Software Engineering Department

Braude College of Engineering

Final Project – Phase A (61998)

TalkJourney

CODE: 26-1-D-20

By:

Reuven Itzhakov

Peleg Canfi

Advisor:

Dr. Moshe Sulamy

Link to GitHub:

https://github.com/pelegc49/TalkJourney

Contents

Abstract

Acquiring a new language is a complex process that requires mastering multiple skills, with a specific emphasis on auditory comprehension and speaking abilities. Traditional learning methods often struggle to simulate the immersive environments necessary for learners to practice these skills with confidence. To address this gap, we present “TalkJourney”, a Virtual Reality (VR) application designed to transform language acquisition into an interactive and engaging experience.

TalkJourney utilizes the Communicative Method and Task-Based Learning to simulate real-world scenarios, such as ordering food in a restaurant or interacting with service providers. The application integrates distinct **Gamification** elements to enhance user motivation, including “Quest Objects” for initiating tasks, an Experience Points (XP) system for tracking progress, and unlockable achievements.

A unique feature of the system is the “Transliterator”, a tool designed to aid pronunciation by converting target language text into phonetic representations familiar to the user, prioritizing sound over literal translation. By combining voice detection technology with interactive Non-Player Characters (NPCs), TalkJourney provides immediate feedback and a safe environment for teenagers and adults to practice speaking. This project aims to demonstrate that combining VR immersion with gamified learning significantly improves engagement and practical language proficiency compared to traditional methods.

1. Introduction

In today's increasingly globalized world, the ability to communicate in a second language has become an essential skill for career advancement and cultural integration. However, the journey to fluency is often fraught with challenges. Despite the high demand for language proficiency, traditional learning methods frequently result in low retention rates. While the digital market is saturated with mobile language learning applications, the majority rely heavily on text-based exercises, multiple-choice questions, and rote memorization of vocabulary [4].

Research indicates that while these existing tools are effective for building a "passive vocabulary" (reading and listening), they frequently fail to develop "active communication skills" (speaking and reacting) [5]. Consequently, a significant number of learners abandon these applications due to a lack of engagement and an inability to bridge the gap between digital exercises and real-world conversations. Furthermore, traditional classroom environments often inadvertently foster performance anxiety, where the fear of making grammatical errors inhibits students from practicing openly with native speakers [6].

To address this "Confidence Gap" between knowing grammar and speaking fluently, we propose **TalkJourney**, a Virtual Reality (VR) language learning game. Unlike traditional tools, TalkJourney places the user in a fully immersive, 3D environment where the primary controller is the user's voice. By simulating realistic scenarios - such as ordering food in a restaurant - the system provides a safe, judgment-free environment where learners can practice speaking aloud. The project shifts the pedagogical focus from rote memorization to "Task-Based Learning" (TBL), where language is used as a tool to accomplish specific goals within the virtual world.

The solution integrates advanced game mechanics to sustain user motivation. Key features include an interactive "Bubble System" for vocabulary acquisition, a unique "Transliterator" tool that aids phonetic pronunciation for non-native readers, and dynamic interactions with Non-Player Characters (NPCs). Gamification elements, such as "Quest Objects," Experience Points (XP), and unlockable badges, function to transform repetitive language drills into an engaging adventure, thereby increasing retention and "Time on Task" [3].

Technologically, TalkJourney is developed using the Unity Real-Time Development Platform, optimized for standalone VR headsets. The system relies on a robust architecture that decouples content from logic, utilizing external data loading (JSON) and Voice Recognition services to process user input in real-time. By combining immersive VR technology with proven pedagogical strategies, TalkJourney aims to transform language learning from a passive, repetitive task into an active, engaging experience that builds real-world fluency.

2. Related Work

2.1 The State of Digital Language Learning

The current landscape of Computer-Assisted Language Learning (CALL) [8] is dominated by mobile applications such as **Duolingo** [9] and **Babbel** [10]. These platforms have democratized access to language education through short, repetitive exercises. However, academic reviews indicate a critical limitation: these applications often focus on reading and listening rather than speaking [4]. Users may acquire vocabulary but struggle to bridge the gap to actual conversation due to a lack of pressure-free speaking practice [5].

2.2 Virtual Reality in Education

Virtual Reality (VR) offers a solution to the "immersion deficit" by placing students in context-rich environments. Existing VR solutions, such as **MondlyVR** [11] or **ImmerseMe** [12], utilize 3D conversational agents to simulate real-world interactions. Like traditional classroom role-playing, these applications rely on scripted scenarios to provide structure and ensure learners practice specific grammatical patterns [2]. While scripted practice is a foundational pedagogical tool [1], a common challenge in existing apps is maintaining user engagement during these repetitive tasks.

