เว็บแอพวิเคราะห์การเล่นเกมบนแพลตฟอร์มสตรีมสำหรับผู้ปกครอง

Game Playtime Analyzer

นายวรรณชัย เชื้อทอง 664230048

หมู่เรียน 66/46

โครงงานนี้เป็นส่วนหนึ่งของการศึกษารายวิชา 7204903

โครงงานด้านเทคโนโลยีสารสนเทศ 2

สาขาวิชาเทคโนโลยีสารสนเทศ คณะวิทยาศาสตร์และเทคโนโลยี

มหาวิทยาลัยราชภัฏนครปฐม

ภาคเรียนที่ 1 ปีการศึกษา 2567

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ในปัจจุบัน การเล่นเกมออนไลน์ได้กลายเป็นส่วนหนึ่งในชีวิตประจำวันสำหรับผู้คนทุกเพศทุกวัย และเป็นกิจกรรมที่ผู้จัดทำเองก็ใช้เวลาค่อนข้างมากหากไม่จัดการเวลาเล่นที่เหมาะสม ซึ่งส่วนหนึ่งที่ชอบเล่นและใช้เวลามากก็คือเด็กๆ ที่ขาดการควบคุมและจัดการแบ่งเวลาด้วยตัวเองจนทำให้เสียสุขภาพ จึงทำให้หน้าที่นั้นเป็นของ ผู้ปกครองจำนวนมาก และประสบปัญหาในการตรวจสอบและยืนยันพฤติกรรมการเล่นเกมจริง ของบุตรหลาน เนื่องจากเครื่องมือที่มีอยู่บนแพลตฟอร์มเกมมักจะจำกัดอยู่แค่ข้อมูลผลรวมที่ไม่มีการแจ้งเตือนและไม่สามารถนำมาเป็น หลักฐาน ในการพูดคุยเพื่อปรับเปลี่ยนพฤติกรรมได้ หากมองในส่วนของผู้ปกครองที่มีลูกแต่ไม่มีเวลามากพอในการดูแล และหากละเลยก็อาจทำให้ลูกเสียสุขภาพได้ แล้วยังขาดเครื่องมือที่ช่วยให้ดูเวลาการเล่นและทำความเข้าใจพฤติกรรมการเล่นเกมของลูกตนเองได้อย่างเป็นระบบ

นอกจากนี้ เวลาที่เด็กใช้ไปกับการเล่นเกม คือเวลาที่ถูกแลกมาจากการทำกิจกรรมอย่างอื่นที่สำคัญต่อพัฒนาการ เช่น การบ้าน การอ่านหนังสือ การออกกำลังกาย การเข้าสังคมกับเพื่อนในโลกแห่งความจริง หรือแม้กระทั่งการนอนหลับพักผ่อน ความสำคัญของปัญหานี้จึงไม่ใช่แค่ เด็กเล่นเกมเยอะ แต่เป็นสิ่งที่ควรจะตระหนักรู้ของเด็กๆเองว่าผลเสียที่จะตามมานั้นเป็นปัญหาแค่ไหน แต่เด็กส่วนใหญ่มักจะไม่รู้ในส่วนนี้เพราะเกมปัจจุบันส่วนใหญ่ได้ถูกออกแบบมาอย่างดีที่ทำให้การเลิกเล่นนั้นยากไปเลย จากเหตุนี้เองที่ผู้ปกครองสูญเสียความสามารถในการดูแลและสร้างสมดุลในชีวิตให้บุตรหลาน เนื่องจากช่องว่างทางข้อมูลและเทคโนโลยี และด้วยเหตุนี้ผู้ปกครองจึงมีส่วนสำคัญมากในการช่วยจัดสรรเวลาให้ลูกๆ อย่างเหมาะสม

1.2 แนวคิดในการแก้ไขปัญหา

เพื่อแก้ไขปัญหานี้ โครงการจึงมุ่งเน้นการพัฒนา ระบบติดตามและวิเคราะห์พฤติกรรมการเล่นเกมบุตรหลาน ที่ใช้ประโยชน์จากแพลตฟอร์มสตรีมเว็บเอพีไอ (Steam Web API) ซึ่งเป็น API สาธารณะที่เปิดให้นักพัฒนาสามารถดึงข้อมูลเวลาการเล่นเกมที่ เป็นหลักฐาน ได้โดยตรง ทำให้สามารถนำข้อมูลมาประมวลผล เปรียบเทียบกับเกณฑ์ที่ตั้งไว้ และส่งการแจ้งเตือนให้ผู้ปกครองทราบทันที

หลังจากกดปุ่มตรวจสอบ เป็นการปิดช่องว่างทางข้อมูลและช่วยให้ผู้ปกครองสามารถสร้างสมดุลในชีวิตดิจิทัลให้กับบุตรหลานได้อย่างมีประสิทธิภา

การออกแบบระบบจะใช้เทคโนโลยี โนตดอทเจเอส และ แอ็กเพรสดอทเจเอส เป็นส่วนกลางในการประมวลผลคำขอจากส่วนหน้าบ้าน และเชื่อมต่อกับ ฐานข้อมูล เพื่อใช้ในการบันทึกข้อมูลสถิติการเล่น อีกทั้งยังมีการพัฒนาระบบแจ้งเตือนผ่านดิสคอร์ด เมื่อผู้เล่นมีเวลาเล่นสะสมเกินเกณฑ์ที่กำหนด ซึ่งจะช่วยให้ผู้ปกครองสามารถติดตามพฤติกรรมการเล่นเกมของบุตรหลานได้ อย่างมีประสิทธิภาพ

1.3 วัตถุประสงค์ของระบบ

การพัฒนาระบบนี้มีวัตถุประสงค์หลักเพื่อแก้ไขปัญหาและตอบสนองความต้องการที่ได้ระบุไว้ในหัวข้อที่ผ่านมา โดยมีวัตถุประสงค์ที่สำคัญดังต่อไปนี้

1.3.1 เพื่อพัฒนาระบบวิเคราะห์เวลาการเล่นเกมจาก สตรีมเว็บเอพีไอ (Steam Web API) ที่สามารถแสดงผลในรูปแบบ แดชบอร์ด (Dashboard) ที่เข้าใจง่าย

1.3.2 เพื่อสร้างระบบแจ้งเตือนทางดิสคอร์ด เมื่อเวลาการเล่นเกมสะสมของผู้เล่นเกินกว่าที่ผู้ปกครองกำหนดชั่วโมง ในรอบ 2 สัปดาห์

1.3.3 เพื่อสร้างระบบฐานข้อมูลที่สามารถบันทึกข้อมูลการเล่นเกมและรายชื่อเกมเพื่อนำไปใช้ในการแสดงผล

1.4 ขอบเขตการศึกษา

1.4.1 ขอบเขตของระบบ

1.4.1.1 ผู้ดูแลระบบ

ก) จัดการฐานข้อมูล supabase ของผู้ใช้งาน

1.4.1.2 ผู้ใช้งานระบบ

ก) ผู้ปกครองที่ต้องการตรวจสอบข้อมูลการเล่นเกมของลูก

ข) ผู้ใช้ทั่วไป ที่ต้องการดูสถิติของการเล่นเกม

1.4.2 ฮาร์ดแวร์ที่ใช้ในการพัฒนา

1.4.2.1 โน้ตบุ๊ก เอเซอร์ เพรเดเตอร์ ไฮลิออส (Acer Predator Helios)

หน่วยประมวลผลกลาง: Intel Core i5-14500HX

แรม: 16 GB DDR5

พื้นที่จัดเก็บข้อมูล: 512 GB PCIe-4.0 NVMe

จอ: 16″ WUXGA (1920×1200) 165Hz sRGB100%

หน่วยประมวลผลกราฟิก: NVIDIA GeForce RTX 4050 6 GB

1.4.3 ซอฟต์แวร์ที่ใช้ในการพัฒนา

1.4.3.1 ระบบปฏิบัติการ ไมโครซอฟท์วินโดวส์ อีเลฟเว่น (Microsoft Window 11)

