being.

At the beginning, I faced many challenges like managing power, microphone issues, sensor calibration, and design problems. But each challenge made me more determined to make the project work. I didn't want Braincap to be just another gadget; I wanted it to be something that would make a real impact. I wanted it to be a fun and simple game that anyone could enjoy.

One of the things I'm most proud of in this project is the accuracy and performance of the components. The accelerometer and microphone were very important parts of the system. The accelerometer detects the position of the helmet, making sure the game works only when the helmet is worn correctly. Getting this right took many attempts and careful calibration. The microphone also had to detect voice commands accurately, even in noisy environments. This was an important feature for the game to work properly, and I'm happy that it functions well, converting speech to text without much delay.

Another part that made me feel proud was the way all the components work together. The system needed to work in harmony with different devices like the OLED display, vibration motors, and sound feedback. I had to make sure that all these components communicated with each other and worked without any delays. When I saw the system run smoothly — with the correct word displayed, the right feedback given, and the game progressing — I knew the hard work had paid off.

The design and comfort of the helmet were also important. At first, the helmet was functional, but it was not very comfortable or attractive. After many redesigns, I was able to make the helmet lighter and more comfortable to wear. Now, users can enjoy the game without feeling uncomfortable. It was a proud moment when I saw users enjoying the game, not just because of the technology, but because of how comfortable and fun the experience was.

The most satisfying part of the project was seeing how much people enjoyed the game. Watching friends and family play, guessing the words, and laughing together showed me the real impact of what I had created. It wasn't just about the technology or the sensors; it was about how the game brought people together and made them interact in a fun way.

Throughout this journey, I also made many new friends who became not just collaborators but also good friends. The project helped me connect with many people, and their support played an important role in overcoming challenges. Whenever I got stuck or needed help, these new friends were always there to guide me, share their knowledge, and help me solve problems. Their support was invaluable. It was a reminder of how important collaboration is and how working together makes everything easier and more enjoyable.

When I think about how Braincap combines technology with fun, I feel proud of what I have achieved. It's not just another tech project; it's a game that encourages teamwork, improves guessing skills, and helps people bond over simple interactions. Unlike the high-tech gaming consoles we see today, Braincap brings the fun back to traditional games in a new and exciting way. It's a reminder that technology doesn't always need to be complicated; it can be simple, fun, and interactive.

Looking back on this journey, I feel a sense of accomplishment. The process wasn't always easy, but the satisfaction of seeing everything come together was priceless. This project allowed me to apply all my technical skills while being creative. I was able to use simple sensors and smart design to create something that was both entertaining and useful.

Braincap is more than just a product to me; it's a symbol of what can be achieved when creativity and technology come together. It's a reminder that even with all the advancements in gaming, there is still room for ideas that bring people together. This project has shown me that wearable tech can be more than just gadgets; it can create real experiences that improve how we connect with others.

As I think about the future, I'm excited about where Braincap can go. There are so many opportunities to improve the game, add new features, and create even more ways for people to enjoy it. I hope to continue developing and expanding Braincap, making it better and adding more games to the system. The success of this project is just the beginning, and I look forward to seeing where this journey takes me in the future.

Final Thoughts

Braincap proves that integrating technology into traditional games can enhance engagement and fun. It brings a modern twist to a well-known guessing game, making it more interactive and exciting for people of all ages. The use of sensors, voice input, and display technology makes the game feel more advanced while still keeping the simple and enjoyable nature of the original game.

This is just the first version, and there is a lot of room for improvement. In future updates, we plan to introduce more games so that users do not get bored of playing the same one repeatedly. The goal is to have over 1000 different games available in a single helmet, giving people more variety and fun options to choose from.

We also aim to improve the accuracy of the system by using better sensors and optimized software, which will help in making the experience smoother and more reliable. Another important upgrade will be better hardware components, ensuring that the device runs faster, lasts longer on battery, and is more durable. We also plan to make the design lighter and more comfortable so that users can wear it for longer periods without any discomfort. Overall, this project showcases how simple innovations can bring new life to traditional games. It proves that by using technology, we can make games more enjoyable and interactive for a wider audience. Braincap has the potential to be a popular and fun group activity, and with further improvements, it could become an entirely new way to experience games.

REFERENCES

APA Format

- OpenAI. (2024). ChatGPT: AI-Powered Assistance for Developers. OpenAI Research.
- Developer Name. (2023). Charades: A Classic Guessing Game. Game Studios.
- Developer Name. (2023). HeadsGame: Interactive Gaming on Mobile Devices. Mobile Phones Inc.
- Amazon. (2024). Electronic Components for DIY Projects. Amazon.com.

MLA Format

- OpenAI. ChatGPT: AI-Powered Assistance for Developers. OpenAI Research, 2024.
- Developer Name. Charades: A Classic Guessing Game. Game Studios, 2023.
- Developer Name. HeadsGame: Interactive Gaming on Mobile Devices. Mobile Phones Inc., 2023.
- Amazon. Electronic Components for DIY Projects. Amazon.com, 2024.

APPENDICES

Appendix A – System Block Diagram

A visual representation showing the overall structure of the Braincap system, including the ESP32 controller, sensors (microphone, accelerometer, ultrasonic sensor), output devices (OLED display, buzzer, LEDs), and power supply.

Appendix B – Electrical Diagrams

Detailed circuit connections and wiring of all components, including:

Electrical Diagram 1: Sensor and input connections

Electrical Diagram 2: Output devices and power circuit layout

Appendix C – PCB Design Diagrams

Custom PCB layouts designed for compact integration of Braincap hardware. Includes:

PCB Design 1: Top layer view

PCB Design 2: Component placement view

Appendix D – Mechanical Diagrams

Design sketches and dimensions used in building the Braincap helmet. Includes mounting positions for sensors, displays, and wiring.

Appendix E – 3D Helmet Design

Images or screenshots of the 3D model of the helmet prototype used in planning and assembly. Designed for fit, comfort, and ease of component placement.

Appendix F – IoT Dashboard Interface

Screenshots and brief descriptions of the web-based dashboard used to:

Add new words to the game

Set the timer

Start or reset the game

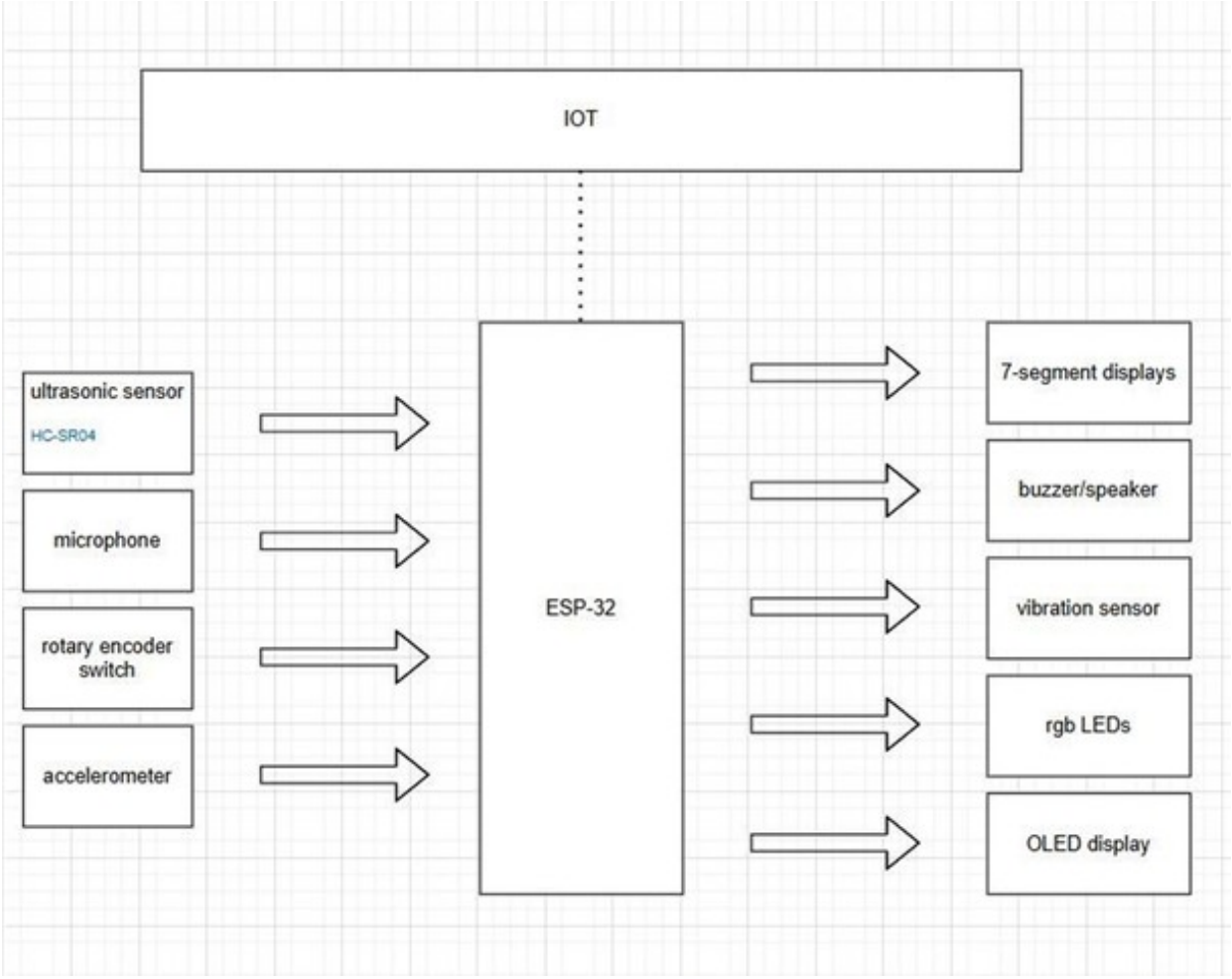
Display WebSocket communication in real time

Appendix G – Coding Flowchart

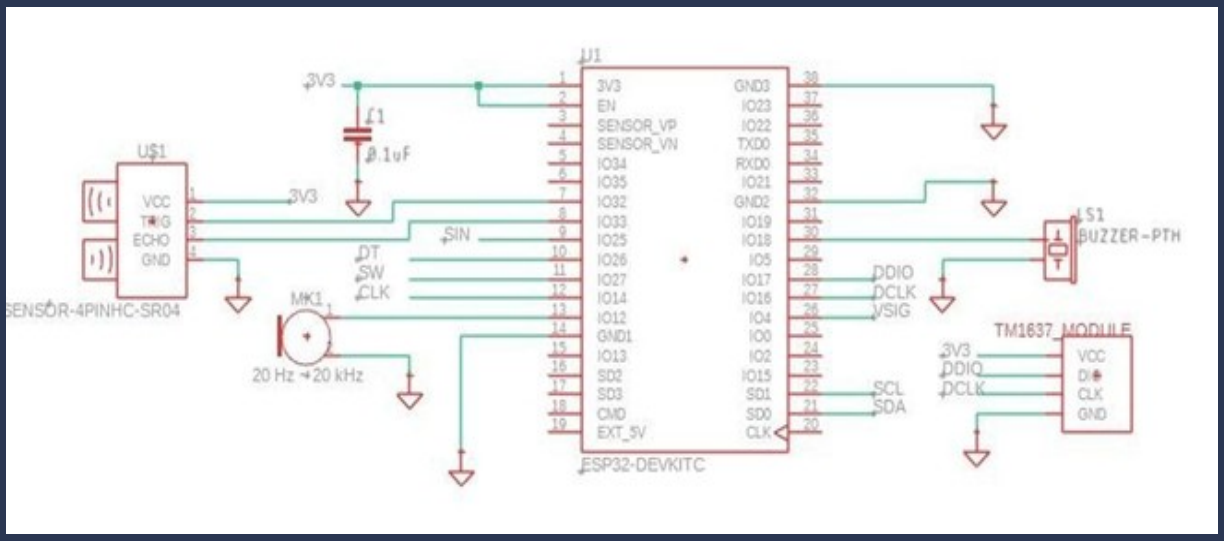
A flowchart showing the main logic behind the Braincap system’s software. It includes input handling, decision-making, and output triggers for different components.

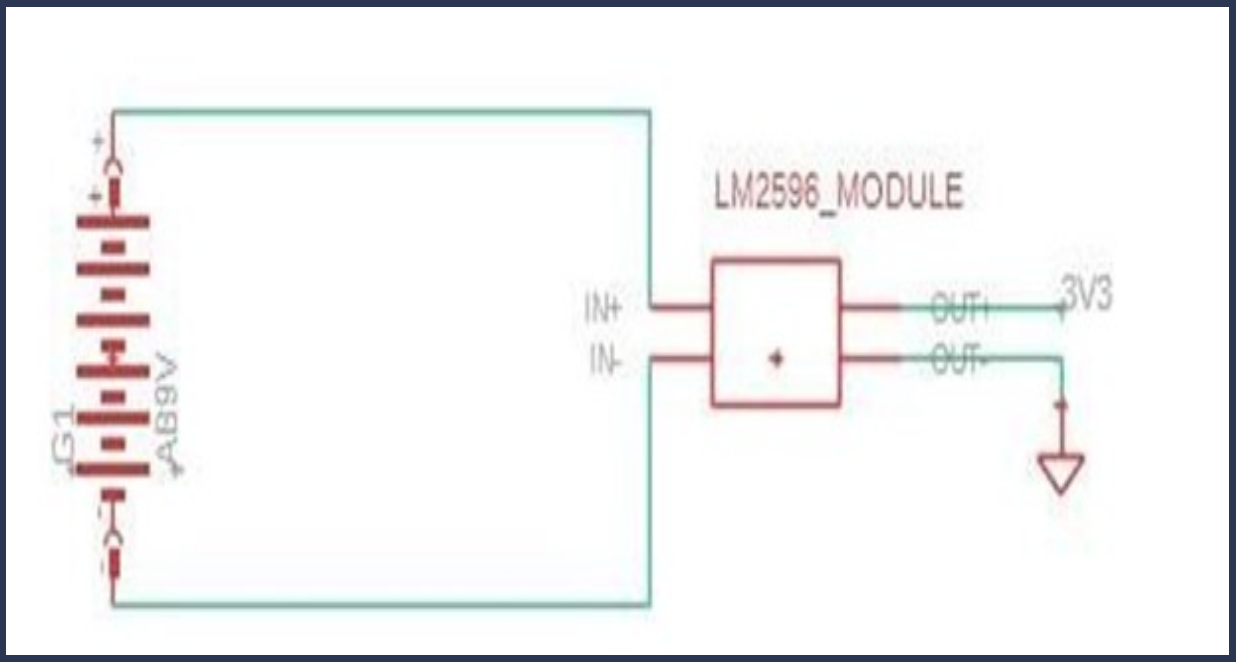
Appendix H – Bill of Materials (BOM)