2.3 Identified Gaps and TalkJourney's Contribution

TalkJourney builds upon the proven methodology of scripted scenarios but addresses specific shortcomings in *how* these scenarios are delivered and experienced:

1. **Enhancing Repetition through Gamification:** While repetition is essential for language retention, it can often feel tedious in standard applications. TalkJourney transforms these necessary scripted interactions (e.g., ordering food) into **Game Quests** initiated by specific "Quest Objects". By integrating an **Experience Points (XP)** system and unlockable **Badges** (e.g., "Food Explorer"), the application reframes repetitive drills as rewarding challenges, sustaining motivation longer than traditional methods [3].
2. **The "Transliterator" - Bridging the Pronunciation Gap:** In many existing language apps, learners act out scripts by reading the target language directly. If the learner cannot read the script (e.g., a new alphabet), the flow of the conversation breaks. TalkJourney addresses this with the **Transliterator**, a unique feature that converts text into a phonetic representation familiar to the user (preserving sound rather than meaning). This allows users to focus entirely on vocal production and confidence within the scripted scenario, rather than struggling with reading comprehension.
3. **Task-Based Context:** TalkJourney moves beyond simple "Listen and Repeat" drills by embedding the script into a **Task-Based Learning (TBL)** framework. The user does not just repeat a sentence; they must use that sentence to trigger a specific reaction from an NPC (e.g., a waiter confirming an order) [2]. This validates the user's speech in real-time, reinforcing the learning loop more effectively than static recording playback.

3. Project Description

This chapter provides a comprehensive technical description of the **TalkJourney** system. It details the system's high-level architecture, core components, and the specific functional and non-functional requirements defined to meet the project's pedagogical goals.

3.1 System Overview

TalkJourney is a Virtual Reality (VR) application developed using the **Unity Real-Time Development Platform**, optimized for standalone VR headsets (e.g., Meta Quest). The system functions as an interactive simulation engine where the user's voice serves as the primary controller.

Technically, the application is structured around a **Speech-to-Action loop**:

1. **Input:** The system captures audio input via the HMD (Head-Mounted Display) microphone.
2. **Processing:** A Voice Recognition Service analyzes the phonetics and matches them against a predefined dictionary of valid keywords (Vocabulary Bank).
3. **Feedback:** The Game Logic Controller triggers immediate visual and auditory feedback based on the confidence score of the recognition.

The user experience is divided into modular “Stages”, starting from isolated vocabulary acquisition using 3D UI elements ("Bubbles") and progressing to complex conversational simulations with NPCs.

3.2 Functional Requirements

The system is designed around core mechanics that facilitate language learning through immersive interaction.

- **Core Mechanics and Interaction:**
    - **Voice Detection:** The primary input method is the user's voice. The system detects spoken words to select answers within “Selection Bubbles”.
    - **Voice Fallback Mechanism:** To ensure a smooth user experience, if the system fails to detect the user's voice three times consecutively (due to accent or hardware issues), a “Bypass” option appears, allowing the user to select the answer manually.
    - **Quest Objects:** Interactive objects in the environment blink to indicate availability. Upon mission completion, these objects change color (Bronze, Silver, Gold) to reflect the user's success rating.
    - **NPCs and Guide:** Users engage in dialogue with Non-Player Characters (NPCs) that utilize “Head Tracking” to maintain eye contact. A floating “Guide” character accompanies the user, providing instructions in their native language and tracking the user's field of view.
- **The “Bubble System”:**

    - **Selection Bubble:** Displays potential answers. Clicking plays the pronunciation, while speaking the text selects it.
    - **Display Bubble:** Contains images or text. Hovering reveals the **Transliterator** (phonetic spelling), and clicking plays the audio.
    - **Sentence Bubble:** Aggregates multiple display bubbles. Interacting with this bubble reads the entire sentence aloud.
- **User Interface (UI) and Feedback:**
    - **Visual Voice Feedback:** A dynamic visual indicator (e.g., a sound wave) responds to input volume, confirming to the user that the microphone is active.
    - **Haptic Feedback:** The VR controllers vibrate slightly when touching UI elements to simulate tactile depth.
    - **Smart Menus:** The main menu allows for "Partial Translation" (text translates on hover) or "Full Translation" modes.
- **Progression and Persistence:**
    - **XP System:** Users gain Experience Points (XP) to level up and unlock new map areas.
    - **Save System:** The system locally saves learned vocabulary, current XP, and unlocked stages to allow continuous play sessions.

3.3 Non-Functional Requirements

To ensure a high-quality VR experience, the system adheres to strict performance and usability constraints.