1.4.3.2 วิชวล สตูดิโอ โค้ด (Visual Studiocode) เป็นเครื่องมือในการพัฒนา

1.4.3.3 เอชทีเอ็มแอล ซีเอสเอส จาวาสคริปต์ สำหรับการพัฒนา ฟรอนท์เอนด์ (UI)

1.4.3.4 โหนด เจเอส (Node.js) สำหรับการรันคำสั่ง JavaScript

1.4.3.5 เอ็กซ์เพรส เจเอส (Express.js) สำหรับการพัฒนาแบ็คเอนด์

1.4.3.6 ซูพาเบส (Supabase) เป็นระบบจัดการฐานข้อมูลของผู้ใช้

1.4.3.7 กิตฮับ (Github) สำหรับจัดเก็บและจัดการโค้ดโปรแกรมของโครงงาน

1.4.3.8 ดิสคอร์ด เว็บฮุก (Discord webhook) สำหรับการส่งแจ้งเตือน

1.4.3.9 ไฟเออร์เบส สตูดิโอ (Firebase Studio) ใช้สำหรับ ดีพลอย เว็บ

1.4.3.10 สตีม เว็บ เอพีไอ (Steam Web API) สำหรับดึงข้อมูลการเล่นเกม

1.4.3.11 โพสต์แมน (Postman) การเทส (Steam Web API) เพื่อทราบว่าสามารถดูข้อมูลของผู้เล่นได้จริง

1.5 ประโยชน์ที่ได้คาดว่าจะได้รับ

1.5.1 ด้านการใช้งาน ระบบนี้จะช่วยให้ผู้ใช้หรือผู้ปกครองสามารถเข้าใจพฤติกรรมการใช้เวลาในการเล่นเกมลูกเพื่อวิเคราะห์เวลาเล่นของลูกได้อย่างเป็นรูปธรรม ทำให้สามารถบริหารเวลาได้อย่างมีประสิทธิภาพมากขึ้น นอกจากนี้ยังช่วยให้ผู้ปกครองสามารถติดตามและได้รับการแจ้งเตือนเกี่ยวกับพฤติกรรมการเล่นเกมของบุตรหลานได้อย่างทันท่วงที ซึ่งจะช่วยส่งเสริมให้เกิดความสมดุลระหว่างการใช้ชีวิตกับการเล่นเกม

1.5.2 ด้านการประยุกต์ใช้ โครงงานนี้สามารถนำไปใช้เป็นพื้นฐานสำหรับการสร้างระบบแจ้งเตือนอัตโนมัติอื่นๆ ในอนาคต โดยสามารถนำหลักการในการเชื่อมต่อ API การจัดการฐานข้อมูล และการสร้าง Dashboard ไปประยุกต์ใช้กับข้อมูลในหลากหลายอุตสาหกรรม

1.5.3 ด้านการสร้างความรู้ การพัฒนาโปรเจกต์นี้จะสร้างองค์ความรู้ใหม่เกี่ยวกับการประยุกต์ใช้ เครื่องมือในการใช้ API และพัฒนาโปรเจกต์ ซึ่งเป็นแนวทางที่ทันสมัยและมีประโยชน์อย่างยิ่งสำหรับนักพัฒนารุ่นใหม่

1.6 คำนิยาม

1.6.1 ผู้ใช้งานหรือ (User) หมายถึงบุคคลที่ใช้ระบบหรือแอปพลิเคชันเพื่อดำเนินการต่าง ๆ โดยผู้ใช้สามารถกรอกข้อมูลที่จำเป็น เช่น การใส่ไอดีสตรีมของตัวเอง หรือ ของลูก และการใส่ข้อมูลของเว็บฮุกในการแจ้งเตือนจากระบบ

1.6.2 สตรีมเว็บเอพีไอ (Steam Web API) หมายถึงชุดคำสั่งและเครื่องมือสำหรับนักพัฒนาเพื่อใช้สำหรับการดึงข้อมูลเวลาเล่นเกมจาก สตรีมมาให้อยู่ในรูปแบบแดชบอร์ด

1.6.3 แดชบอร์ด (Dashboard) หน้าจอที่แสดงข้อมูลสำคัญสรุปในรูปแบบภาพ (เช่น กราฟ ตาราง) ทำให้เข้าใจง่ายและเห็นภาพรวมได้อย่างรวดเร็ว เพื่อใช้ติดตามและวิเคราะห์ข้อมูล

1.6.4 โนตดอทเจเอส (Node.js) และ แอ็กเพรสดอทเจเอส (Express.js) หมายถึงเทคโนโลยีที่ใช้ในการพัฒนาส่วนหลังบ้าน (Backend) ของระบบ โดย Node.js ทำหน้าที่เป็นสภาพแวดล้อมให้โค้ด JavaScript สามารถทำงานบนเซิร์ฟเวอร์ได้ ส่วน Express.js เป็นเฟรมเวิร์กที่ช่วยจัดการการทำงานของเซิร์ฟเวอร์ เช่น การสร้าง API เพื่อรับคำขอจากส่วนหน้าบ้าน การประมวลผล และการส่งข้อมูลกลับไป

1.6.5 ซูพาเบส (Supabase) หมายถึง ระบบจัดการฐานข้อมูลที่โครงงานนี้เลือกใช้เพื่อบันทึกข้อมูลสถิติการเล่นเกม มีหน้าที่สำคัญในการจัดเก็บข้อมูลการเล่นเกมล่าสุดไว้ เพื่อให้ระบบสามารถนำข้อมูลมาเปรียบเทียบและคำนวณหาเวลาเล่นที่เกิดขึ้นจริงในรอบ 2 สัปดาห์ได้ ซึ่งตอบสนองวัตถุประสงค์ในการสร้างระบบฐานข้อมูลเพื่อการแสดงผล

1.6.6 ดิสคอร์ด เว็บฮุก (Discord Webhook) หมายถึง ช่องทางที่ใช้ในการส่งข้อความแจ้งเตือนอัตโนมัติไปยังแอปพลิเคชัน Discord ในโครงงานนี้ ระบบจะใช้ Discord Webhook เพื่อส่งการแจ้งเตือนไปยังผู้ปกครองทันที เมื่อตรวจพบว่าเวลาการเล่นเกมสะสมของบุตรหลานเกินกว่าเกณฑ์ที่กำหนดไว้ในรอบ 2 สัปดาห์

1.6.7 วิชวล สตูดิโอ โค้ด (Visual Studio Code) หมายถึง โปรแกรมประเภท Code Editor ที่ผู้จัดทำใช้เป็นเครื่องมือหลักในการเขียนและแก้ไขชุดคำสั่ง (Source Code) ทั้งหมดของโปรเจกต์นี้ ตั้งแต่ส่วนของ Frontend (HTML, CSS, JavaScript) ไปจนถึงส่วนของ Backend (Node.js)

บทที่ 2

หลักการและทฤษฎีที่เกี่ยวข้อง

การศึกษาเนื้อหาเกี่ยวกับทฤษฎีและงานวิจัยที่เกี่ยวข้องถือเป็นสิ่งสำคัญอย่างยิ่งต่อการศึกษาค้นคว้าในครั้งนี้ ซึ่งเป็นการแสดงให้เห็นถึงการสำรวจตรวจสอบหรือทบทวนเอกสารต่าง ๆ ที่เกี่ยวข้องอย่างละเอียดครบถ้วนของผู้จัดทำ โดยมีจุดประสงค์เพื่อให้ผู้อ่านเห็นประเด็นปัญหาของการศึกษาและแนวความคิดในการแก้ไขปัญหาอย่างชัดเจน ผู้จัดทำได้นำทฤษฎีและองค์ความรู้ที่เกี่ยวข้องมาวิเคราะห์เพื่อแสดงให้เห็นถึงความสัมพันธ์กับปัญหาที่กำลังศึกษา เพื่อให้การพัฒนาระบบสามารถบรรลุเป้าหมายที่ตั้งไว้ได้อย่างสำเร็จลุล่วง