System Block Diagram

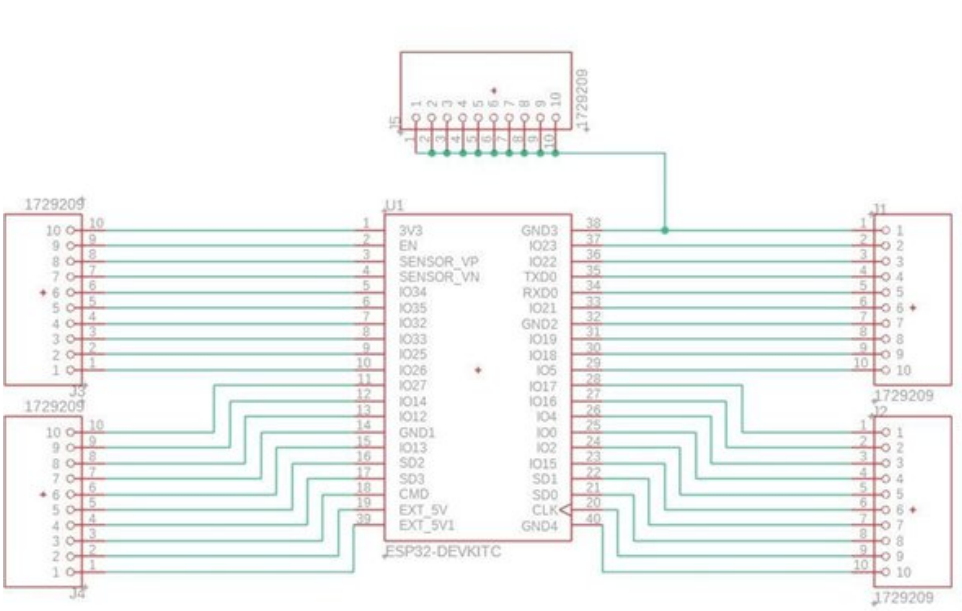

IOT
ultrasonic sensor
HC-SR04
microphone
rotary encoder switch
accelerometer
ESP-32
7-segment displays
buzzer/speaker
vibration sensor
rgb LEDs
OLED display