- **User Experience (UX) and Comfort:**
    - **Height Calibration:** The system supports both seated and standing play by adjusting the in-game camera height.
    - **Motion Sickness Prevention:** A "Vignette" effect (darkening screen edges) is triggered during movement to reduce nausea.
    - **Audio Control:** An "Audio Mixer" allows users to separately adjust Master, Voice, and Ambient volumes, which is critical for focusing on language tasks.
- **Pedagogical Constraints:**
    - **Auditory Focus:** The learning process prioritizes listening and speaking over writing.
    - **Phonetic Accuracy:** The "Transliterator" must preserve the sound of the word rather than its meaning to aid pronunciation.

4. Solution Description

4.1 Software Architecture

The system is built upon the **Unity Game Engine**, utilizing a modular architecture designed to decouple data, logic, and presentation. This separation ensures scalability, maintainability, and adherence to clean code principles. (See Figure 4.1.1 and Figure 4.1.2).

The Data & Infrastructure Layer (Model): This layer manages the loading and storage of content, adhering to the **Open/Closed Principle**.

- **External Data (JSON):** All dialogue strings, menu texts, and phonetic data are decoupled from the code and stored in external JSON files. This allows for dynamic updates to the curriculum without requiring a recompilation of the game code.
- **Asset Management (Addressables):** The system utilizes Unity Addressables for “Lazy Loading”. Heavy assets, such as high-quality audio files, are only loaded into memory when the user enters the relevant location (e.g., the restaurant), optimizing memory usage.

The Logic Layer (Event-Driven System): The core application logic utilizes the **Observer Pattern** to manage game state changes without tight coupling.

- **Event Manager:** A central Event Manager handles global events (e.g., “Sentence Completed”, “Level Finished”). Subsystems such as the *AudioManager*, *ScoreManager*, and *NPCController* listen for these events independently. This prevents “spaghetti code” and ensures that the NPC script does not need direct references to the Audio system.

The Presentation Layer (VR View & Interaction): This layer handles the visual representation and user input. It includes the "Bubble System" and VR controller handling.

- **Optimization Pattern - Object Pooling:** To maintain high frame rates on mobile VR hardware (like Meta Quest), UI elements utilize Object Pooling. Instead of instantiating and destroying bubbles repeatedly (which causes memory lag), inactive bubbles are recycled and returned to a pool [7].

4.2 Game Flow and User Journey

The user experience is structured into a cyclical “Mission Flow” designed to introduce, practice, and test new vocabulary in a safe environment. (See Figure 4.2).

1. Warm-up Phase (Acquisition): The session begins with the **Guide** introducing specific vocabulary items (e.g., “Soup”, “Pizza”, “Water”) relevant to the upcoming scenario. (See Appendix A, Figure A.1).

- **Interaction:** The user sees “Display Bubbles” containing images and text.

- **Action:** Pointing at a bubble reveals the **Transliterator**; speaking the word triggers the Voice Detection system to validate pronunciation.

2. Sentence Building (Integration): Once individual words are mastered, the user enters the "Sentence Completion" phase. (See Appendix A, Figure A.2).

- **Interaction:** A "Sentence Bubble" appears with missing words (e.g., "I would like to order _____").
- **Action:** The user must select the correct word from a bank of options to complete the sentence structure logic.

3. Guided Practice (Simulation): The user is placed in the simulation environment (e.g., a Restaurant) to interact with the **NPC (Waiter)**. This utilizes a Task-Based Learning (TBL) approach where language is used to achieve a specific goal rather than just repetition [2]. The NPC utilizes Head Tracking to maintain eye contact, creating a sense of social pressure and realism. (See Appendix A, Figure A.3).

- **Interaction:** The NPC asks a question (e.g., "What would you like to eat?").

- **Action:** The user must respond using the full sentences learned previously. The NPC utilizes **Head Tracking** to maintain eye contact, creating a sense of social pressure and realism.
- **Feedback:** If the user speaks correctly, the NPC reacts positively. If voice detection fails three times, the "Fallback Mechanism" activates, allowing manual selection.

4. Summary (Reinforcement): The session concludes with a performance review. (See Appendix A, Figure A.4).

- **Interaction:** The Guide reappears to display the results.
- **Action:** The user receives **XP** and unlocks **Badges** (e.g., "Food Explorer") based on their pronunciation accuracy and task completion speed, leveraging gamification to improve retention [3].

4.3 Technical Implementation Highlights

To support this flow, several specific technical solutions are implemented:

- **Spatial Audio:** 3D sound sources are used for NPCs so that audio volume and directionality change based on the user's physical position, enhancing immersion.
- **Addressables (Lazy Loading):** To prevent long load times, heavy audio assets (like restaurant ambience or food-specific dialogue) are only loaded into memory when the specific scene is entered.
- **In-Game Debug Console:** A hidden canvas allows developers to view logs inside the VR headset during runtime, facilitating easier debugging of microphone issues.

