Capstone Project Proposal

Timothy Beers

CST-451: Senior Project I

Grand Canyon University

Professor Michael Landreth

Revision: 1.1

2 February 2025

**ABSTRACT**

This project delivers an iPad-based typing game specifically designed to help children with reading disabilities develop touch typing skills. By substituting letters with emojis, the game removes the reading barrier and focuses on cultivating muscle memory for efficient typing. The core gameplay—built using SwiftUI for the interface and SpriteKit for real-time animations—features an “Alien Invasion” mode where users must type the correct emoji to prevent descending UFOs from reaching the bottom of the screen. An adaptive difficulty algorithm continuously adjusts the game’s speed according to the user’s performance, ensuring that learners remain both challenged and encouraged. This local-only design requires no external network calls and securely stores performance metrics on the device, safeguarding user data and simplifying deployment.

Building on a modular architecture, the application lays the groundwork for future expansions such as additional game modes, enhanced statistics, and broader accessibility features. Its integrated stats page tracks time-to-type and error rates, giving users or educators valuable insight into progress over time. By emphasizing adaptability, ease of access, and a positive learning environment, this typing game demonstrates how thoughtfully designed technology can empower people to learn at speeds that were meant for them.

|  |
| --- |
| History and Signoff Sheet |

**Change Record**

|  |  |  |
| --- | --- | --- |
| **Date** | **Author** | **Revision Notes** |
| 2 February 2025 | Timothy Beers | Initial draft for review/discussion |
| 9 August 2025 | Timothy Beers | Update abstract and work breakdown table |
|  |  |  |

|  |
| --- |
| **Overall Instructor Feedback/Comments** |

|  |
| --- |
| **Overall Instructor Feedback/Comments** |

**Integrated Instructor Feedback into Project Documentation**

Yes  No

**Project Approval**

Professor Michael Landreth

**TABLE OF CONTENTS**

Project Overview and Project Objectives 4

Project Scope 5

Project Success Measures 6

Project High-Level Solution 7

Project Controls 8

Project Cost and Schedule 10

Project Overview and Project Objectives

**Problem and Background**

Many educational tools assume that reading is a prerequisite for learning to type. However, children with learning disabilities—such as dyslexia—often face significant challenges with literacy, which can delay or prevent them from learning touch typing. Traditional typing instruction relies heavily on letter recognition, meaning that kids who struggle with reading miss out on the benefits of developing automatic motor patterns essential for efficient typing. This project addresses that gap by developing an iPad-based typing game that removes the reading component from the learning process. By replacing letters with emojis and focusing on the physical act of typing, the game aims to teach and reinforce proper finger placement and movement, enabling children to develop touch typing skills even if they have not yet mastered reading.

**Christian Worldview**

This project is developed with a Christian worldview that emphasizes compassion, support for the vulnerable, and ethical educational practices. Rooted in the belief that every child deserves the opportunity to grow and learn, the application is designed to be inclusive and uplifting. It reflects principles such as kindness and respect for human dignity, ensuring that the game is encouraging rather than punitive. For instance, rather than presenting clear win/loss outcomes that could discourage struggling learners, the game gently increases its difficulty based on the user’s performance, thereby nurturing growth without negative feedback.

**Project Objectives**

1. **Facilitate Touch Typing Without Dependence on Reading**: Develop an engaging iPadOS 18+ application that teaches the fundamental motor patterns of typing using emojis instead of letters, specifically designed for children with reading difficulties.
2. **Implement (at least) One Adaptive Game Mode**: Create an “alien invasion” game mode where UFOs descend with emoji labels. Instead of traditional win/loss feedback, the game will adjust its difficulty progressively as the user improves, ensuring continuous challenge without discouragement.
3. **Reinforce Automatic Typing Patterns**: Focus on helping users learn the muscle memory required for efficient typing, such as reaching for specific keys based solely on their spatial location rather than visual letter cues.
4. **Enhance Accessibility and User Control**: Incorporate features that allow users to customize the display, such as resizing the emoji keyboard and adjusting contrast, to accommodate various physical and cognitive needs.
5. **Track and Analyze Performance Metrics**: Monitor key metrics including accuracy (correct vs. total inputs), time taken per level, and commonly missed emojis. This data will not only help in fine-tuning the game’s adaptive difficulty but also serve as a tool for tracking progress over time.