2.1 ระบบงานเดิม

ระบบงานเดิมที่ใช้ในการติดตามเวลาการเล่นเกมส่วนใหญ่จะจำกัดอยู่เพียงการดูข้อมูลสรุปในโปรไฟล์ส่วนตัวบนแพลตฟอร์มเกมนั้น ๆ ซึ่งทำให้ผู้ใช้งานขาดเครื่องมือในการวิเคราะห์พฤติกรรมการเล่นเกมอย่างเป็นระบบ และทำให้เกิดข้อจำกัดหลายประการ ได้แก่ ขาดการแจ้งเตือน ผู้ปกครองหรือผู้เล่นไม่ได้รับการแจ้งเตือนอัตโนมัติเมื่อมีการเล่นเกมเกินเวลาที่กำหนด ขาดการวิเคราะห์เชิงลึก ข้อมูลที่แสดงผลเป็นเพียงผลรวมของเวลาเล่น ไม่มีการวิเคราะห์ในรูปแบบกราฟหรือสถิติที่เข้าใจง่าย ขาดการเชื่อมโยงข้อมูล ไม่สามารถผูกข้อมูลการเล่นเกมของบุตรหลานเข้ากับบัญชีของผู้ปกครองเพื่อการติดตามได้อย่างเป็นระบบ

2.2 ระบบงานอื่นที่เกี่ยวข้อง

ระบบ "Screen Time" ของ Apple และ "Digital Wellbeing" ของ Google เป็นฟีเจอร์ระดับปฏิบัติการที่ถูกติดตั้งมาพร้อมกับอุปกรณ์พกพา มีวัตถุประสงค์เพื่อให้ผู้ใช้สามารถตรวจสอบและจัดการเวลาที่ใช้ไปกับแอปพลิเคชันต่าง ๆ รวมถึงเกมได้ด้วยตนเอง ระบบดังกล่าวมีความสามารถในการแสดงผลข้อมูลการใช้งานในรูปแบบของกราฟสถิติรายวันและรายสัปดาห์ พร้อมทั้งมีฟังก์ชันจำกัดเวลาการใช้งาน (App Limits) ซึ่งจะส่งการแจ้งเตือนเมื่อใช้งานเกินเวลาที่กำหนด แม้ว่าระบบเหล่านี้จะครอบคลุมการใช้งานที่กว้างกว่าแค่การเล่นเกม แต่ก็มีแนวคิดหลักที่คล้ายคลึงกับโครงงานนี้ในด้าน การสร้างความตระหนักรู้ (Awareness) ผ่านการแสดงข้อมูลที่เป็นรูปธรรมและการแจ้งเตือน อย่างไรก็ตาม ระบบดังกล่าวมีข้อจำกัดในการเข้าถึงข้อมูลจากแพลตฟอร์มเฉพาะทางอย่าง Steam และไม่ถูกออกแบบมาสำหรับการติดตามโดยผู้ปกครองจากระยะไกลโดยเฉพาะ ซึ่งเป็นช่องว่างที่โครงงานนี้ได้เข้ามาพัฒนาเพิ่มเติม ฟังก์ชันการทำงาน: ระบบจะติดตามและบันทึกระยะเวลาการใช้

งานของทุกแอปพลิเคชันบนอุปกรณ์โดยอัตโนมัติ จากนั้นจะสรุปผลและแสดงข้อมูลในรูปแบบของแดชบอร์ดที่เข้าใจง่าย ประกอบด้วยกราฟสถิติรายวันและรายสัปดาห์ ผู้ใช้สามารถตั้งค่าจำกัดเวลาการใช้งาน (App Limits) สำหรับแอปพลิเคชันบางประเภท และระบบจะส่งการแจ้งเตือนเมื่อใช้งานใกล้ครบหรือเกินเวลาที่กำหนด นอกจากนี้ยังมีฟังก์ชัน "Downtime" เพื่อกำหนดช่วงเวลาปลอดหน้าจออีกด้วย

2.3.1 ไมโครซอฟท์ วินโดวส์ 11

เป็นระบบปฏิบัติการที่ใช้ในการทำโปรแกรมต่างๆ ของโครงงานนี้ เนื่องจากเป็นระบบปฏิบัติการที่มีเสถียรภาพสูง รองรับซอฟแวร์และเครื่องมือพัฒนาที่หลากหลาย อีกทั้งยังมีระบบการจัดไฟล์ การจัดการทรัพยากร และระบบรักษาความปลอดภัยที่เหมาะสมต่อการพัฒนาและทดสอบระบบ

ภาพที่ 2.3.1 ไมโครซอฟท์ วินโดวส์ 11

ที่มา https://news.microsoft.com/th-th/2021/06/25/windows11-th/

2.3.2 วิชวล สตูดิโอ โค้ด

โปรแกรมแก้ไขซอร์สโค้ด (code editor) โครงงานนี้ ผู้จัดทำได้ใช้ Visual Studio Code เป็นเครื่องมือหลักในการพัฒนาโค้ดทั้งหมด ทั้งในส่วนของแก้ไขโค้ดและใช้คำสั่ง

ภาพที่ 2.3.2 วิชวล สตูดิโอ โค้ด

ที่มา https://shorturl.asia/lSKEJ

2.3.3 แฮชทีเอ็มแอลไฟท์

HTML5 มีบทบาทสำคัญในการสร้าง โครงสร้าง (Structure) ของเว็บเพจ ไม่ว่าจะเป็นหน้าหลัก หน้าแสดงข้อมูลของผู้ใช้ หรือหน้าตั้งค่าการแจ้งเตือน โดยคุณสามารถยกตัวอย่างการใช้แท็ก (Tag) เช่น <table> สำหรับการแสดงข้อมูลตาราง หรือ <form> สำหรับการกรอกข้อมูล Steam ID เพื่อให้เห็นภาพชัดเจนขึ้น

ภาพที่ 2.3.3 แฮชทีเอ็มแอลไฟท์

ที่มา https://shorturl.asia/qlQZx

2.3.4 ซีเอสเอสสาม

CSS3 คือหัวใจสำคัญของการ ออกแบบ (Styling) หน้าเว็บให้สวยงาม น่าใช้งาน และแสดงผลได้ถูกต้องบนอุปกรณ์ต่าง ๆ (Responsive Design) คุณสามารถยกตัวอย่างการใช้ CSS ใน

การกำหนดสี ขนาดตัวอักษร การจัดวางองค์ประกอบต่างๆ หรือการสร้างกราฟเพื่อแสดงสถิติเวลาการเล่นเกม คุณสมบัติเด่นของ CSS3 เช่น Flexbox หรือ Grid Layouts ที่ช่วยให้การจัดวางองค์ประกอบมีความยืดหยุ่นและเป็นระเบียบ รวมถึงการใช้ Animation หรือ Transition เพื่อเพิ่มความน่าสนใจให้กับ UI

ภาพที่ 2.3.4 ซีเอสเอสสาม
ที่มา https://shorturl.asia/K54Qa

2.3.5 จาวาสคริปต์

JavaScript ทำหน้าที่เป็นตัวกลางสำคัญในการสื่อสารระหว่าง Frontend กับ Backend ผ่านการเรียกใช้ API เพื่อให้ผู้ใช้ได้รับข้อมูลแบบ Real-time โดยไม่ต้องโหลดหน้าเว็บใหม่ทั้งหมด บทบาทของ JavaScript เป็นภาษาที่ใช้ในการสร้าง การโต้ตอบ (Interactivity) บนหน้าเว็บ ไม่ใช่แค่การแสดงผลแบบคงที่ ยกตัวอย่างการใช้ JavaScript ในการส่งข้อมูล Steam ID จากฟอร์มไปยัง Backend หรือการดึงข้อมูล JSON กลับมาแสดงผลในรูปแบบของกราฟหรือข้อมูลสถิติ