Figure 4.1.1: TalkJourney System Use Case Diagram.
This diagram illustrates the primary actors (User/Learner) and their interactions with the system, including core gameplay loops (Mission, Level) and auxiliary systems (Settings, Progress Tracking).

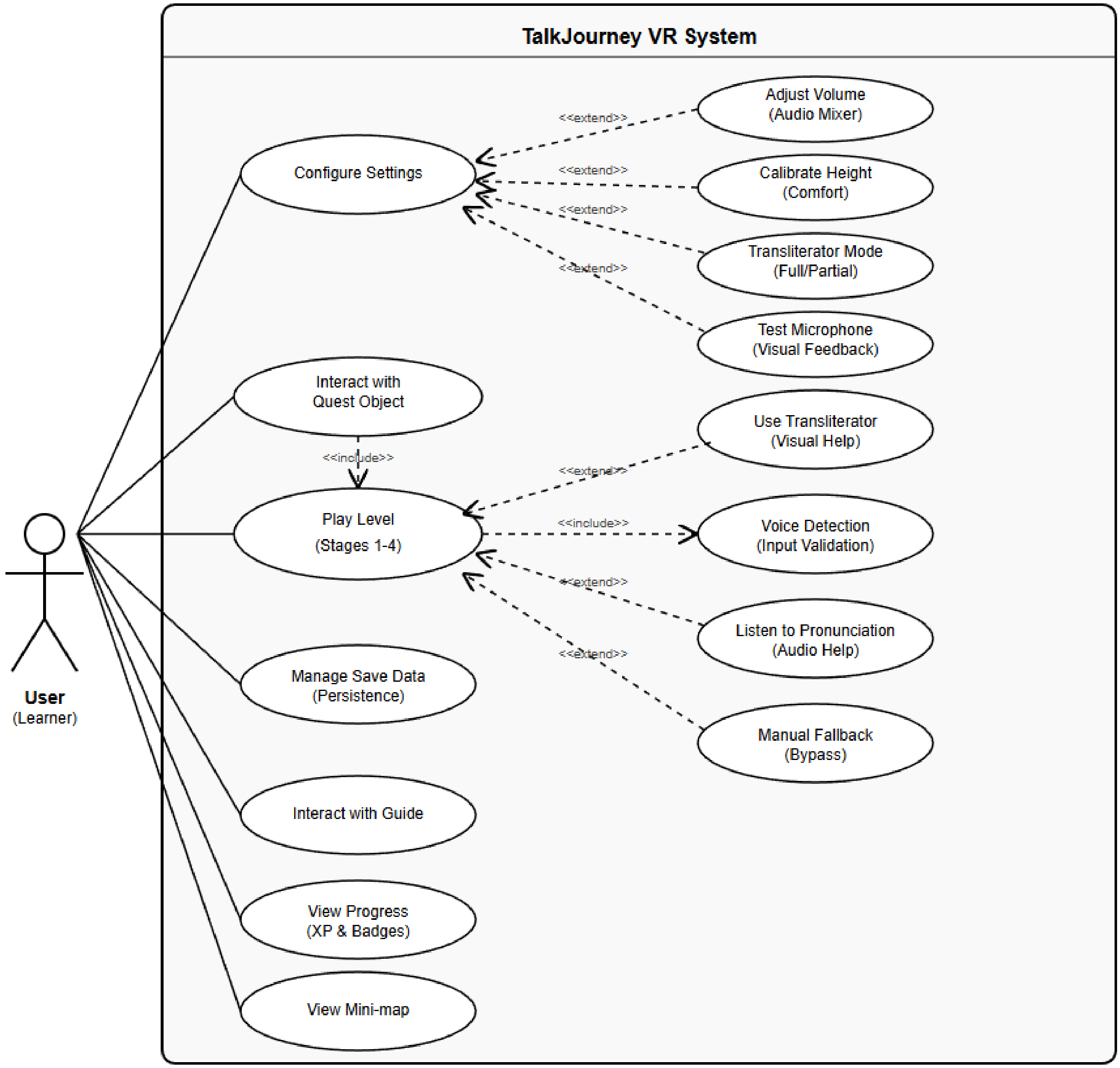


Figure 4.1.2: High-Level System Architecture.
The diagram displays the modular separation of the system into three distinct layers: Presentation (Unity/VR), Logic (C# Controllers), and Data (JSON/Persistence). It highlights the use of the Observer Pattern for event handling and Object Pooling for performance optimization.