**Challenges**

1. **Responsive Gameplay and Smooth Animation**: Ensuring that animations and collision detections are fluid on iPadOS 18+ to provide a seamless user experience.
2. **Balancing Adaptivity and Engagement**: Designing the game to progressively increase difficulty in a way that motivates users without causing frustration.
3. **Accessibility Design**: Creating an interface that is intuitive and adaptable for users with varying motor and cognitive challenges.
4. **Data Management**: Implementing effective performance tracking that respects user privacy and accurately reflects progress without overwhelming the user.
5. **Solo Development Constraints**: Managing time and resources efficiently as a single developer to deliver for the initial code deadline.

**Benefits and Opportunities**

The proposed application offers significant benefits by providing an engaging, accessible, and adaptive learning tool for children with reading disabilities, allowing them to develop touch typing skills without the prerequisite of strong literacy. By decoupling typing instruction from reading, the game focuses on reinforcing muscle memory and spatial awareness through non-punitive, progressively challenging gameplay, which not only builds confidence but also encourages continuous improvement. Additionally, the customizable interface ensures that the experience can be tailored to the unique needs of each user. This innovative approach not only has the potential to transform the way children with learning challenges acquire typing skills but also opens up broader opportunities for enhancing early education methods. As the game evolves, its data-driven design and adaptive difficulty system can further personalize learning experiences, making it a versatile tool for educators and therapists alike, and paving the way for future developments that integrate additional game modes and advanced analytics to support individual learning trajectories.

Project Scope

The initial iteration of this project will focus on developing a single functional game mode—the alien invasion—designed to help children with reading disabilities learn touch typing through an engaging, non-punitive approach. This iteration will include a functional emoji keyboard and an animated scene that reacts to keyboard input, where the game adapts its pace by slowing down to give the user sufficient time to input the correct key. Key features for this iteration include tracking the time-to-type for each emoji (with averages calculated per emoji), a control that allows the user to adjust the keyboard size, a settings page for managing difficulty controls, and a dedicated stats page. Out of scope for this phase are additional game modes, extended difficulty settings, advanced stats tracking (beyond time-to-type averages), and functionalities such as a letter-to-emoji slider or Bluetooth keyboard support—these enhancements are planned for future iterations.

|  |  |  |
| --- | --- | --- |
| Stakeholder Name | Role(s) | Responsibilities |
| Timothy Beers | Developer/Project Lead | Design, Coding, Testing, Project Management |
| Kelsie Olds | Occupational Therapist | Provide domain expertise and feedback |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Work Breakdown Structure | | | | | | | | | | |
| ID | Task | Dependencies | Status | Effort Hours | Cost | Start Date | Planned Completion | Estimate to Completion | Actual Completion | Resource |
| 1 | Define Requirements and Create Project Plan | None | Completed | 10 | $0 | 1/31/25 | 2/23/25 | 4 hours | 4 hours | Timothy Beers |
| 2 | Project Design | None | Completed | 8 | $0 | 2/16/25 | 3/23/25 | 8 hours | 6 hours | Timothy Beers |
| 3 | Develop Emoji Keyboard | Requirements Defined | Completed | 8 | $0 | 2/22/25 | 3/9/25 | 8 hours | 10 hours | Timothy Beers |
| 4 | Build Alien Invasion Game Mode | Keyboard Component Complete | Completed | 24 | $0 | 3/10/25 | 3/24/25 | 24 hours | 24 hours | Timothy Beers |
| 5 | Implement Stats Tracking (Time-to-type per emoji) | Game Mode Functional | Completed | 8 | $0 | 3/24/225 | 3/28/25 | 8 hours | 8 hours | Timothy Beers |
| 6 | Create Settings and Stats Pages | Stat tracking and game play implemented | Completed | 8 | $0 | 3/28/25 | 3/30/25 | 8 hours | 6 hours | Timothy Beers |
| 7 | Testing & Debugging | All Components Developed | Completed | 16 | $0 | 3/31/25 | 4/3/25 | 16 hours | 12 hours | Timothy Beers |