ภาพที่ 2.3.5 จาวาสคริปต์
ที่มา https://logos-world.net/javascript-logo/

2.3.6 โนตเจเอส

บทบาทหลักของ โนตเจเอส คือ การจัดการตรรกะทางธุรกิจ (Business Logic) การจัดการฐานข้อมูล และการจัดการ API ภายนอกอย่าง Steam Web API คือส่วนที่ทำให้ JavaScript ทำงานได้นอกเหนือจากเว็บเบราว์เซอร์ ซึ่งในที่นี้คือการทำงานบนเซิร์ฟเวอร์ ส่วน Express.js คือ Web Application Framework ที่ช่วยให้การสร้าง API และจัดการการร้องขอ (Request) จาก Frontend เป็นเรื่องง่ายและเป็นระบบ

ภาพที่ 2.3.6 โนตดอทเจเอส

ที่มา https://www.curotec.com/insights/node-js-vs-express-js/

2.3.7 เอ็กซ์เพรสเจเอส

เอ็กซ์เพรสเจเอส เว็บเฟรมเวิร์ก (Web Framework) สำหรับ Node.js ที่ช่วยให้การสร้างเว็บเซิร์ฟเวอร์หรือ REST API ง่ายและเป็นระบบมากขึ้น

ภาพที่ 2.3.7

ที่มา https://humand.co.uk/best-nodejs-frameworks/

2.3.9 ซูปาร์เบส

การเลือกใช้ Supabase โดยเฉพาะนั้นมาจากแนวคิด Backend-as-a-Service (BaaS) ที่คล้ายคลึงกับ Firebase แต่ทำงานบนรากฐานของ PostgreSQL ซึ่งเป็นการรวมข้อดีของทั้งสองเข้าด้วยกัน คือ ความง่ายในการใช้งานของแพลตฟอร์มที่มีการจัดการให้ (เช่น การสร้าง API อัตโนมัติ

ระบบยืนยันตัวตน พื้นที่จัดเก็บไฟล์) นอกจากนี้ การที่ Supabase เป็นโอเพนซอร์สยังช่วยให้ไม่ต้องผูกติดกับผู้ให้บริการรายใดรายหนึ่ง (Vendor Lock-in)

ภาพที่ 2.3.9 ซูปาร์เบส

ที่มา SUPABASE - DEV Community

2.3.10 ไฟร์เบสสตูดิโอ

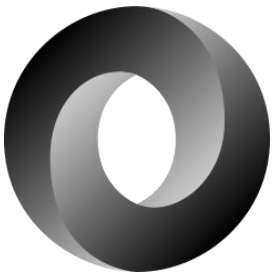
Firebase Studio ใช้ในการ Deploy เว็บ และ เป็นบริการจาก Google สำหรับโครงงานนี้จะใช้สำหรับโฮสต์ (Deploy) เว็บแอปพลิเคชัน ทำให้ผู้ใช้งานสามารถเข้าถึงได้ผ่านอินเทอร์เน็ต

ภาพที่ 2.3.10

ที่มา https://firebase.studio/

2.3.11 เจซัน

JSON คือรูปแบบข้อมูลที่มนุษย์สามารถอ่านและเข้าใจได้ง่าย ซึ่งถูกนำมาใช้เป็นรูปแบบมาตรฐานในการแลกเปลี่ยนข้อมูลในทุกส่วนของระบบ โดยมีบทบาทสำคัญดังนี้ การรับ-ส่งข้อมูลกับ Steam Web API ระบบจะรับข้อมูลดิบ (Raw Data) ที่เป็นไฟล์ JSON จาก Steam Web API ซึ่งข้อมูลจะมาในรูปแบบของ Objects และ Arrays ที่แสดงถึงข้อมูลเกมและเวลาการเล่น การสื่อสารระหว่าง Backend และ Frontend: JSON เป็นรูปแบบหลักที่ใช้ในการรับ-ส่งข้อมูลระหว่าง Frontend (เบราว์เซอร์) และ Backend (Node.js) ผ่าน RESTful API ไฟล์ตั้งค่า ข้อมูลการตั้งค่าต่าง ๆ ของโปรเจกต์อาจถูกเก็บในไฟล์ JSON เช่น package.json ที่ระบุ Library ที่ต้องใช้



ภาพที่ 2.3.11 เจซัน

ที่มา File:JSON vector logo.svg - Wikimedia Commons

2.3.12 ดิสคอร์ด เว็บฮุค

เป็นวิธีที่ง่ายและปลอดภัยในการส่งข้อความอัตโนมัติไปยัง Discord Channel โดยไม่จำเป็นต้องใช้ Discord Bot ที่มีความซับซ้อนและต้องตั้งค่าเพิ่มเติม ในโครงการนี้ Webhook ถูกนำมาใช้เพื่อทำหน้าที่เป็น API Endpoint สำหรับรับข้อมูลจาก Backend ของเรา และทำการส่งข้อความแจ้งเตือนอัตโนมัติไปยังผู้ปกครอง เมื่อเวลาการเล่นเกมของบุตรหลานเกินกว่าที่กำหนด

ภาพที่ 2.3.12 ดิสคอร์ด เว็บฮุค

ที่มา https://coda.io/packs/discord-webhook-28767

2.3.13 กิตฮับ

แพลตฟอร์มออนไลน์สำหรับจัดเก็บและจัดการโค้ดโปรแกรม ของโครงงาน โดยอาศัยระบบกิตทำให้ผู้พัฒนาสามารถจัดการโค้ดในแต่ละเวอร์ชั่นได้อย่างเป็นระบบอีกทั้ง กิตฮับยังเป็นพื้นที่สำหรับจัดเก็บและเผยแพร่ซอร์สโค้ดให้ผู้อื่นเข้ามาศึกษาหรือพัฒนาต่อยอดได้

ภาพที่ 2.3.13

ที่มา https://foundations.projectpythia.org/foundations/github/what-is-github/

2.3.14 **แอ็ก-เชียส**

เป็นไลบรารีของโครงงานนี้โดยจะอยู่ในส่วนของ Backend จะได้ใช้ Axios ในการส่งคำขอไปยัง Steam Web API เพื่อดึงข้อมูลการเล่นเกมของผู้เล่นมาทำการวิเคราะห์ และจัดเรียง

ภาพที่ 2.3.14

ภาพที่ https://shorturl.asia/yvNGL

2.3.15 ชาร์ตดอทเจเอส

คือ JavaScript library แบบโอเพนซอร์สที่ช่วยให้นักพัฒนาสามารถสร้างกราฟและแผนภูมิที่สวยงามและตอบสนองได้ (Responsive) บนหน้าเว็บได้อย่างง่ายดาย ในโปรเจกต์นี้ Chart.js ถูกนำมาใช้เพื่อแปลงข้อมูลตัวเลข (เวลาการเล่นเกม) ให้อยู่ในรูปแบบของแผนภูมิแท่งที่ผู้ใช้งานสามารถเข้าใจได้ทันที

ภาพที่ 2.3.15 ชาร์ตดอทเจเอส

ที่มา https://dev.to/prathameshk73/using-chart-js-in-angular-app-11ek

2.3.16 สตรีมเว็บเอพีไอ

ข้อมูลหัวใจหลักของโครงงานนี้ได้มาจาก Endpoint IPlayerService/GetOwnedGames ของ Steam Web API การเรียกใช้งาน Endpoint นี้เป็นการส่งคำร้องขอแบบ GET ซึ่งจำเป็นต้องมีพารามิเตอร์ที่สำคัญคือ key (คีย์ API ของผู้พัฒนา) และ steamid (รหัสผู้ใช้ Steam 64-bit) นอกจากนี้ ยังมีพารามิเตอร์เสริมที่สำคัญอย่างยิ่งคือ include_appinfo=true เพื่อให้ได้ข้อมูลชื่อและไอคอนของเกมกลับมาด้วย และ include_played_free_games=true เพื่อให้แน่ใจว่าได้ข้อมูลการเล่นเกมฟรีที่ผู้ใช้อาจเคยเล่นมาด้วย ซึ่งจะทำให้ได้ภาพรวมกิจกรรมของผู้ใช้ที่สมบูรณ์ที่สุด