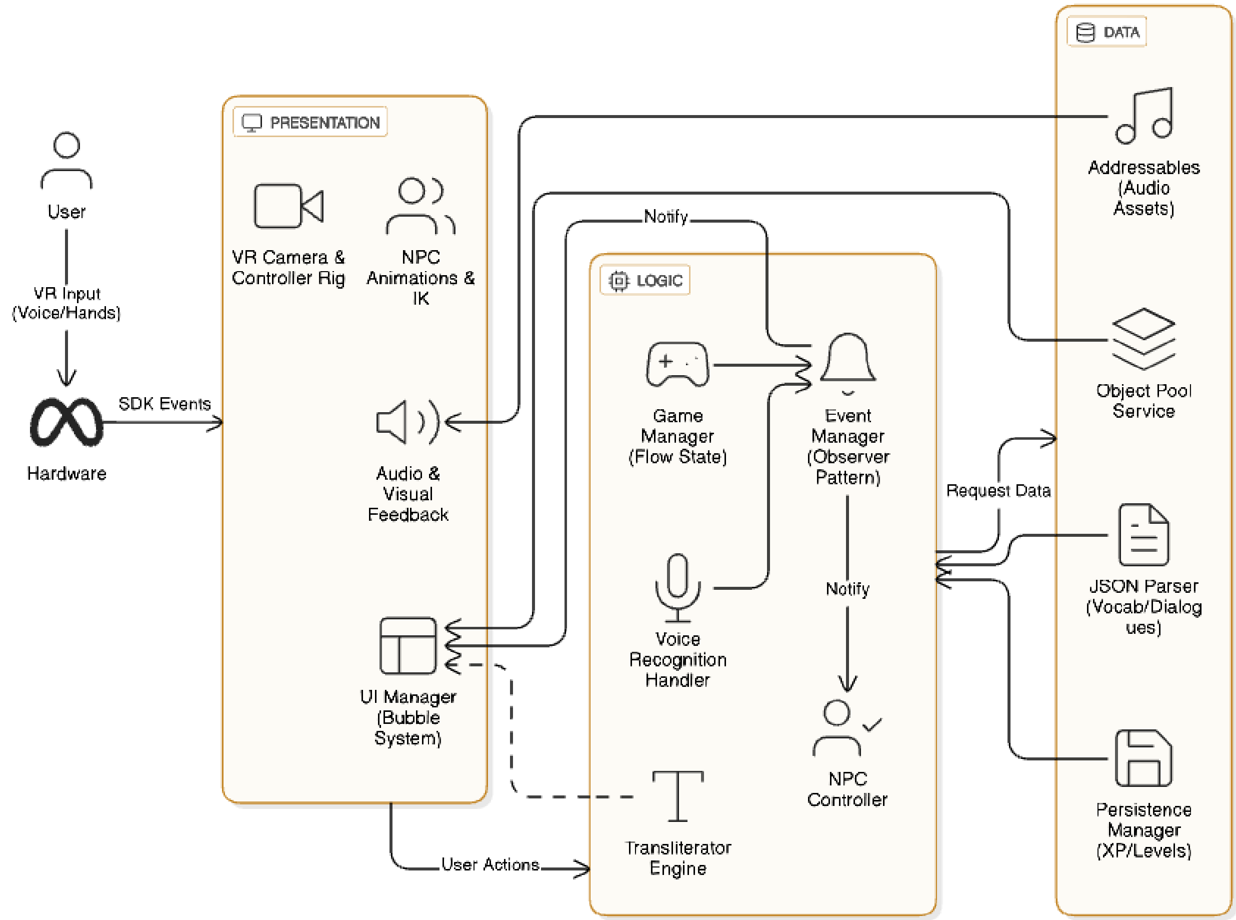
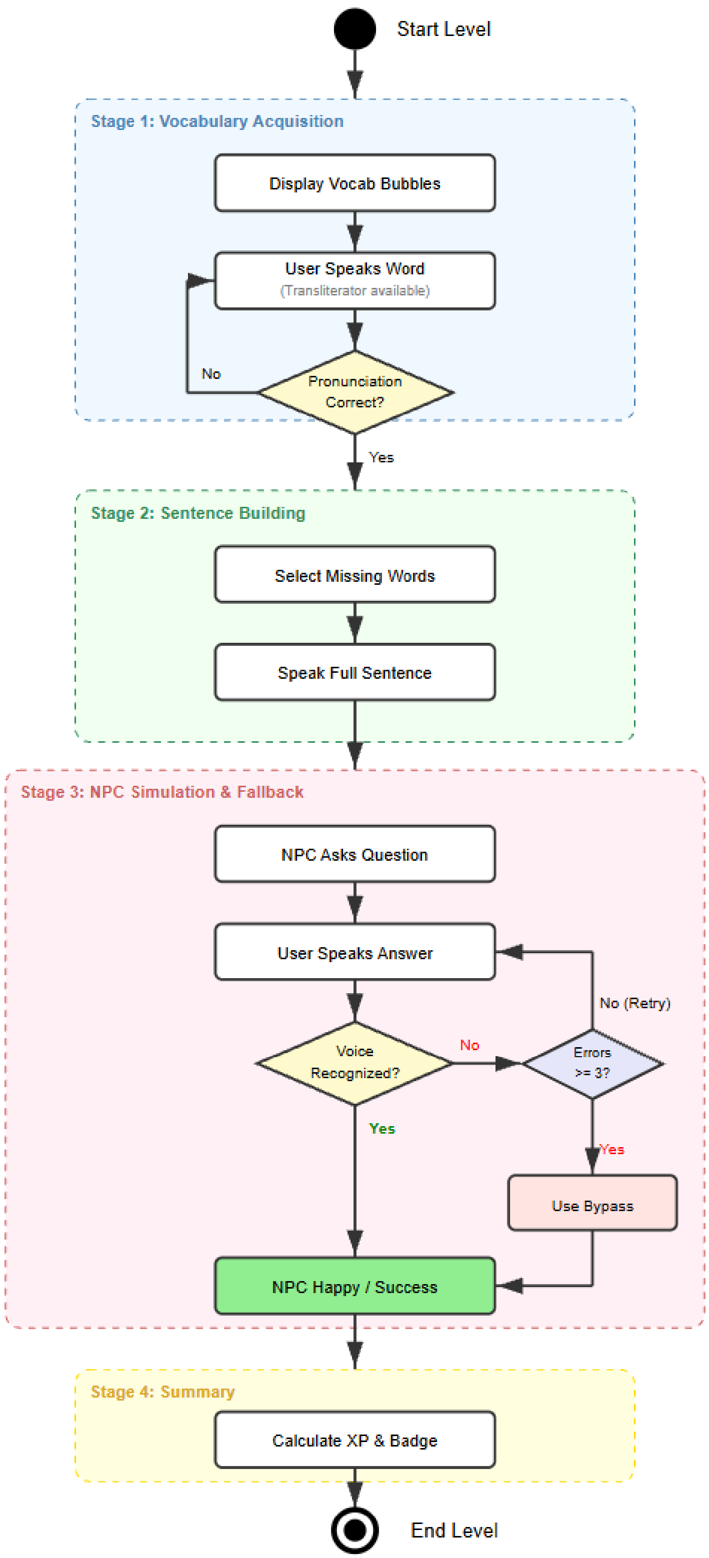


