Project: Hide and Beep

Student id: UP2213439

0. Introduction to the Project

My initial idea for this project was to develop an application for the visually impaired. The idea was to provide a more precise mapping application. The challenge with existing mapping applications is that they inform you that you have arrived at your destination, but as a visually impaired person you still would struggle to find the entrance of your destination. The application would then provide more precise location information by using sound to guide the user to the entrance of your destination. However, upon review with my lecturers they identified some ethical issues. These being that the application would also need to consider hazardous objects, such as moving objects (cars, bikes) and solid objects in the way (such as gates, barricades, etc). Due to this I agreed to keep a similar concept but apply it to a game.

Therefore, this project involved the design and development of Hide and Beep, a mobile-first, browser-based multiplayer game inspired by the traditional game of hide-and-seek. The aim of the project was to explore how real-world data, particularly GPS location, could be combined with audio feedback to create an engaging outdoor game experience. Unlike traditional video games that rely on visuals alone, Hide and Beep focuses heavily on sound and physical movement, encouraging players to explore their surroundings.

In the game, one player is assigned the role of the hunter, while the remaining players become hiders. After an initial hiding period, the hunter begins searching for the hiders using proximity-based feedback. This feedback includes a numerical distance display, a visual meter, and a repeating audio beep that becomes faster as the hunter approaches a hider. The game ends when a hider is found or when the game is manually ended.

The project was implemented using HTML, CSS, and JavaScript and makes extensive use of modern browser APIs such as Geolocation, Web Audio, Vibration, and Wake Lock. Artificial intelligence tools, specifically GitHub Copilot and Claude.ai, were used throughout the development process to assist with writing code, solving technical problems, and understanding unfamiliar concepts. This report reflects on how these AI tools were used strategically, how their suggestions were evaluated, and how they influenced both the technical and creative aspects of the project.

1. Strategic AI Briefing Documentation

At the beginning of the project, I provided AI tools with a clear description of the overall goal and constraints of the system I was trying to build. The initial project brief explained that the game needed to run entirely in a web browser, be playable on mobile devices, and rely on GPS and audio rather than complex graphics. I also stated that the project was an academic assignment, meaning that clarity, maintainability, and learning value were more important than extreme optimisation.

When interacting with GitHub Copilot and Claude.ai, I frequently described the technical context in detail. For example, when asking about audio generation, I explained that I wanted to use short sine-wave beeps created with the Web Audio API rather than prerecorded sound files. I also mentioned that mobile browsers require a user interaction before audio playback can begin, which helped the AI generate more realistic and usable solutions.

Although the project did not use SuperCollider directly, I often framed audio questions using concepts similar to digital signal processing. This included references to oscillators, gain envelopes, timing intervals, and latency. By using this terminology, the AI was better able to produce code that aligned with audio programming best practices.

Over time, my prompting strategy became more refined. Early prompts were often broad and produced generic answers. As I gained a better understanding of the problem space, I began writing more focused prompts that included constraints such as update frequency, battery usage, GPS accuracy thresholds, and user experience considerations. I also started pasting short sections of my existing code into prompts so that AI-generated suggestions would integrate more smoothly with the current structure of the project.

2. Example Prompts and AI Outputs

Several AI interactions played an important role in shaping the final implementation of Hide and Beep. One of the earliest and most important interactions focused on mapping GPS distance to audio feedback timing. I asked the AI to suggest a simple way of converting distance in metres into a repeating beep interval. The AI responded with a threshold-based approach that grouped distances into bands. This solution was easy to understand, computationally cheap, and well suited to a mobile game context.

Another significant interaction involved generating audio using the Web Audio API. The AI provided an example using an oscillator and gain node to produce a short beep. While the example was technically correct, it did not initially account for mobile browser restrictions such as the need for a user gesture to unlock audio. By modifying the suggested code to include an audio unlock step and a smoother gain envelope, I was able to adapt the AI output into a reliable solution.

The AI was also used to assist with calculating distances between players using GPS coordinates. In this case, the AI suggested an implementation of the Haversine formula. This response was particularly useful because it provided an accurate and widely accepted method for calculating distances on the Earth's surface. Integrating this function into the project allowed the hunter's proximity feedback to be based on real-world measurements rather than approximations.

Another example involved reducing GPS jitter. When I asked how to stabilise noisy GPS data, the AI suggested averaging recent location fixes. Although this approach is simple, it significantly improved the smoothness of distance updates during gameplay. I extended this idea slightly by limiting the number of stored fixes and incorporating accuracy checks.

Overall, these examples demonstrate that AI outputs were most useful when they addressed clearly defined sub-problems. In each case, the AI provided a starting point that I then adapted, tested, and integrated into the wider system.

3. Critical Filtering and Decision Making

A crucial part of working with AI tools during this project was deciding which suggestions to accept, modify, or reject. Not all AI outputs were appropriate for the specific goals and constraints of Hide and Beep. One example of a rejected suggestion was the idea of using continuous audio pitch changes to represent distance. Although this could have provided detailed feedback, I decided against it because it would likely be distracting and difficult for players to interpret, especially in outdoor environments.