ภาพที่ 2.3.16

ที่มา https://shorturl.asia/Mh9dL

2.3.17 เรสฟูเอพีไอ

เพื่อให้การสื่อสารระหว่างไคลเอนต์ เซิร์ฟเวอร์ และบริการภายนอก (Steam API) เป็นไปอย่างมีมาตรฐานและประสิทธิภาพ โครงงานจึงได้นำระบบได้นำหลักการของ REST (Representational State Transfer) มาใช้ในการออกแบบ เป็นรูปแบบสถาปัตยกรรม (Architectural Style) ที่กำหนดข้อจำกัดบางประการเพื่อสร้างเว็บเซอร์วิสที่มีความยืดหยุ่นและง่ายต่อการพัฒนา API ที่ออกแบบตามหลักการนี้จะถูกเรียกว่า "RESTful API" API ที่ออกแบบตามหลักการนี้จะถูกเรียกว่า 'RESTful API' ซึ่งในโครงงานนี้ Backend ที่พัฒนาขึ้นทำหน้าที่เป็นทั้งผู้ให้บริการ RESTful API แก่ Frontend และในขณะเดียวกันก็เป็นผู้เรียกใช้ RESTful API ภายนอกอย่าง Steam Web API ไปด้วย

ภาพที่ 2.3.17

ที่มา https://shorturl.asia/FZO32

2.3.18 โพสต์แมน

คือโปรแกรม (หรือ extension บนเว็บ/เดสก์ท็อป) ที่ใช้สำหรับ ทดสอบ (Test) พัฒนา (Develop) และ จัดการ (Manage) API โดยจะเกี่ยวข้องในส่วนของการเทส (Steam Web API) เพื่อทราบว่าสามารถดูข้อมูลของผู้เล่นได้จริง

ภาพที่ 2.3.18

ที่มา https://apidog.com/blog/what-is-postman/

บทที่ 3
วิธีการดำเนินงาน

3.1 การศึกษาเบื้องต้น

3.1.1 การศึกษาปัญหาของระบบ Game Playtime Analyzer แบบดั้งเดิม

จากการทบทวนวรรณกรรมและศึกษาข้อมูลเกี่ยวกับปัญหาที่ผู้ปกครองในยุคดิจิทัลต้องเผชิญในการดูแลพฤติกรรมการเล่นเกมของบุตรหลานตามที่ระบุไว้ในบทที่ 1 พบว่าเครื่องมือที่มีอยู่บนแพลตฟอร์ม Steam หรือซอฟต์แวร์ควบคุมโดยผู้ปกครองทั่วไป มักเน้นการจำกัดการเข้าถึง แต่ขาดข้อมูลเชิงลึกที่เป็นรูปธรรมสำหรับใช้ในการสื่อสารพูดคุย ซึ่งสร้าง "ช่องว่างทางข้อมูล" ที่ทำให้ผู้ปกครองไม่สามารถทำความเข้าใจบริบทและพฤติกรรมการเล่นเกมของบุตรหลานได้อย่างแท้จริง

3.1.2 การวิเคราะห์ความต้องการของผู้ใช้

ต้องการเครื่องมือที่สามารถแสดงข้อมูลเวลาการเล่นเกมของบุตรหลานย้อนหลังได้อย่างชัดเจนและเข้าใจง่าย ต้องการระบบที่สามารถแจ้งเตือนได้เมื่อมีการเล่นเกมเกินกว่าเวลาที่ตกลงกันไว้ เพื่อใช้เป็นข้อมูลเริ่มต้นในการพูดคุยอย่างมีเหตุมีผล

3.1.3 แนวทางการแก้ปัญหาและวัตถุประสงค์

จากปัญหาและความต้องการของผู้ใช้ จึงได้กำหนดแนวทางการแก้ไขปัญหาเป็นการพัฒนาเว็บแอปพลิเคชันที่เชื่อมต่อกับ Steam Web API เพื่อดึงข้อมูลเวลาการเล่นเกมมาประมวลผลและแสดงผลในรูปแบบแดชบอร์ด พร้อมทั้งพัฒนาระบบแจ้งเตือนผ่าน Discord Webhook แนวทางดังกล่าวได้ถูกกลั่นกรองและกำหนดเป็นวัตถุประสงค์ของโครงงาน 3 ข้อหลักตามที่ระบุไว้ในหัวข้อ 1.3

3.1.4 การศึกษาความเป็นไปได้ทางเทคนิค

ทำการประเมินความเป็นไปได้ทางเทคนิคของแนวทางการแก้ไขปัญหา โดยศึกษาเอกสารของ Steam Web API และทำการทดสอบโดยใช้ Postman เพื่อยืนยันว่าสามารถดึงข้อมูล playtime_forever ของแต่ละเกมได้จริง และศึกษาเทคโนโลยีที่เกี่ยวข้องในชุดเทคโนโลยี (Technology Stack) ที่เลือกใช้ เช่น Node.js Express.js Supabase และ Chart.js เพื่อให้มั่นใจว่าเทคโนโลยีเหล่านี้สามารถทำงานร่วมกันเพื่อสร้างระบบตามวัตถุประสงค์ที่ตั้งไว้ได้สำเร็จ

3.2 การกำหนดความต้องการของระบบ

ในขั้นตอนนี้เป็นการกำหนดรายละเอียดข้อกำหนดของระบบทั้งในเชิงฟังก์ชันการทำงานและเชิงคุณภาพ เพื่อให้ทีมพัฒนามีเป้าหมายที่ชัดเจนในการออกแบบและพัฒนาระบบต่อไป

3.2.1 ขอบเขตของระบบ

3.2.1.1 ขอบเขตที่ระบบสามารถทำได้

ก) รับค่า Steam ID และ Discord Webhook จากผู้ใช้เพื่อนำไปประมวลผล

ข) ระบบสามารถดึงข้อมูลเวลาการเล่นเกมทั้งหมด (Playtime Forever) จาก Steam Web API และคำนวณเวลาการเล่นย้อนหลังในรอบ 2 สัปดาห์ได้

ค) แสดงผลข้อมูลสรุปและสถิติเวลาการเล่นเกมในรูปแบบแดชบอร์ดและแผนภูมิวงกลมที่เข้าใจง่าย

ง) ส่งข้อความแจ้งเตือนผ่าน Discord Webhook ได้ เมื่อตรวจพบว่ามีเวลาการเล่นสะสมเกินกว่าเกณฑ์ที่ผู้ใช้กำหนด

จ) ระบบสามารถบันทึกข้อมูลการค้นหาและสถิติลงในฐานข้อมูล Supabase โดยผูกกับบัญชีของผู้ใช้

ฉ)สามารถสมัครและล็อกอินเข้าใช้งานได้

3.2.1.2 ขอบเขตที่ระบบไม่สามารถทำได้

ก) ระบบไม่สามารถติดตามเวลาการเล่นเกมแบบเรียลไทม์ได้ ข้อมูลจะอัปเดตเมื่อผู้ใช้ทำการกดตรวจสอบเท่านั้น

ข) ระบบไม่สามารถจำกัดหรือบล็อกการเล่นเกมได้ ทำหน้าที่เพียงวิเคราะห์และแจ้งเตือนเท่านั้น

ค) ระบบไม่รองรับแพลตฟอร์มอื่นนอกเหนือจาก Steam

3.2.2 ฮาร์ดแวร์ที่ใช้กับระบบงาน

3.2.2.1 โน้ตบุ๊ก เอเซอร์ เพรเดเตอร์ ไฮลิออส (Acer Predator Helios)