Figure 4.2: Game Flow Activity Diagram.
A flowchart depicting the logical progression of a typical game level. It details the stages from vocabulary acquisition to the NPC simulation, highlighting the decision nodes for voice validation and the error-handling loop that triggers the fallback mechanism.



5. Success Metrics

To evaluate the effectiveness of the TalkJourney system, we have defined specific quantitative criteria. The project will be considered successful if the following metrics are met during the testing phase:

5.1 Pedagogical Effectiveness

- **Vocabulary Retention Rate:** Users should be able to recall and use the taught vocabulary (e.g., "Soup", "Pizza") within the dialogue context.
  - *Success Criterion:* 80% of users successfully complete the "Guided Practice" phase (ordering food) without triggering the "Bypass" (manual selection) mechanism more than once per session.
- **Pronunciation Improvement:** The system facilitates active speaking practice.
  - *Success Criterion:* The Voice Detection system successfully recognizes the user's spoken input on the first or second attempt in at least 85% of interactions.

5.2 Technical Performance

- **Frame Rate Stability:** To prevent motion sickness and ensure immersion, the application must maintain a stable frame rate on the target hardware (Meta Quest III).
  - *Success Criterion:* The application maintains a minimum of **40 FPS** (Frames Per Second) during 95% of gameplay, with no frame drops below 20 FPS during ASR (Automatic Speech Recognition) interpretation, in addition ensuring smooth performance via techniques like Object Pooling [7].
- **Latency:** Interaction with UI elements (Bubbles) must feel instantaneous.
  - *Success Criterion:* The time delay between speaking a command and the system providing visual feedback (e.g., bubble selection) is less than **500ms**.

5.3 Usability and User Experience

- **Task Completion Time:** The interface should be intuitive enough for new users to navigate without external help.
  - *Success Criterion:* New users can complete the "Warm-up" tutorial phase in under **3 minutes**.
- **Navigation Comfort:** The VR movement system should not cause discomfort.
  - *Success Criterion:* Less than 10% of test users report feelings of nausea or disorientation after a 10-minute continuous play session, validating the implementation of the "Vignette" comfort mode to reduce VR anxiety [6].

5.4 Speech Recognition Reliability

- **Fallback Rate:** To ensure the primary voice mechanic is working and users aren't forced to use the manual backup often.

    - *Success Criterion:* The "Bypass" (manual selection) option is triggered in less than **5%** of all total interactions, proving that the voice recognition is the primary viable input method.
- **False Rejection Rate (FRR):** To measure frustration levels where the user speaks correctly but the system fails.
    - *Success Criterion:* The system produces a "False Negative" (rejecting a correct pronunciation) in less than **10%** of valid speech attempts.