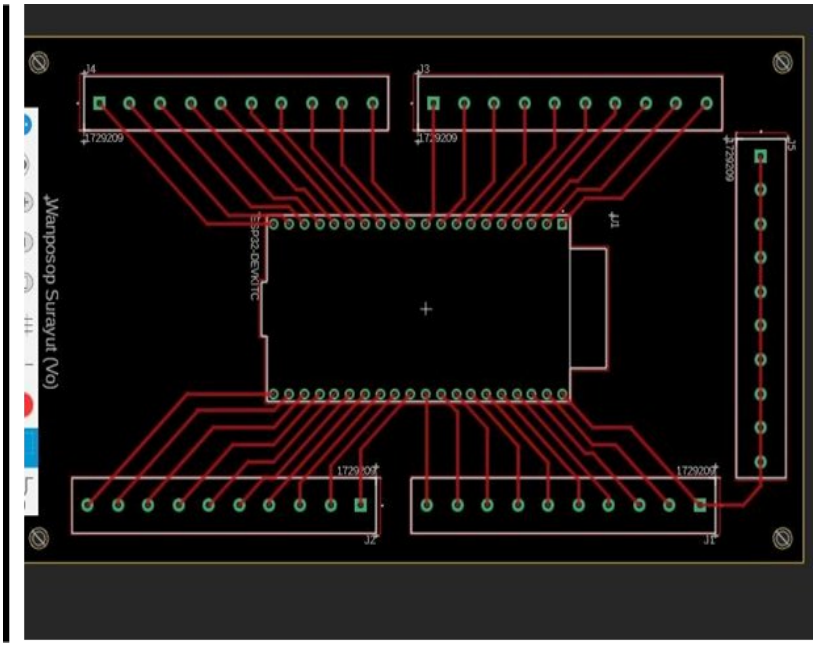
Electrical Diagram





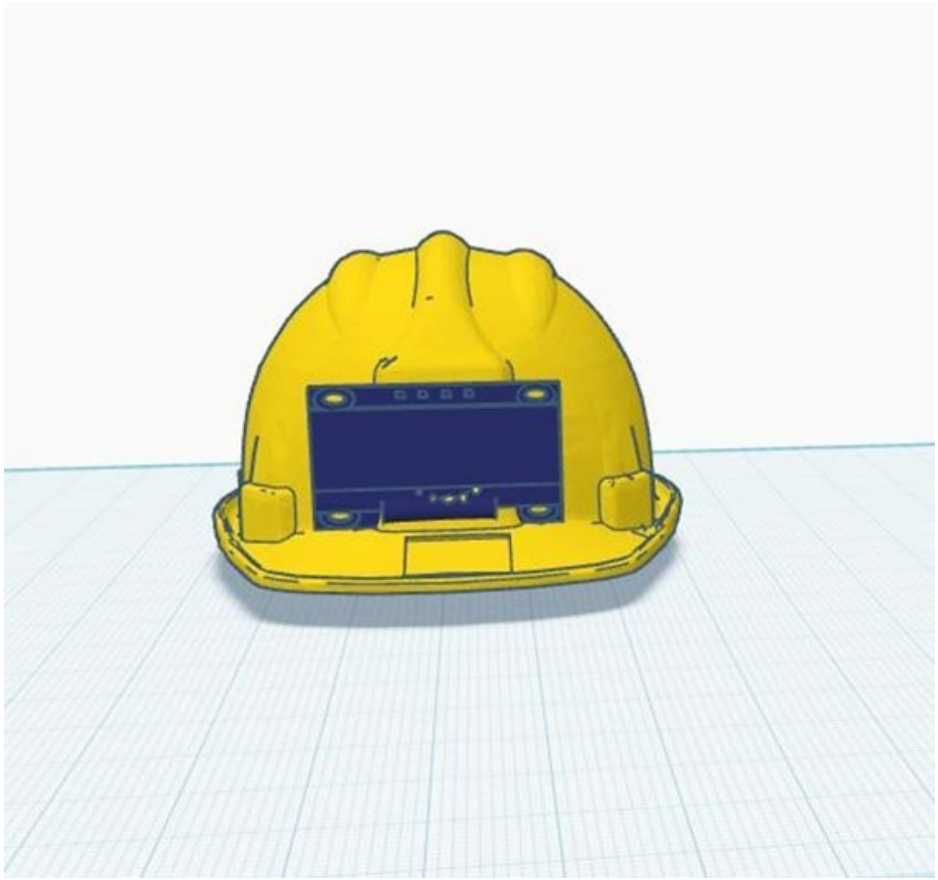
PCB design Diagrams

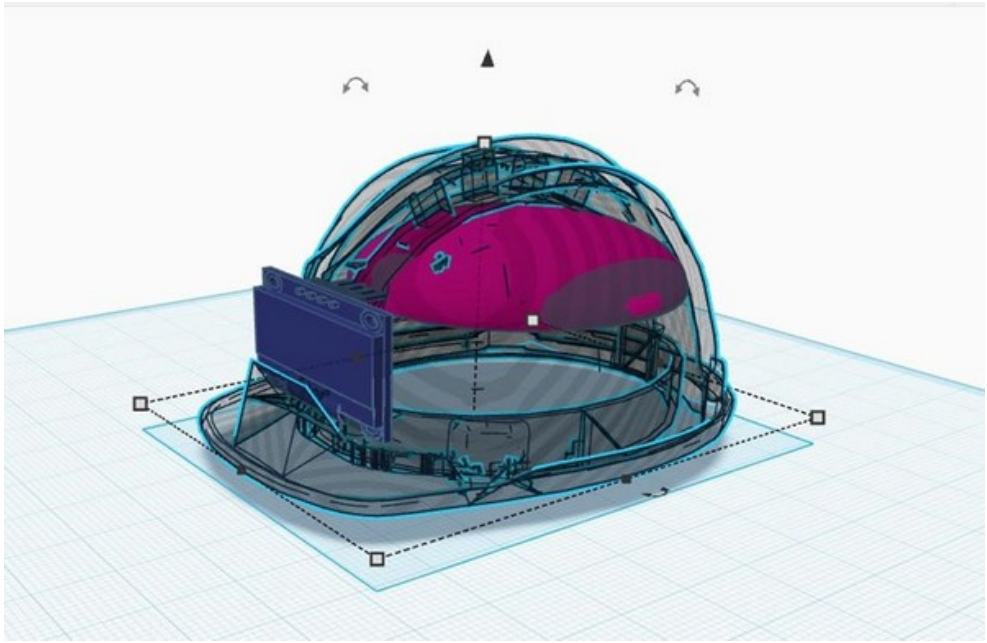




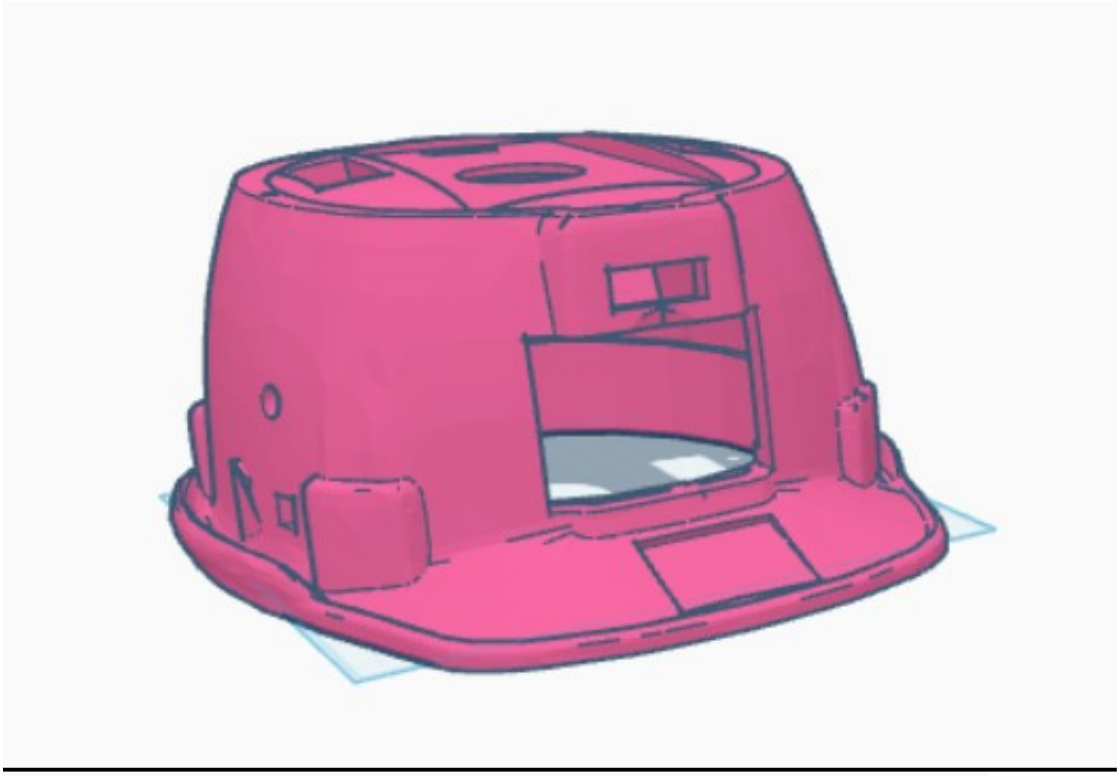
Mechanical diagrams

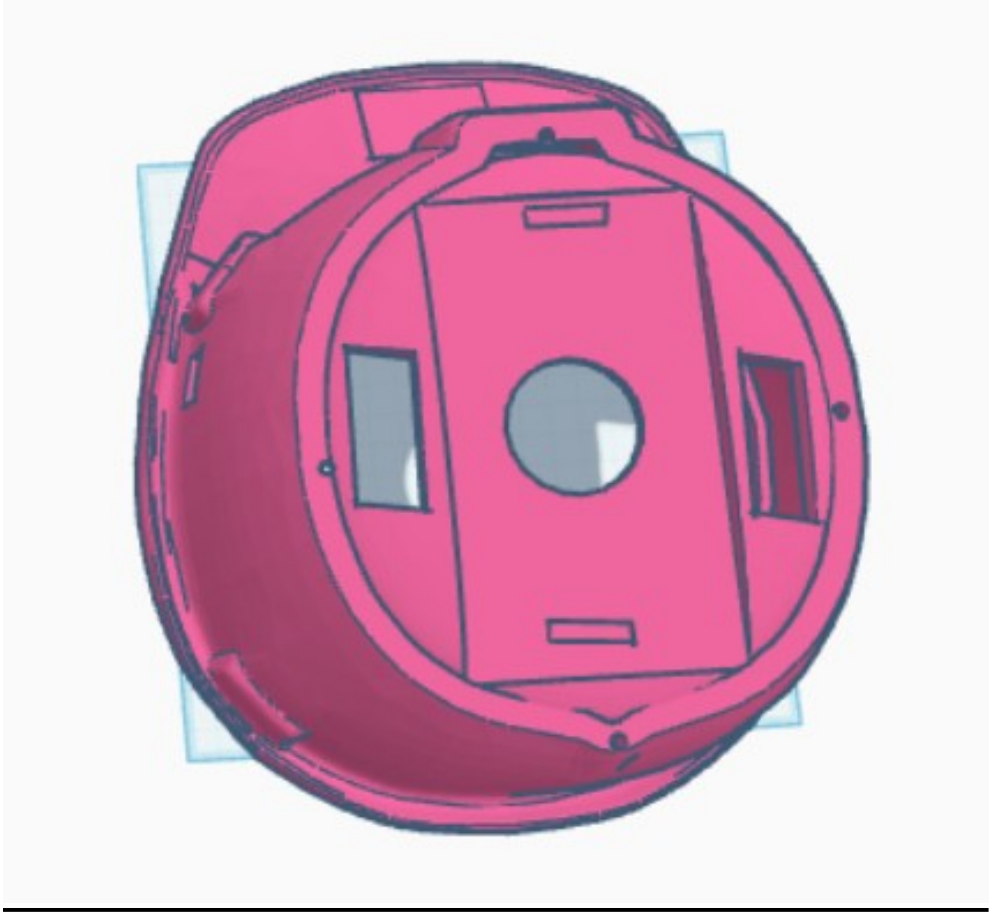
Initial plans and 3-d Diagrams

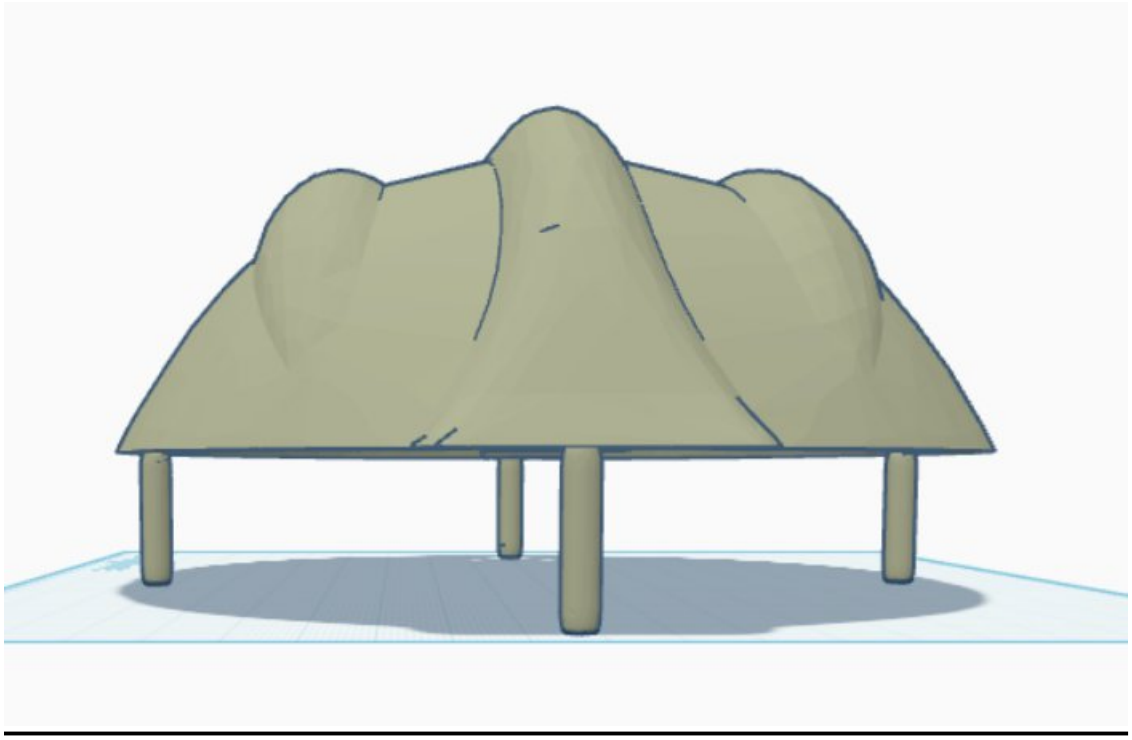




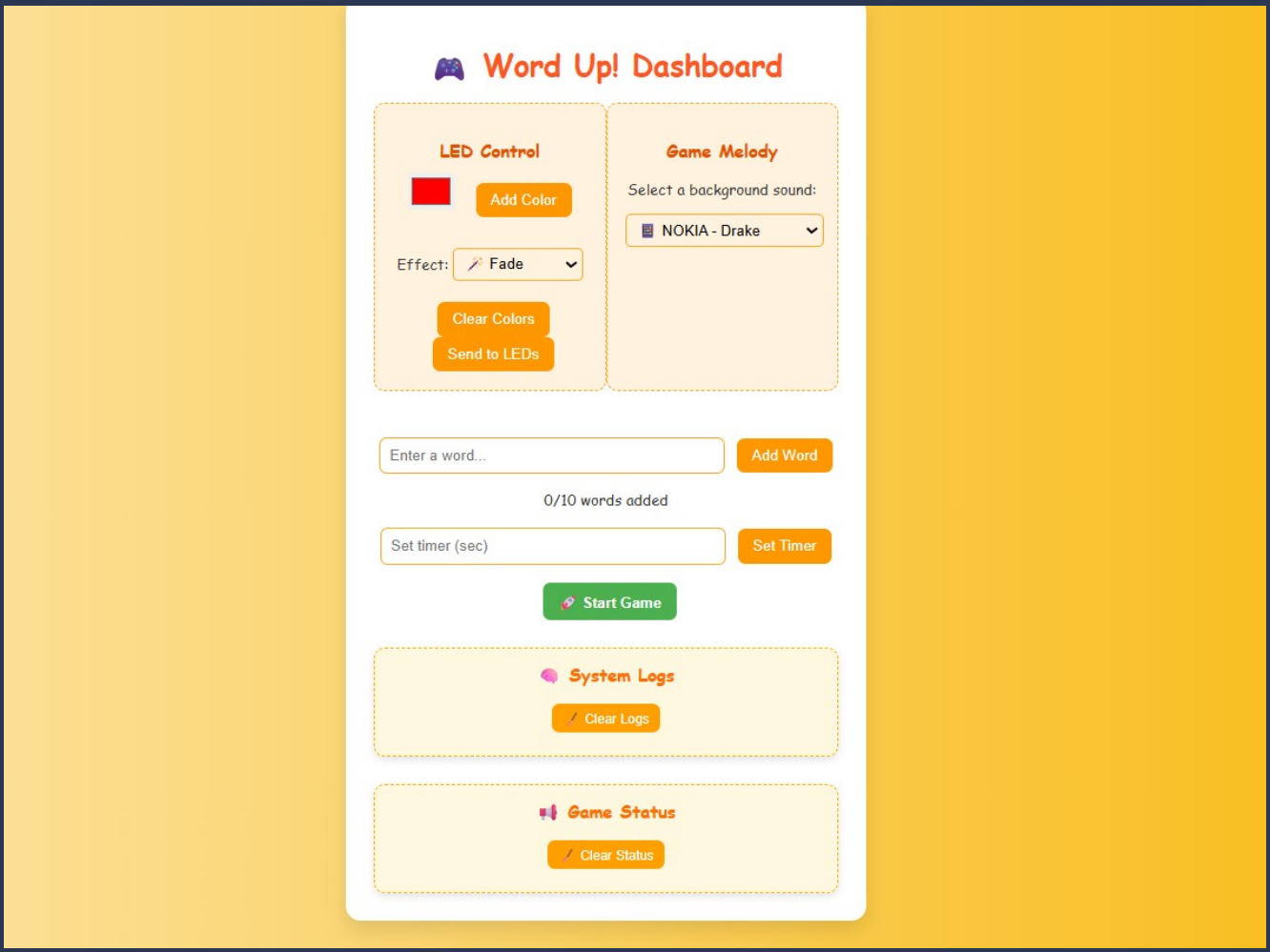
Final #-D Design







IOT dashboard



Bill of materials

	Description		part name	real price	quantity	total
uxcell	Microphone		Electret Microphone;Diameter : 6mm / 0.24""	7$	1	7$
	ultrasonic sensor		HC-SR04	4.5$	1	4.5$
ZUDKSUY	rotary encoder switch		EK2045x4C Rotary Encoder Switch	4$	1	4$
DWEII	OLED display		SSD1322	33$	1	33$
DUTTY	Vibration sensors		SW-420	2.2$	2	2.2$
Freenove	ESP-32		FNK0090B-ESP32-WI	16$	1	16$
INIU	POWER BANK		INIU Power Bank, USB C in&Out Slimmest 10000mAh Portable Charger	25$	1	25$
Soberto	Speakers		SPEAKER 8OHM 1W TOP	3$	2	6$
Broadcom	7-segment display		HDSP-521A	2$	1	2$
RZHBYS	Accelerometer		MPU6500 6-Axis Gyroscope Accelerometer Sensor Module	13$	1	13$
	Helmet			7$		7$
Lxyoug	RGB lights		zx-5050-65-ZXMEA N Led Lights	17$	1	17$
TDK	Buzzer		PS1240P02BT-BUZZER PI	1$	1	1$
					Total	137$

Contact Informations

Name: Wanposop Surayut (Vo)

Phone: 4375456566

Email- Swanposop@myseneca.ca

Coding flowchart

HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
  <title>Word Up Game Dashboard</title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <div class="container">
    <h1>🎮 Word Up! Dashboard</h1>

    <div class="input-section">
      <input type="text" id="wordInput" placeholder="Enter a word..." maxlength="20"/>
      <button onclick="addWord()">Add Word</button>
      <p id="wordCount">0/10 words added</p>
```

```
    </div>

    <ul id="wordList" class="word-list"></ul>

    <div class="input-section">
      <input type="number" id="timerInput" placeholder="Set timer (sec)" min="1" max="999"/>
      <button onclick="setTimer()">Set Timer</button>
    </div>