หน่วยประมวลผลกลาง: Intel Core i5-14500HX

แรม: 16 GB DDR5

พื้นที่จัดเก็บข้อมูล: 512 GB PCIe-4.0 NVMe

จอ: 16″ WUXGA (1920×1200) 165Hz sRGB100%

หน่วยประมวลผลกราฟิก: NVIDIA GeForce RTX 4050 6 GB

3.2.1 ซอร์ฟแวร์ที่ใช้กับระบบงาน

ระบบปฏิบัติการ ไมโครซอฟท์ วินโดวส์ อีเลฟเว่น

1.4.3.1 ระบบปฏิบัติการ ไมโครซอฟท์วินโดวส์ อีเลฟเว่น (Microsoft Window 11)

1.4.3.2 วิชวล สตูดิโอ โค้ด (Visual Studiocode) เป็นเครื่องมือในการพัฒนา

1.4.3.3 เอชทีเอ็มแอล ซีเอสเอส จาวาสคริปต์ สำหรับการพัฒนา ฟรอนท์เอนด์ (UI)

1.4.3.4 โหนด เจเอส (Node.js) สำหรับการรันคำสั่ง JavaScript

1.4.3.5 เอ็กซ์เพรส เจเอส (Express.js) สำหรับการพัฒนาแบ็คเอนด์

1.4.3.6 ซูพาเบส (Supabase) เป็นระบบจัดการฐานข้อมูลของผู้ใช้

1.4.3.7 กิตฮับ (Github) สำหรับจัดเก็บและจัดการโค้ดโปรแกรมของโครงงาน

1.4.3.8 ดิสคอร์ด เว็บฮุก (Discord webhook) สำหรับการส่งแจ้งเตือน

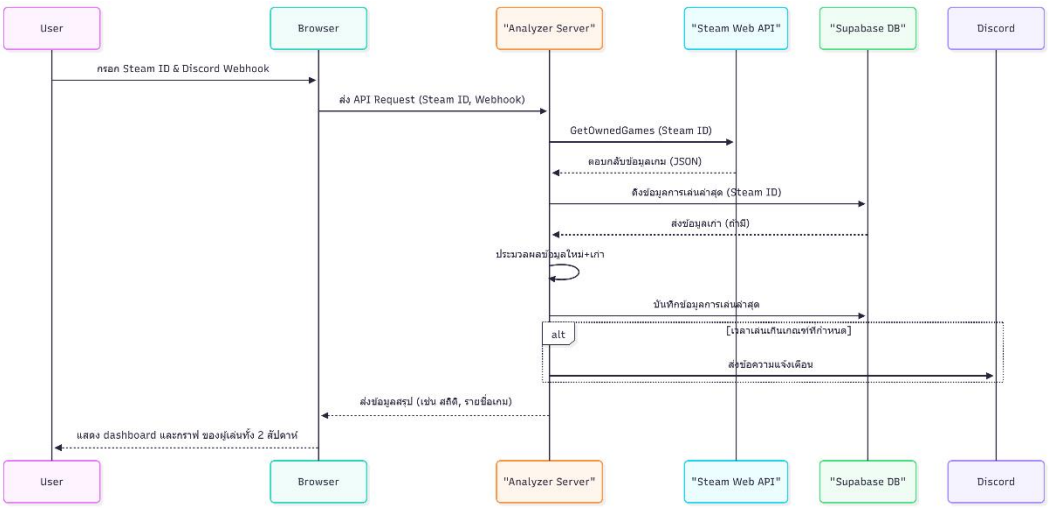
1.4.3.9 ไฟเออร์เบส สตูดิโอ (Firebase Studio) ใช้สำหรับ ดีพลอย เว็บ

1.4.3.10 สตีม เว็บ เอพีไอ (Steam Web API) สำหรับดึงข้อมูลการเล่นเกม

1.4.3.11 โพสต์แมน (Postman) การเทส (Steam Web API) เพื่อทราบว่าสามารถดูข้อมูลของผู้เล่นได้จริง

3.3 การออกแบบระบบ

3.3.1 การออกแบบแผนลำดับการทำงานของระบบ



ภาพที่ 3.1 แผนภาพลำดับ (Sequence Diagram)

3.3.1.1 เริ่มต้น ผู้ใช้งาน (User) เข้าสู่หน้าเว็บแอปพลิเคชันผ่านเบราว์เซอร์

3.3.1.2 กรอกข้อมูล ผู้ใช้กรอก Steam ID และ Discord Webhook ที่ต้องการให้ระบบตรวจสอบและส่งการแจ้งเตือน

3.3.1.3 ส่งคำขอ (API Request) เบราว์เซอร์ทำการส่งข้อมูลที่ผู้ใช้กรอกไปยังเซิร์ฟเวอร์ (Analyzer Server)

3.3.1.4) เรียก Steam API เซิร์ฟเวอร์ได้รับคำขอและนำ Steam ID ที่ได้ไปเรียกใช้ Steam Web API เพื่อดึงข้อมูลการเล่นเกมทั้งหมด

3.3.1.5 ดึงข้อมูลเก่า เซิร์ฟเวอร์ติดต่อไปยังฐานข้อมูล (Supabase DB) เพื่อดึงข้อมูลการเล่นเกมครั้งล่าสุดที่เคยบันทึกไว้ (ถ้ามี)

3.3.1.6 ประมวลผลข้อมูล เซิร์ฟเวอร์นำข้อมูลใหม่ที่ได้จาก Steam Web API มาเปรียบเทียบกับข้อมูลเก่าเพื่อคำนวณหาเวลาเล่นที่เกิดขึ้นจริงในรอบ 2 สัปดาห์

3.3.1.7 บันทึกข้อมูลใหม่ หลังจากประมวลผลเสร็จสิ้น เซิร์ฟเวอร์จะบันทึกข้อมูลการเล่นเกมล่าสุดลงในฐานข้อมูลเพื่อใช้อ้างอิงในอนาคต

3.3.1.8 ตรวจสอบเงื่อนไขและแจ้งเตือน ระบบจะตรวจสอบว่าเวลาเล่นที่คำนวณได้นั้นเกินเกณฑ์ที่กำหนดหรือไม่ หากเกินจะส่งข้อความแจ้งเตือนไปที่ Discord Webhook ของผู้ใช้

3.3.1.9 ส่งข้อมูลกลับไปแสดงผล: เซิร์ฟเวอร์ส่งข้อมูลสถิติที่ประมวลผลเสร็จแล้กลับไปยังเบราว์เซอร์

3.3.1.10แสดงผลบนแดชบอร์ด: เบราว์เซอร์นำข้อมูลที่ได้รับมาแสดงผลในรูปแบบของแดชบอร์ดและกราฟแผนภูมิให้ผู้ใช้เห็น