6. Testing Plan

To ensure the reliability, performance, and pedagogical effectiveness of TalkJourney, we will implement a comprehensive testing strategy. This plan covers four primary layers: Unit Testing, Integration Testing, Usability Testing, Performance Testing, and Acceptance Testing.

6.1 Unit Testing

These tests focus on verifying individual scripts and logic blocks in isolation, ensuring the underlying architecture is robust before running the full game.

- **Data Loading:** Verify that the External Data Loader correctly parses the JSON files containing dialogues and vocabulary without errors or missing fields.
- **Object Pooling:** Test the BubblePoolManager to ensure bubbles are correctly recycled (set to inactive) rather than destroyed, and that new bubbles are successfully retrieved from the pool when requested.
- **Save/Load System:** Verify that the PersistenceManager correctly serializes and deserializes local JSON data, specifically checking that XP, Level, Word Dictionary, Quest Objects, and Unlocked Stages are retained between sessions.

6.2 Integration Testing

These tests verify that different subsystems (Voice, Audio, Game Logic) communicate correctly via the Event System.

- **Voice-to-Action Response:** Verify that when the Voice Recognition engine detects a correct keyword (e.g., "Pizza"), the Selection Bubble triggers the "Selected" state and the NPC triggers the "Happy" animation.
- **Fallback Trigger:** Simulate three consecutive failed voice recognitions to ensure the "Bypass" mode (manual selection) enables automatically, preventing player frustration.
- **Event Propagation:** Verify that the OnSentenceCompleted event correctly triggers three simultaneous independent actions: playing the success sound, updating the XP score, and playing the NPC animation.

6.3 Usability and VR Experience Testing

Given the nature of Virtual Reality, testing the user experience is critical to prevent physical discomfort and ensure accessibility.

- **Motion Sickness Check:** Testers will perform rapid movements in the environment to verify that the "Vignette" effect (darkening screen edges) activates correctly to reduce nausea.
- **Height Calibration:** Verify that the game correctly adjusts the camera height for both seated and standing players, ensuring UI elements remain at eye level.
- **Feedback Perception:** Confirm that users notice the "Visual Voice Feedback" (sound wave icon) when speaking and feel the "Haptic Feedback" (vibration) when pointing at UI elements.

6.4 Performance Testing

VR applications on mobile chipsets (like Meta Quest III) require strict resource management.

- **Frame Rate Stability:** We will use the Unity Profiler to ensure the application maintains above **40 FPS**. We will specifically test the "Restaurant" scene to verify that the Addressables system successfully loads heavy audio assets only upon entry and unloads them upon exit.
- **Memory Usage:** Monitor RAM usage during the instantiation of the "Bubble System" to confirm that the Object Pool prevents memory spikes during dialogue sequences.

6.5 Acceptance Testing

The following Acceptance Tests verify that the system meets the functional and non-functional requirements defined in the Project Description. Each test corresponds to a specific feature implemented in the VR environment.

Test. ID	Feature / Requirement	Test Procedure	Acceptance Criteria (Pass Condition)
AT-01	**Voice Fallback Mechanism**	Simulate 3 consecutive failed attempts to recognize a spoken word (e.g., by speaking gibberish or silence).	The "Bypass" mode (manual selection) enables immediately after the 3rd failure, allowing the user to progress.
AT-02	**Quest Object States**	1. Identify a blinking object. 2. Complete the associated mission with a high score. 3. Observe the object's material.	1. The object blinks to indicate availability. 2. The object changes color to **Gold** (or Silver/Bronze based on score) upon completion.
AT-03	**NPC Head Tracking**	Move the player character (camera) left and right while standing in front of an NPC.	The NPC's head rotates smoothly to maintain eye contact with the player at all times.
AT-04	**Guide Behavior**	Enter a scene with the "Guide" character enabled. Move the player's head.	The Guide floats within the user's field of view and provides instructions in the user's defined **native language**.
AT-05	**Bubble Interaction**	1. Point at a "Display Bubble". 2. Click the bubble.	1. The **Transliterator** text appears on hover. 2. The correct audio pronunciation plays upon clicking.
AT-06	**Selection Bubble**	Speak the text displayed inside a "Selection Bubble".	The system recognizes the phrase and automatically "selects" the bubble without physical controller input.