    <div class="input-section">
      <button class="start-btn" onclick="startGame()">🚀 Start Game</button>
    </div>

    <div id="statusArea"></div>
  </div>

  <script>
    const socket = new WebSocket(`ws://${location.hostname}/ws`);
    let wordCounter = 0;
```

```
  let wordList = [];

  socket.onopen = () => {
    console.log("WebSocket connected");
  };

  socket.onmessage = (event) => {
    const msg = event.data;
    const area = document.getElementById("statusArea");
    area.innerHTML += `<p>📥 ${msg}</p>`;
  };

  function addWord() {
    const input = document.getElementById("wordInput");
    const word = input.value.trim();
    if (word && wordCounter < 10
&& !wordList.includes(word)) {
      socket.send(JSON.stringify({ word: word }));
      wordList.push(word);
      wordCounter++;
      document.getElementById("wordCount").innerText =
`${wordCounter}/10 words added`;
```

```
      input.value = "";
      updateWordList();
    } else if (wordCounter >= 10) {
      alert("Maximum 10 words reached!");
    }
  }

  function removeWord(word) {
    const index = wordList.indexOf(word);
    if (index > -1) {
      wordList.splice(index, 1);
      wordCounter--;
      socket.send(JSON.stringify({ remove: word }));
      document.getElementById("wordCount").innerText =
`${wordCounter}/10 words added`;
      updateWordList();
    }
  }

  function updateWordList() {
    const list = document.getElementById("wordList");
    list.innerHTML = "";
```

```
    wordList.forEach((word, index) => {
      const li = document.createElement("li");
      li.innerHTML = `${index + 1}. ${word} <button
class="remove-btn"
onclick="removeWord('${word}')">✖</button>`;
      list.appendChild(li);
    });
  }

  function setTimer() {
    const input = document.getElementById("timerInput");
    const time = parseInt(input.value);
    if (!isNaN(time) && time > 0) {
      socket.send(JSON.stringify({ timer: time }));
      document.getElementById("statusArea").innerHTML +=