Project Success Measures

The success of this project will be determined by a combination of timely delivery and functional performance. Key performance indicators include the successful implementation of the alien invasion game mode with a fully operational emoji keyboard, accurate statistic tracking, responsive adaptive gameplay, and accessible settings and stats pages. The application should operate seamlessly on iPadOS 18+ and provide a smooth, intuitive experience for children with reading difficulties. Additionally, the project’s data tracking must reliably record performance metrics to facilitate adaptive difficulty adjustments. Overall, the project will be considered successful if it is completed within the allocated timeframe and budget and meets the established technical requirements.

|  |
| --- |
| Project Completion Criteria |
| 1 - The alien invasion mode is fully implemented, including an animated scene that responds accurately to emoji keyboard input, with real-time stats tracking. |
| 2 - The application adjusts its pace based on user performance, ensuring a smooth transition that challenges the user without discouraging them. |
| 3 - The emoji keyboard, settings page, and stats page are fully functional, with customizable controls to resize the keyboard. |
| 4 - Time-to-type metrics for each emoji and their averages are recorded correctly and displayed in a clear, user-friendly manner. |
| 5 - The project is developed and refined within the allocated time, with the initial code iteration delivered by April 6th. |

1. Use the template to list the project assumptions and constraints, if applicable. An assumption is an educated guess that a likely condition or circumstance is presumed to be true. A constraint is a limiting condition or circumstance that defines the project boundaries. Assumptions allow the project to succeed. Constraints restrict or limit the project execution.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Assumptions and Constraints | | | | | |
| ID | Description | Comments | Type | Status | Date Entered |
| 1 | Limited development timeframe | The initial iteration is due by April 6th, requiring efficient planning and execution. | Constraint | Active | 2/1/25 |
| 2 | Single Developer Resource | The project is being developed by one person, which may limit the pace and require strict scope management. | Constraint | Active | 2/1/25 |
| 3 | Stability of User Environment | Assumes that users have access to an iPad that meets or exceeds iPadOS 18 requirements and that their usage environment remains stable | Assumption | Active | 2/1/25 |
| 4 | iPadOS 18+ Compatibility | Assumes that all development tools and frameworks used will fully support iPadOS 18+ | Assumption | Active | 2/1/25 |

Project High-Level Solution

**Introduction**

This project addresses the challenge of teaching touch typing skills to children with reading disabilities by decoupling the act of typing from letter recognition. The objective is to create an engaging, iPad-based application that utilizes emojis in place of traditional letters, thereby reinforcing the physical muscle memory necessary for efficient typing. In the primary game mode—the alien invasion—animated UFOs descend on the screen, each labeled with an emoji. Users interact via a custom emoji keyboard that mirrors the QWERTY layout, and their input is matched against the UFOs in a non-punitive, adaptive gameplay environment. The system is designed to track key performance metrics such as the time taken to type each emoji and overall averages, with the gameplay pace adjusting according to the user’s performance. The solution is built for iPadOS 18+, with all interactions optimized for touch input and visual clarity.

**Solution**

The solution is developed primarily using SwiftUI for the user interface and SpriteKit for the game mode animations. Key components include:

1. Emoji Keyboard Component
   1. Implements a QWERTY-style keyboard where each key displays an emoji, positioned corresponding to its traditional letter.
   2. A SwiftUI modifier captures key tap events from the KeyboardObserver and calls an escaping closure passed to the modifier.
   3. Features a draggable control to adjust keyboard size
2. Game Mode (Alien Invasion)
   1. Built with SpriteKit, this module renders an animated scene where UFOs descend with emoji labels.
   2. The View hosting the SpriteKit scene receives key inputs from the KeyboardObserver and validates them against the descending UFOs. A correct input triggers an animation (e.g., destroying a UFO) and records performance metrics.
   3. The game adjusts its speed and timing based on input accuracy and response times, ensuring the user is neither rushed nor discouraged.
3. Performance Tracking Component
   1. This component logs the time-to-type for each emoji and calculates average performance metrics.
   2. Recorded data is passed to the Stats Display in SwiftUI, providing real-time feedback and performance analysis.
4. Settings and Stats Pages
   1. Built using SwiftUI, these pages offer an intuitive interface for users to adjust game difficulty, keyboard dimensions, and other visual settings, as well as review their performance metrics.