Test. ID	Feature / Requirement	Test Procedure	Acceptance Criteria (Pass Condition)
AT-07	**Visual Voice Feedback**	Speak into the microphone with varying volume levels.	A visual indicator (e.g., sound wave) animates in real-time.
AT-08	**Haptic Feedback**	Move the VR controller to point at a UI button or Bubble.	The controller emits a short, distinct vibration to simulate tactile contact.
AT-09	**Smart Menus**	Open the main menu and hover over a button in "Partial Translation" mode.	The button text changes from the target language to the native language while hovering, and reverts when the hand moves away.
AT-10	**Persistence (Save System)**	1. Earn XP and unlock a level. 2. Quit the application. 3. Relaunch the application.	The XP count and Unlocked Level status remain exactly as they were before quitting.
AT-11	**Height Calibration**	Toggle between "Seated" and "Standing" modes in settings.	The in-game camera height adjusts so that UI elements and NPCs remain at comfortable eye level for the chosen posture.
AT-12	**Motion Sickness Prevention**	Use the joystick to move the character forward.	A "Vignette" effect (black faders) appears on the edges of the screen during movement and disappears when stopping.
AT-13	**Audio Mixer**	Go to Settings and lower the "Ambience" volume slider to 0 while keeping "Voice" at 100%.	Background noise (e.g., restaurant chatter) becomes silent, but the NPC's dialogue remains audible and clear.

References

[1] W. Yan and V. L. Lowell, "Design and Evaluation of Task-Based Role-Play Speaking Activities in a VR Environment for Authentic Learning", 2024. Available: https://www.researchgate.net/publication/385998676_Design_and_Evaluation_of_Task-Based_Role-Play_Speaking_Activities_in_a_VR_Environment_for_Authentic_Learning

[2] M. Christoforou, T. Thrasher and A. Ijiri, "Technology-mediated Task-Based Language Teaching in a Virtual Reality learning environment", 2024. Available: https://www.researchgate.net/publication/387233381_Technology-mediated_Task-Based_Language_Teaching_in_a_Virtual_Reality_learning_environment

[3] P. K. Kumar and C. Vairavan, "The Impact of Gamification on Motivation and Retention in Language Learning: An Experimental Study", 2025. Available: https://www.researchgate.net/publication/386068382_The_Impact_of_Gamification_on_Motivation_and_Retention_in_Language_Learning_An_Experimental_Study_Using_a_Gamified_Language_Learning_Application

[4] M. Mihaylova, S. Gorin, T. P Reber and N. Rothen, "A Meta-Analysis on Mobile-Assisted Language Learning Applications: Benefits and Risks", 2022. Available: https://www.researchgate.net/publication/363627180_A_Meta-Analysis_on_Mobile-Assisted_Language_Learning_Applications_Benefits_and_Risks

[5] E. Ayedoun, Y. Hayashi and K. Seta, "A Conversational Agent to Encourage Willingness to Communicate in the Context of English as a Foreign Language", 2015. Available: https://www.researchgate.net/publication/281412908_A_Conversational_Agent_to_Encourage_Willingness_to_Communicate_in_the_Context_of_English_as_a_Foreign_Language

[6] S. Park, D. Carlisle, M. Gillies, and X. Pan, "Reducing Foreign Language Anxiety with Virtual Reality", 2025. Available: https://www.researchgate.net/publication/370779036_Reducing_Foreign_Language_Anxiety_with_Virtual_Reality

[7] "Implementing Object Pooling in Unity for Performance", 2025. Available: https://www.wayline.io/blog/how-to-download-unity-unity-hub-unity-editors-and-unity-download-archive

[8] Dr. M. Thenmozhi, A. T. Selvan, and N. Kumari, " Computer Assisted Language Learning", 2023. Available: https://www.researchgate.net/publication/372371825_Computer_Assisted_Language_Learning

[9] "Duolingo – Learn Languages Online", 2025. Available: https://www.duolingo.com/

[10] "Babbel – Language Learning App", 2025. Available: https://www.babbel.com/

[11] "Mondly VR – Learn Languages in Virtual Reality", 2025. Available: https://www.mondly.com/vr

[12] "ImmerseMe – Virtual Reality Language Learning", 2025. Available: https://immerseme.co/

Appendix A: User Interface Visualizations

Figure A.1: Vocabulary Acquisition & Transliterator.
The "Display Bubble" introduces new vocabulary ("Pizza"). The "Transliterator" tooltip appears above, offering phonetic guidance ("Pi-tza"), while the visual voice indicator on the left confirms microphone readiness.

Figure A.2: Sentence Integration Interface.
The user constructs a full sentence by selecting the correct word from the bottom "Selection Bubbles" to fill the gap in the upper "Sentence Bubble", integrating vocabulary into grammatical structures.

Figure A.3: NPC Interaction
The simulation stage featuring a Waiter NPC who uses head-tracking to maintain eye contact.

Figure A.4: Session Summary & Gamification Feedback.
The session concludes with a performance review presented by the virtual Guide. Users receive Experience Points (XP) and unlockable badges (e.g., "Food Explorer") to reinforce learning and leverage gamification for improved retention.