`<p>⏱ Timer set to ${time} seconds</p>`;
      input.value = "";
    }
  }

  function startGame() {
    if (wordCounter === 0) {
```

```
      alert("Add at least 1 word before starting!");
      return;
    }
    socket.send(JSON.stringify({ start: true }));
    document.getElementById("statusArea").innerHTML +=
`<p>□ Game started!</p>`;
  }
 </script>
</body>
</html>
```

CSS

```
body {
  font-family: 'Comic Sans MS', cursive, sans-serif;
  background: linear-gradient(to right, #fceabb, #f8b500);
  color: #333;
  margin: 0;
  padding: 0;
}

.container {
  max-width: 500px;
  margin: 50px auto;
  padding: 30px;
  background: white;
  border-radius: 16px;
  box-shadow: 0 10px 20px rgba(0,0,0,0.1);
  text-align: center;
}

h1 {
  font-size: 2rem;
```

```
  margin-bottom: 20px;
  color: #ff5722;
}

.input-section {
  margin: 20px 0;
}

input[type="text"],
input[type="number"] {
  padding: 10px;
  width: 70%;
  font-size: 1rem;
  border: 2px solid #ffa726;
  border-radius: 8px;
}

button {
  padding: 10px 16px;
  margin-left: 8px;
  font-size: 1rem;
  border: none;
```

```
  background-color: #ff9800;
  color: white;
  border-radius: 8px;
  cursor: pointer;
  transition: background 0.3s;
}

button:hover {
  background-color: #fb8c00;
}

.start-btn {
  background-color: #4caf50;
  font-weight: bold;
}

.start-btn:hover {
  background-color: #388e3c;
}

#statusArea {
  margin-top: 20px;
```

```
  font-size: 1rem;
  background: #fffde7;
  padding: 10px;
  border-radius: 8px;
  border: 1px dashed #fbc02d;
}

.word-list {
  list-style-type: none;
  padding: 0;
  margin: 10px auto;
  text-align: left;
  max-width: 300px;
  font-size: 1.1rem;
}

.word-list li {
  background: #ffecb3;
  margin: 5px 0;
  padding: 8px 12px;
  border-radius: 8px;
  border-left: 5px solid #f57c00;
```

```
  display: flex;
  justify-content: space-between;
  align-items: center;
}

.remove-btn {
  background: transparent;
  border: none;
  font-size: 1.2rem;
  cursor: pointer;
  color: red;
}
```

Main Esp-32 Code