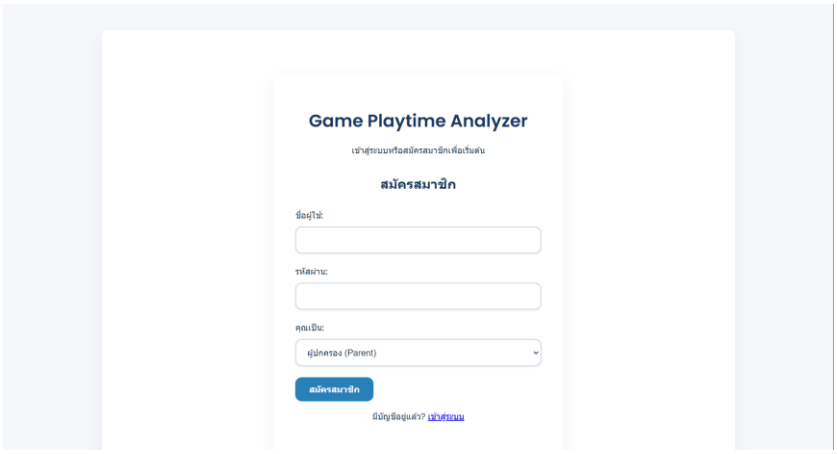
3.3.2 การออกแบบฐานข้อมูล

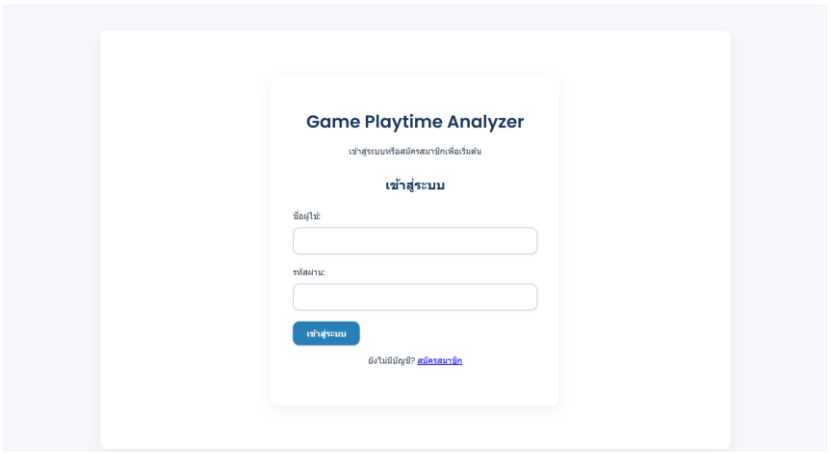
การออกแบบฐานข้อมูลสำหรับระบบ "Game Playtime Analyzer" นี้ได้คำนึงถึงความต้องการในการจัดเก็บข้อมูลผู้ใช้, ข้อมูลโปรไฟล์ Steam, และบันทึกสถิติการเล่นเกม เพื่อให้สามารถขยายระบบในอนาคตได้ จึงได้ออกแบบโครงสร้างฐานข้อมูล (Database Schema) โดยใช้หลักการของ Relational Database บน Supabase (PostgreSQL)

3.3.3 การออกแบบส่วนติดต่อกับผู้ใช้

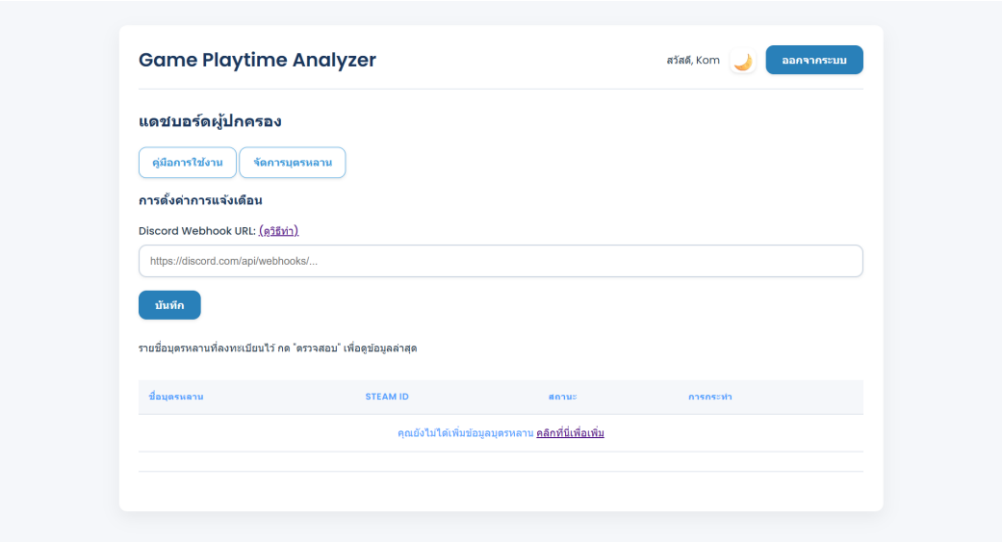
การออกแบบส่วนติดต่อกับผู้ใช้สำหรับระบบ "Game Playtime Analyzer" มุ่งเน้นไปที่ความเรียบง่ายและใช้งานง่าย เพื่อให้ผู้ปกครองซึ่งเป็นผู้ใช้หลักสามารถเข้าถึงและเข้าใจข้อมูลได้อย่างรวดเร็ว โดยมีการออกแบบหน้าจอหลักๆ ที่จำเป็นต่อการใช้งานดังนี้

3.3.3.1 ผลการออกแบบหน้าจอเข้าสู่ระบบ (Login Page) หน้าจอแรกที่ผู้ใช้จะพบคือหน้าจอสำหรับเข้าสู่ระบบและสมัครสมาชิก ซึ่งถูกออกแบบมาเพื่อยืนยันตัวตนของผู้ใช้ก่อนเข้าถึงฟังก์ชันหลัก ทำให้ข้อมูล Steam ID และ Discord Webhook ของผู้ใช้ถูกจัดเก็บอย่างปลอดภัยภายใต้บัญชีของตนเอง

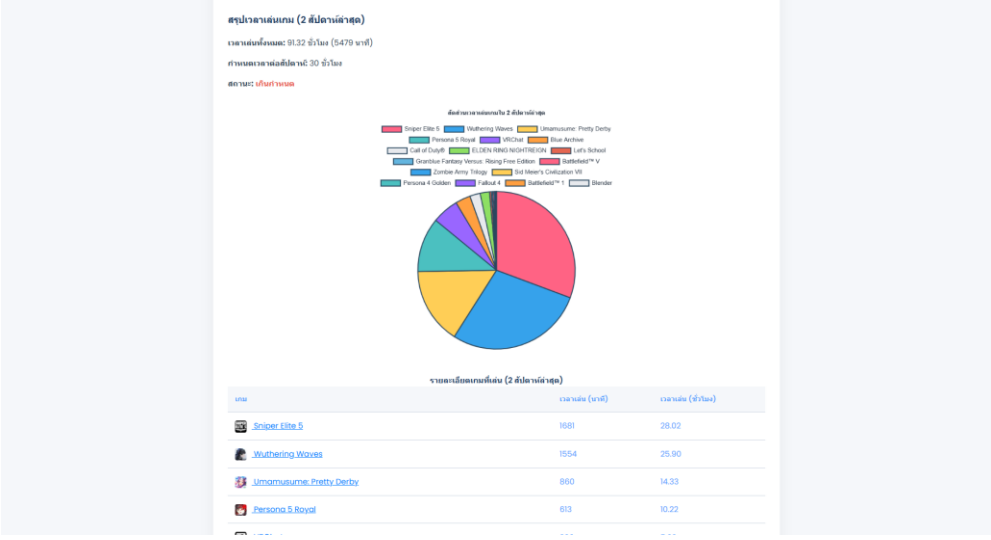




3.3.3.2 ผลการออกแบบหน้าจอหลักและแดชบอร์ด (Main Dashboard Page) หลังจากเข้าสู่ระบบสำเร็จ ผู้ใช้จะพบกับหน้าจอหลักซึ่งเป็นศูนย์กลางของการใช้งาน ประกอบด้วยสองส่วนหลักคือ: ส่วนรับข้อมูล (Input Section): มีช่องสำหรับให้ผู้ปกครองกรอกข้อมูล Steam ID ของบุตรหลาน และ Discord Webhook URL สำหรับรับการแจ้งเตือน พร้อมปุ่ม "ตรวจสอบ" ที่ชัดเจน ส่วนแสดงผล (Display Section): เป็นพื้นที่ว่างสำหรับแสดงผลลัพธ์การวิเคราะห์หลังจากผู้ใช้กดปุ่มตรวจสอบ



