Team 11: Sabotage - Group Report

Bowei Ren, Henry Huang, Moiz Lokhandwala, William Gong

Introduction

Our project is a game called Sabotage. The game is an Android app that uses GPS to direct the player to a game session with a hidden De1-SoC board. This GPS location is general and imprecise. Once the player is in the vicinity of the board, they play "hot or cold" to pinpoint its precise location using WiFi Received Signal Strength Indicator (RSSI) proximity detection. The player plays by tapping the button on their phone, which responds with red if the player's current location is closer to the De1-SoC than their previous location, or with blue for the opposite. In this "hot or cold" stage, other players can act as saboteurs to interfere with another player's progress by, for example, giving false information. Once a player finds a De1-SoC board, they verify the find by entering their user ID on the touchscreen connected to the De1-SoC. If a valid user ID is entered, it is added to the leaderboard for that game session. The score is based on how long it took and how many saboteurs they overcame.

The target market for this game is people who are looking to explore the UBC campus in a fun and educational way.

System Description and Accomplishments

Table 1: Successfully Implemented Components

Hardware	**Backend/Server**	**App**	**Network/Mixed**
Display number pad on VGA screen	HTTP POST and GET requests from De1-SoC and app	Player location tracking	User management
Receive touch screen inputs associated with number pad	Persistent data and consistency for arbitrary number of client devices	Initiating, and receiving sabotages on the user end.	Sabotages
Receives GPS data in NMEA format and parses them	Relay when a sabotage is active and clearing them when done	Hot or Cold button using GPS and/or WiFi RSSI data	Multiple sessions
Send HTTP POST requests over WiFi	Create an alert when a De1-SoC moves	Sending HTTP GET and POST requests to server	Leaderboard and Scoring
Setup ESP8266 as a WiFi access point		Maps UI display	

Table 2: Challenging Features

Challenging Features	**Reason**
Hardware Integration	Functions from stdio.h used in previous versions busy-waited for touch screen input, making it impossible to fetch other data periodically. We had to rewrite some functions and add interrupt handlers and timeouts to them.
Data consistency across database and every device	How data was stored in our database and how they were returned to clients were often formatted differently due to frequently changing the structure of both to meet new requirements and challenges. This led to weird conversions and inconsistencies that needed to be remedied every time the data was accessed.

Table 3: Other Components

Partially Implemented/Quirky	**Failed to implement**
GPS tracking in-doors	Linux on the De1-SoC
VGA response time is not immediate.	Accelerometer

Member Contributions

Bowei Ren	Touchscreen driver, Hardware integration
Henry Huang	WiFi module/support for De1-SoC, Backend Server and Database
Moiz Lokhandwala	GPS, VGA
William Gong	UI - Android App

Conclusions and Solution Assessment

Our design meets all the requirements for module 2. Our user requirement was fulfilled by the gameplay on the Android application and VGA and touch screen connected to the De1-SoC. The hardware requirements were fulfilled by using the Nios II to interface with the GPS, touch screen, VGA, and WiFi modules. The cloud component requirement was fulfilled by our server, which connects to the De1-SoC via the WiFi module. Since we integrated all our components into a complete product, our networking component was fulfilled.

As the GPS was not able to consistently receive accurate data indoor, it affected the robustness of our product in buildings. We would utilise a portable power supply so it can be used outdoors in future projects. We would also develop the server on a more secure framework and host it on a more scalable service.