```
#include <Arduino.h>
#include <Wire.h>                    // ✈ I2C communication for MPU6050
// Enables communication between ESP32 and the MPU6050 using I2C protocol

#include <TM1637Display.h>           // 🕒 4-digit 7-segment display
// Used to display numbers (like a timer) on a 4-digit LED display

#include <NewPing.h>                 // 📡 HC-SR04 ultrasonic distance sensor
// Helps detect if helmet is on by measuring distance in front of the sensor

#include <ESPAsyncWebServer.h>       // 🌐 Async web server
// Lets us serve a dashboard and communicate in real-time via WebSocket

#include <AsyncTCP.h>                // ✈ Required for AsyncWebServer
// Backend support for Async WebSocket and HTTP on ESP32

#include <WiFi.h>                    // 📡 WiFi support
```

```
// Lets ESP32 connect to your WiFi so it can talk to the dashboard

#include <SPIFFS.h>                    // 📁 SPI Flash for HTML dashboard
// Stores your HTML/JS/CSS dashboard files directly on the ESP32

#include <RotaryEncoder.h>             // 🔄 Rotary encoder input
// For turning/skipping through words manually via knob

#include <MPU6050.h>                   // 🧭 Accelerometer/gyroscope sensor
// Reads motion like nodding to auto-skip words

#include <Adafruit_NeoPixel.h>         // ✨ NeoPixel LED library
// Controls the LED strip for colorful feedback and alerts

// ---------------------------
// 🌐 WiFi credentials
// ---------------------------
// Used to connect the ESP32 to the specified WiFi network

const char* ssid = "PI34";
const char* password = "vo123456";

// ---------------------------
// □ Pin assignments
// ---------------------------
// Mapping all hardware parts to specific ESP32 GPIO pins

#define ROTARY_CLK 26
#define ROTARY_DT 27
#define ROTARY_SW 33
#define MIC_PIN 34
#define BUZZER_PIN 25
#define TRIG_PIN 5
#define ECHO_PIN 14
#define TM1637_CLK 18
#define TM1637_DIO 19
#define I2C_SDA 21
#define I2C_SCL 22
#define LED_PIN 4
#define NUM_LEDS 16

// ---------------------------
// 🔧 Hardware objects
// ---------------------------
// Initializing hardware libraries and creating component instances

TM1637Display display(TM1637_CLK, TM1637_DIO);
NewPing sonar(TRIG_PIN, ECHO_PIN, 400);
```

```
RotaryEncoder encoder(ROTARY_CLK, ROTARY_DT);
AsyncWebServer server(80);
AsyncWebSocket ws("/ws");
MPU6050 mpu;
Adafruit_NeoPixel leds(NUM_LEDS, LED_PIN, NEO_GRB + NEO_KHZ800);

// ----------------------------
// 🎮 Game state variables
// ----------------------------
// Keeps track of game logic: timer, word list, positions, helmet state, etc.

unsigned long timerDuration = 0;
unsigned long startTime = 0;
bool timerRunning = false;
bool helmetOn = false;
bool buzzerTriggered = false;
String wordList[10];
int wordCount = 0;
int currentWordIndex = -1;
long lastPosition = 0;
unsigned long lastNodTime = 0;
bool gameStarted = false;

// ----------------------------
// ✨ LED helper functions
// ----------------------------
// Utility functions to control LED strip behavior

void setAllLEDs(uint32_t color) {
  for (int i = 0; i < NUM_LEDS; i++) {
    leds.setPixelColor(i, color);
  }
  leds.show();
}

void updateLEDColorPerWord() {
  // Change color depending on word index, like a visual indicator of progress
  uint8_t hue = (currentWordIndex * 40) % 255;
  for (int i = 0; i < NUM_LEDS; i++) {
    leds.setPixelColor(i, leds.ColorHSV(hue * 256));
  }
  leds.show();
}

void countdownLEDAlert(int remaining) {
  // Show color based on how close timer is to ending
  if (remaining <= 3) setAllLEDs(leds.Color(255, 0, 0));       // 🔴 Danger
  else if (remaining <= 6) setAllLEDs(leds.Color(255, 165, 0)); // □ Caution
```

```
  else setAllLEDs(leds.Color(255, 255, 0));                // □ Safe
}

// ---------------------------
// 🛜 WebSocket handler
// ---------------------------
// Handles messages from the dashboard (add word, start game, set timer)

void onWebSocketEvent(AsyncWebSocket *server, AsyncWebSocketClient *client,
                      AwsEventType type, void *arg, uint8_t *data, size_t len) {
  if (type == WS_EVT_DATA) {
    AwsFrameInfo *info = (AwsFrameInfo*)arg;
    if (info->final && info->index == 0 && info->len == len) {
      data[len] = 0;
      String msg = (char*)data;
      Serial.print("Received: ");
      Serial.println(msg);

      if (msg.startsWith("{")) {
        // ⏲ Set timer from dashboard
        if (msg.indexOf("timer") > 0) {
          int valIndex = msg.indexOf(":") + 1;
          timerDuration = msg.substring(valIndex).toInt();
          display.showNumberDec(timerDuration, true);
          startTime = millis();
          timerRunning = true;
          buzzerTriggered = false;
        }

        // 📝 Add word to list
        if (msg.indexOf("word") > 0) {
          int valIndex = msg.indexOf(":") + 2;
          int endQuote = msg.lastIndexOf("\"");
          if (wordCount < 10) {
            wordList[wordCount] = msg.substring(valIndex, endQuote);
            wordCount++;
          }
        }

        // ✖ Remove word from list
        if (msg.indexOf("remove") > 0) {
          int valIndex = msg.indexOf(":") + 2;
          int endQuote = msg.lastIndexOf("\"");
          String toRemove = msg.substring(valIndex, endQuote);
          for (int i = 0; i < wordCount; i++) {
            if (wordList[i] == toRemove) {
              for (int j = i; j < wordCount - 1; j++) {
                wordList[j] = wordList[j + 1];
```

```
          }
          wordCount--;
          break;
        }
      }
    }

    // ▶ Start the game
    if (msg.indexOf("start") > 0) {
      if (wordCount > 0) {
        gameStarted = true;
        currentWordIndex = 0;
        ws.textAll("{\"word\":\"" + wordList[currentWordIndex] + "\"}");
        updateLEDColorPerWord();
      }
    }
  }
 }
}

// ---------------------------
// 🛠 Setup
// ---------------------------
// Initialization logic: setup sensors, screen, WiFi, filesystem, etc.

void setup() {
 Serial.begin(115200);
 encoder.setPosition(0);
 pinMode(BUZZER_PIN, OUTPUT);
 pinMode(MIC_PIN, INPUT);
 digitalWrite(BUZZER_PIN, LOW);
 display.setBrightness(7);
 display.showNumberDec(0, true);

 leds.begin();
 leds.clear();
 leds.show();