3.3.3.3 ผลการออกแบบหน้าจอแสดงผลการวิเคราะห์ (Analysis Display Page) เมื่อผู้ใช้กดปุ่มตรวจสอบและระบบประมวลผลเสร็จสิ้น ข้อมูลจะถูกแสดงผลในส่วนแสดงผลบนหน้าจอหลัก ซึ่งออกแบบมาเพื่อสรุปข้อมูลที่ซับซ้อนให้อยู่ในรูปแบบที่เข้าใจง่าย ประกอบด้วย: ข้อมูลสรุป: ข้อความสรุปเวลาการเล่นเกมทั้งหมดในช่วง 2 สัปดาห์ กราฟแผนวงกลม: แสดงผลข้อมูลเวลาการเล่นของแต่ละเกมโดยใช้ไลบรารี Chart.js เพื่อให้ผู้ปกครองเห็นภาพชัดเจนว่าบุตรหลานใช้เวลากับเกมใดมากที่สุด



ตารางที่ 3.1 users ใช้บันทึกข้อมูลการเล่นเกม ณ เวลาที่ทำการตรวจสอบ เพื่อนำไปคำนวณหาผลต่างในอนาคต

ลำดับ (No)	คุณสมบัติ (Attribute)	คำอธิบาย (Description)	ประเภท (Type)	ประเภทคีย์
1	id	รหัสเฉพาะของข้อมูลที่บันทึก (เรียงลำดับอัตโนมัติ)	BIGSERIAL	PK
2	username	รหัสอ้างอิงถึงโปรไฟล์	VARCHAR	FK
3	email	ID เกมบน Steam	VARCHAR	-
4	password	เวลาเล่นเกมทั้งหมด	TEXT	-
5	steam_id	วันและเวลาที่ทำการบันทึกข้อมูล	VARCHAR	-
6	role	บทบาทของผู้ใช้ในระบบ ('parent' หรือ 'user')	VARCHAR	-
7	discord_webhook_url	URL ของ Discord Webhook สำหรับรับการแจ้งเตือน	TEXT	-
8	created_at	วันและเวลาที่สร้างบัญชี	TIMESTAMPTZ	-

ตารางที่ 3.2 children ใช้สำหรับจัดเก็บข้อมูลของบุตรหลานที่ผู้ปกครองต้องการติดตาม โดยจะมีความสัมพันธ์กับตาราง users

ลำดับ (No)	คุณสมบัติ (Attribute)	คำอธิบาย (Description)	ประเภท (Type)	ประเภทคีย์
1	id	รหัสเฉพาะของข้อมูลบุตรหลาน	BIGINT	PK
2	parent_id	รหัสอ้างอิงถึงผู้ปกครองในตาราง users	VARCHAR	FK
3	child_name	ชื่อของบุตรหลานสำหรับแสดงผล	VARCHAR	-
4	steam_id	รหัส Steam ID ของบุตรหลาน	VARCHAR	-
5	playtime_limit_hours	เกณฑ์เวลาเล่นที่ผู้ปกครองกำหนด (หน่วยเป็นชั่วโมง)	INT	-
6	last_notified_at	วันและเวลาล่าสุดที่มีการส่งแจ้งเตือน	TIMESTAMPTZ	-
7	created_at	วันและเวลาที่เพิ่มข้อมูลบุตรหลาน	TIMESTAMPTZ	-

3.4 การพัฒนาระบบ

3.4.1 วิเคราะห์ระบบ

การศึกษาเบื้องต้นในหัวข้อ 3.1 พบว่าปัญหาหลักคือ "ช่องว่างทางข้อมูล" ที่ทำให้ผู้ปกครองไม่สามารถเข้าใจพฤติกรรมการเล่นเกมของบุตรหลานได้อย่างแท้จริง. ความต้องการของผู้ใช้จึงมุ่งเน้นไปที่เครื่องมือที่สามารถแสดงข้อมูลเวลาเล่นย้อนหลังได้อย่างเข้าใจง่ายและมีการแจ้งเตือนเมื่อเล่นเกินเวลาที่กำหนด. การวิเคราะห์ความเป็นไปได้ทางเทคนิคยืนยันว่า Steam Web API สามารถให้ข้อมูลที่จำเป็นได้จริงผ่านการทดสอบด้วย Postman และชุดเทคโนโลยีที่เลือกใช้มีความสามารถเพียงพอในการสร้างระบบได้สำเร็จ

3.4.2 ออกแบบระบบระบบถูกออกแบบตามสถาปัตยกรรม Client-Server โดยใช้หลักการของ RESTful API เป็นตัวกลางในการสื่อสาร

3.4.2.1 การออกแบบลำดับการทำงาน: ได้กำหนดขั้นตอนการทำงานที่ชัดเจนตั้งแต่ผู้ใช้กรอกข้อมูล, การส่งต่อไปยังเซิร์ฟเวอร์, การเรียกใช้ API ภายนอก, การประมวลผล, การบันทึกข้อมูล, การแจ้งเตือน, และการส่งข้อมูลกลับมาแสดงผล ดังที่แสดงในแผนภาพลำดับ (Sequence Diagram).

3.4.2.2 การออกแบบฐานข้อมูล ได้ออกแบบโครงสร้างฐานข้อมูลบน Supabase ซึ่งเป็นฐานข้อมูลแบบ Relational Database เพื่อรองรับการจัดเก็บข้อมูลผู้ใช้, โปรไฟล์ Steam, และบันทึกสถิติการเล่นเกมได้อย่างเป็นระบบ

3.4.3 พัฒนาระบบ

การพัฒนาระบบแบ่งออกเป็น 2 ส่วนหลัก คือ ส่วนหลังบ้าน (Backend) และส่วนหน้าบ้าน (Frontend) โดยใช้ Visual Studio Code เป็นเครื่องมือหลักในการพัฒนา

3.4.3.1 การพัฒนาส่วนหลังบ้าน (Backend): พัฒนาเซิร์ฟเวอร์ด้วย Node.js และ Express.js เพื่อสร้าง API Endpoint ใช้ไลบรารี Axios ในการส่งคำขอเพื่อดึงข้อมูลจาก Steam Web API จากนั้นเชื่อมต่อกับฐานข้อมูล Supabase เพื่อบันทึกและเปรียบเทียบข้อมูลเวลาเล่น และส่งการแจ้งเตือนผ่าน Discord Webhook เมื่อเข้าเงื่อนไขที่กำหนด

3.4.3.2 การพัฒนาส่วนหน้าบ้าน (Frontend) สร้างโครงสร้างหน้าเว็บด้วย HTML5 และจัดรูปแบบให้สวยงามด้วย CSS3. ใช้ JavaScript ในการจัดการการโต้ตอบกับผู้ใช้, ส่งข้อมูลไปยัง Backend ผ่านการเรียกใช้ API, และนำข้อมูลที่ได้รับกลับมาแสดงผล. ส่วนของการแสดงผลข้อมูลเชิงสถิติจะใช้ไลบรารี Chart.js เพื่อแปลงข้อมูลตัวเลขให้อยู่ในรูปแบบแผนภูมิที่เข้าใจง่าย

3.4.4 ทดสอบระบบ

เพื่อให้มั่นใจว่าระบบทำงานได้ถูกต้องตามที่ออกแบบไว้ ได้มีการทดสอบในหลายระดับ ในเบื้องต้นได้ใช้โปรแกรม Postman เพื่อทดสอบการเรียกใช้ Steam Web API และได้ผลออกมาเป็นการเล่นเกมในช่วง 2 สัปดาห์และหลังจากนั้น ก็ได้ใช้ Steam Web API ต่อเพื่อนำข้อมูลที่ได้จาก user มา จากนั้นจึงทำการทดสอบการทำงานร่วมกันทั้งหมดตั้งแต่หน้าบ้านไปจนถึงหลังบ้าน เพื่อให้แน่ใจว่าผู้ใช้จะได้รับข้อมูลที่ถูกต้อง และระบบการแจ้งเตือนทำงานได้จริงตามเงื่อนไข

บรรณานุกรม

Apple Inc. (2024). Use Screen Time on your iPhone, iPad, or iPod touch. Apple Support. Retrieved October 15, 2025, from https://support.apple.com/en-us/HT208982