Architecture Overview:

The application is organized into two primary layers: the UI layer (built with SwiftUI) and the game logic layer (implemented using SpriteKit). Keyboard interactions are captured by the KeyboardObserver which is utilized by a custom modifier that calls an escaping closure with each tapped key. The view hosting the keyboard and game view acts as a mediator, informing the game scene to trigger relevant logic when a key has been tapped. Simultaneously, performance metrics are logged in the game logic layer and fed back to the UI layer for display in the stats page.

Game Logic Pseudocode Example:

onKeyboardTap(emojiInput):

if emojiInput == targetUFO.emoji:

targetUFO.triggerDestruction()

recordTimeToType(emojiInput)

increaseSpeed()

else:

reduceSpeed()

displayFeedback(“Try again!”)

```

```

Project Controls

1. Use the template to define the risk and list the steps to prevent the risk from occurring or the steps to minimize the chances of it happening. The contingency plan describes alternative solutions to reduce the impact of the risk. An example of a contingency plan is to provide the customer a temporary web server if there are delays in delivery/completion. If the risk has already happened then provide an entry in the issue log.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Risk Management | | | | |
|  | **Risk Probability** | **Risk Impact** |  |  |
| **Event Risk** | **(high, medium, low)** | **Risk Mitigation** | **Contingency Plan** |
| Single Developer Resource/Development Delays | High | High | Plan tasks meticulously; adhere to a strict work schedule and use agile practices to track progress | Prioritize core functionality and consider simplifying non-critical features if delays occur. |
| Integration Challenges between SwiftUI and SpriteKit | Medium | Medium | Build early prototypes to test communication between UI components and the game logic; regularly review framework documentation | Allocate extra time in the schedule for troubleshooting and testing integration issues. |
| Device Compatibility with iPadOS 18+ | Very Low | High | Ensure use of up-to-date frameworks and perform tests on multiple devices to verify compatibility. | If issues arise, adjust configurations or provide a minimum system requirement notice in the app documentation. |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Issues Log | | | | | | | | |
| **ID** | **Description** | **Project Impact** | **Action Plan/Resolution** | **Owner** | **Importance** | **Date Entered** | **Date to Review** | **Date Resolved** |
| 1 | What is the issue? | How will this impact scope, schedule & cost? | How do you intend to deal with this issue? | Who manages this issue? |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Change Control Log | | | | | | | | | |
| **ID** | **Change Description** | **Priority** | **Originator** | **Date Entered** | **Date Assigned** | **Evaluator** | **Status** | **Date of Decision** | **Included in Rev. #** |
| 1 |  |  |  |  |  |  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Roles and Responsibilities | | | |
| Name | Team | Project Role | Responsibility |
| Timothy Beers | Development | Developer/Project Lead | Overall project design, coding, testing, and project management. |
| Kelsie Olds | Advisory | Occupational Therapist | Provide domain expertise and feedback when available. |

Project Cost and Schedule

The project will be executed over a clearly defined timeline with a focus on iterative development, testing, and refinement. There are no financial costs associated with the development of the application. Publishing the finished product to the Apple App Store would require a $99/year developer account.

|  |  |  |  |
| --- | --- | --- | --- |
| **Task** | **Start Date** | **End Date** | **Duration** |
| Define Requirements & Project Plan | 1/31/25 | 2/23/25 | 4 weeks |
| Project Design | 2/16/25 | 3/23/25 | 5 weeks |
| Develop Emoji Keyboard | 2/22/25 | 3/9/25 | 2.5 weeks |
| Build Alien Invasion Game Mode | 3/10/25 | 3/24/25 | 2 weeks |
| Implement Stats Tracking | 3/24/25 | 3/28/25 | 1 week |
| Create Settings and Stats Pages | 3/28/25 | 3/30/25 | 2 days |
| Testing & Debugging | 3/31/25 | 4/3/25 | 4 days |