 WiFi.begin(ssid, password);
 Serial.print("Connecting to WiFi");
 while (WiFi.status() != WL_CONNECTED) {
  delay(500); Serial.print(".");
 }
 Serial.println("\nConnected. IP: " + WiFi.localIP().toString());

 if (!SPIFFS.begin(true)) {
  Serial.println("SPIFFS mount failed!");
```

```
    return;
  }

  server.serveStatic("/", SPIFFS, "/").setDefaultFile("index.html");
  ws.onEvent(onWebSocketEvent);
  server.addHandler(&ws);
  server.begin();

  Wire.begin(I2C_SDA, I2C_SCL);
  mpu.initialize();
  Serial.println(mpu.testConnection() ? "MPU6050 ready." : "MPU6050 FAILED.");
}

// ----------------------------
// 🔁 Main loop
// ----------------------------
// Continuous checks for helmet, timer, encoder, nods, mic input

void loop() {
  ws.cleanupClients();

  // □ Helmet detection
  // Triggers when helmet is worn (<=3cm) or removed (>5cm)
  int distance = sonar.ping_cm();
  if (distance > 0 && distance <= 3 && !helmetOn) {
    helmetOn = true;
    Serial.println("Helmet detected (<= 3cm)");
    ws.textAll("{\"helmet\":\"on\"}");
  } else if (distance > 5 && helmetOn) {
    helmetOn = false;
    Serial.println("Helmet removed (> 5cm)");
    ws.textAll("{\"helmet\":\"off\"}");
  }

  // ⏲ Timer logic
  // Countdown + buzzer and LED warning in last 10 seconds
  if (timerRunning) {
    unsigned long elapsed = (millis() - startTime) / 1000;
    int remaining = timerDuration - elapsed;

    if (remaining > 0) {
      display.showNumberDec(remaining, true);
      if (remaining <= 10 && !buzzerTriggered) {
        countdownLEDAlert(remaining);
        tone(BUZZER_PIN, 1000); delay(200);
        noTone(BUZZER_PIN); delay(800);
      }
    } else {
```

```
    timerRunning = false;
    display.showNumberDec(0, true);
    ws.textAll("{\"word\":\"Game Over\"}");
    setAllLEDs(leds.Color(255, 0, 0));
    Serial.println("Timer done!");
    if (!buzzerTriggered) {
      tone(BUZZER_PIN, 1000); delay(5000);
      noTone(BUZZER_PIN);
      buzzerTriggered = true;
    }
  }
}

// 🔄 Rotary encoder
// Manual word skip using the knob
encoder.tick();
long newPosition = encoder.getPosition();
if (gameStarted && newPosition != lastPosition && wordCount > 0) {
  currentWordIndex++;
  if (currentWordIndex >= wordCount) {
    ws.textAll("{\"word\":\"Game Over\"}");
    currentWordIndex = -1;
  } else {
    ws.textAll("{\"word\":\"" + wordList[currentWordIndex] + "\"}");
    updateLEDColorPerWord();
  }
  lastPosition = newPosition;
  delay(500);
}

// 🤖 Nod detection
// Auto-skip word if player nods (using z-axis acceleration)
if (gameStarted && millis() - lastNodTime > 1000) {
  int16_t ax, ay, az;
  mpu.getAcceleration(&ax, &ay, &az);
  if (az < -15000 || az > 15000) {
    Serial.println("Nod detected!");
    currentWordIndex++;
    if (currentWordIndex >= wordCount) {
      currentWordIndex = 0;
      ws.textAll("{\"word\":\"Game Over\"}");
    } else {
      ws.textAll("{\"word\":\"" + wordList[currentWordIndex] + "\"}");
      updateLEDColorPerWord();
    }
    lastNodTime = millis();
  }
}
```

```
// 🎤 Microphone (volume-based spike detection)
// Loud voice detection to trigger skip — say "Correct!" or clap
int micLevel = analogRead(MIC_PIN);
if (gameStarted && micLevel > 2000 && wordCount > 0 && helmetOn && timerRunning) {
  Serial.println("Loud voice detected - skip");
  currentWordIndex++;
  if (currentWordIndex >= wordCount) {
    currentWordIndex = 0;
    ws.textAll("{\"word\":\"Game Over\"}");
  } else {
    ws.textAll("{\"word\":\"" + wordList[currentWordIndex] + "\"}");
    updateLEDColorPerWord();
  }
  delay(500);
 }
}
```

Code for The Oled Display

```
/*****************************************************************
****

   Modified ESP32 OLED Code for Helmet Project

   - Uses ESP32-2432S028R ILI9341 Display

   - Connects to Main ESP32 via WebSocket

   - Displays word updates and helmet status

   - Beautified screen with centered text

*****************************************************************
***/


#include <Arduino.h>

#include <WiFi.h>

#include <WebSocketsClient.h>
```

```
#include <TFT_eSPI.h>
#include <ArduinoJson.h>

// WiFi Credentials (phone hotspot)
const char* ssid = "PI34";
const char* password = "vo123456";

// IP of the main ESP32 WebSocket server
const char* serverIP = "192.168.26.167"; // Set to your main ESP32's IP
const int serverPort = 80;

TFT_eSPI tft = TFT_eSPI();
WebSocketsClient webSocket;

String currentWord = "";
String helmetStatus = "";

void drawCenteredText(const String& text, int y, uint8_t size, uint16_t color) {
  tft.setTextSize(size);
  int16_t x = (tft.width() - (text.length() * 6 * size)) / 2;
  tft.setCursor(x, y);
  tft.setTextColor(color, TFT_BLACK);
  tft.print(text);
}

void webSocketEvent(WStype_t type, uint8_t *payload, size_t length) {
```

```
  if (type == WStype_TEXT) {
    Serial.printf("Received: %s\n", payload);

    StaticJsonDocument<200> doc;
    DeserializationError error = deserializeJson(doc, payload);

    if (!error) {
      if (doc["word"]) {
        currentWord = doc["word"].as<String>();
      } else if (doc["helmet"]) {
        helmetStatus = doc["helmet"].as<String>() == "on" ? "Helmet On!" :
"Helmet Off!";
      } else if (doc["command"]) {
        String command = doc["command"].as<String>();
        if (command == "skip") {
          currentWord = "Skipped!";
        }
      }
    } else {
      currentWord = (char*)payload; // fallback plain text
    }

    tft.fillScreen(TFT_NAVY);
    drawCenteredText("Word:", 30, 2, TFT_YELLOW);
    drawCenteredText(currentWord, 70, 3, TFT_WHITE);

    if (helmetStatus != "") {
```

```
      drawCenteredText(helmetStatus, 140, 2, TFT_GREEN);
    }
  }
}

void setup() {
  Serial.begin(115200);
  WiFi.begin(ssid, password);
  Serial.println("Connecting to WiFi...");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("\nConnected to WiFi");
  Serial.print("IP Address: ");
  Serial.println(WiFi.localIP());

  tft.init();
  tft.setRotation(1);
  tft.fillScreen(TFT_NAVY);
  drawCenteredText("Waiting for word...", 100, 2, TFT_WHITE);

  webSocket.begin(serverIP, serverPort, "/ws");
  webSocket.onEvent(webSocketEvent);
  webSocket.setReconnectInterval(5000);
}
```

```
void loop() {
  webSocket.loop();
}
```

esp32 screen TFT

Project Final Product