Another suggestion that was rejected involved increasing the frequency of GPS updates to improve accuracy. While this might have worked in theory, it raised concerns about battery consumption and performance on mobile devices. Instead, I chose to update GPS data once per second and added checks to pause gameplay feedback when accuracy dropped below an acceptable threshold.

Many AI-generated code snippets were also modified before being used. For example, audio examples were extended with gain envelopes to prevent clicking, and GPS-related suggestions were combined with spoof-detection logic to handle unrealistic movement. These changes were guided by criteria such as reliability, simplicity, user experience, and performance. Only solutions that met these criteria and could be clearly understood were included in the final codebase.

4. Collaboration Strategy Reflection

My approach to collaborating with AI tools evolved significantly over the course of the project. Initially, I relied on AI to generate larger sections of code quickly. However, this often resulted in solutions that did not fully match the project's needs or required extensive debugging.

As the project progressed, I began using AI more strategically as a problem-solving assistant rather than a code generator. I found that asking focused questions about specific issues, such as GPS smoothing or audio timing, produced more useful results. This shift also improved my own understanding, as I spent more time analysing and adapting the AI's suggestions.

Working with AI tools had a positive impact on my learning and creative process. It exposed me to correct implementations of algorithms and APIs while still requiring me to take responsibility for integration and testing. From a professional perspective, this project demonstrated how AI can support audio programming workflows when used thoughtfully, while also highlighting the importance of human judgement and ethical responsibility.

5. Reflections

Original Aims and Intended Audience

The original aim of the Hide and Beep artefact was to design an interactive audio-led game that encourages players to engage with their physical environment rather than a screen alone. The project was designed primarily for casual players, such as students or friends, who are looking for a simple outdoor multiplayer experience that can be played using their own smartphones.

From a Jobs To Be Done (JTBD) perspective, the main job of the artefact is to help players organise and play a hide-and-seek style game without needing physical props or complex rules. The system provides clear feedback to guide the hunter while keeping hiders uncertain and engaged. Key heuristics for the artefact included responsiveness, clarity of audio feedback, fairness between players, and minimal setup effort.

Influences and Precedents

Several existing tools and systems influenced the design and implementation of this project. Traditional playground games such as hide-and-seek were an obvious conceptual influence. In addition, mobile games that use location data, such as Pokémon GO, helped shape ideas about how GPS could be used responsibly and playfully.

From an audio and interaction perspective, proximity-based feedback systems used in NIME projects and interactive sound art were also influential. These systems often rely on simple sonic cues rather than complex audio, which reinforced the decision to use short beeps instead of music or speech. Studying these examples helped me understand how minimal audio can still convey meaningful information.

Implementation Process

The implementation phase was approached incrementally. I began by creating a basic single-page web application structure with multiple views representing different game states. Once navigation was stable, I added GPS tracking and distance calculation. Audio feedback was implemented later, after ensuring that location data was reliable.

Several technical decisions were made during implementation. For example, I chose to use standard browser APIs rather than external libraries to reduce complexity and improve transparency. Viability testing was carried out informally by running the game on different smartphones to check GPS accuracy, audio latency, and battery usage.

One major challenge encountered was unreliable GPS data, particularly in areas with poor signal. This was addressed by adding smoothing and accuracy checks. Another challenge was mobile audio restrictions, which required the addition of an explicit audio unlock step tied to a user gesture.

Evaluation of the Artefact

When evaluated against the original aims, the finished artefact successfully meets most of its goals. The game functions as an accessible, audio-led outdoor experience that requires minimal explanation. The proximity-based audio feedback is effective and easy to interpret, fulfilling the core heuristic of clarity.

The elements I am most proud of are the distance-to-beep mapping and the robustness of the GPS handling. These systems work together to create a sense of tension and discovery during gameplay. If more time were available, I would improve the multiplayer backend, add accessibility options for different hearing abilities, and conduct more structured user testing.

6. Conclusion and Lessons Learned

In conclusion, the Hide and Beep project demonstrates how artificial intelligence tools can effectively support the development of interactive, audio-driven applications when used responsibly. GitHub Copilot and Claude.ai helped accelerate development, provided useful reference implementations, and supported my understanding of unfamiliar technical concepts.

The most important lesson learned from this project is that AI tools are most effective when used as collaborators rather than replacements for human decision-making. Clear prompts, critical evaluation, and a solid understanding of the underlying problem are essential for successful AI-assisted development. Through this project, I not only produced a functional game but also developed stronger technical judgement and confidence in my programming abilities.

AI Usage Declaration and Ethics Statement

This project made use of artificial intelligence tools, specifically GitHub Copilot and Claude.ai, as assistive technologies during the development process. These tools were used to generate code suggestions, explain programming concepts, and support problem-solving.

All AI-generated outputs were critically reviewed, tested, and modified by the author before being included in the final codebase. The overall design, creative decisions, and final implementation remain the responsibility of the author. The use of AI tools complies with academic integrity guidelines and is transparently documented in this report.

Resource used

NaviLens EMPOWERING blind and partly sighted people

Accessibility Technology for blind & low vision people - Be My Eyes

Seeing AI - A Visual Assistant for the Blind

Microsoft Copilot: Your AI companion

Claude

https://chatgpt.com/