# Exercises

Each exercise follows the same basic UI look as the main screen. Each exercise also uses the same correct/incorrect toolbar at the top of the screen to indicate where the user is in the exercise.

### Important Concepts:

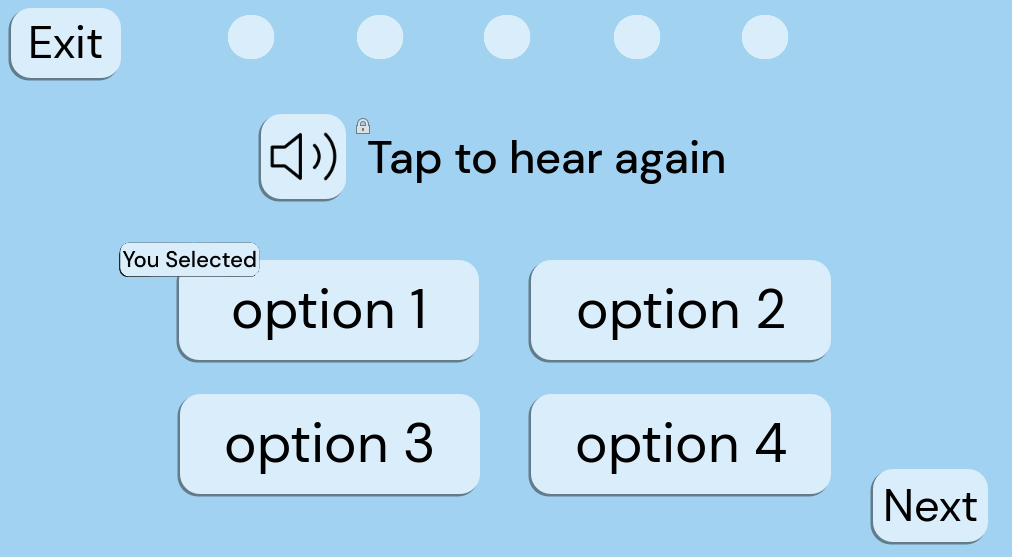
* The button labels must be easy to read and do not overlap with any other exercise elements.
* The exercise screens must not be too cluttered, as that hinders the learning experience for the users.

## Exercise One

This exercise is a four-word choice type of exercise. There are a predetermined number of questions in the exercise, and each question contains different words. The questions will play a sound (which can be replayed) and will give four options as to what the word could be. Then the user can select which word matches the sound. This allows the user to test their hearing for similar-sounding words.It is made up of two files, ExerciseOne.gd and ExerciseOne.tscn.

### In-app Description:

Let’s practice hearing and differentiating between similar-sounding words!



Example of Exercise 1

### 1. ExerciseOne.gd

#### Overview:

This script controls the behavior and logic of the first interactive exercise within the application. It manages user interactions with buttons, word generation, scoring, and progression through rounds.

#### Key Features:

* Word generation and display on buttons.
* User interaction handling for selecting words.
* Logic for determining correct answers.
* Handling game progression, including rounds and completion.

#### Usage:

1. Attach this script to a Node2D in your scene hierarchy.
2. Ensure that the necessary UI elements are properly linked in the scene.
3. Customize the script variables, such as the maximum number of rounds and the word lists.
4. Run the scene to allow users to interact with the exercise.

#### Functions:

* \_ready(): Initializes the exercise when the node enters the scene tree.
* onButton1Pressed(), onButton2Pressed(), onButton3Pressed(), onButton4Pressed(): Handlers for button presses.
* onSoundButtonPressed(): Handles sound button press to play correct word audio.
* onNextButtonPressed(): Handles the next button press to progress or end the game.
* onExitButtonPressed(): Handles exit button press to return to the previous screen.
* buttonLogic(buttonNum): Logic for processing button presses and determining correctness.
* generateWords(): Generates new word options for the buttons.
* checkCorrect(pressedWord, correctWord): Checks if the pressed word matches the correct word.
* changeNextStar(correctIncorrect, numRounds): Updates the visual indicator for correct and incorrect answers.
* buttonColorChange(colorBool): Changes button colors based on correctness and game state.
* gameDone(): Handles completion of the game and additional functionalities (not fully implemented).

### 2. ExerciseOne.tscn

#### Overview:

This scene file defines the layout and hierarchy of UI elements for Exercise One. It includes buttons, text elements, and visual indicators necessary for user interaction.

#### Components:

* Background: Node2D serving as the background for the exercise.
* ColorRect: ColorRect providing a colored background for text.
* TopText: Label displaying text instructions or feedback.
* Button: Node2D containing multiple Button nodes for user selection.
* SoundButton, ExitButton, NextButton: Buttons for specific actions within the exercise.
* Node2D: Container for visual elements like stars and indicators.
* Star1 to Star5: Visual indicators representing user performance.
* RoundTimer: Timer node for tracking time per round.
* SelectedIndicator: Visual indicator showing the selected word.

#### Usage:

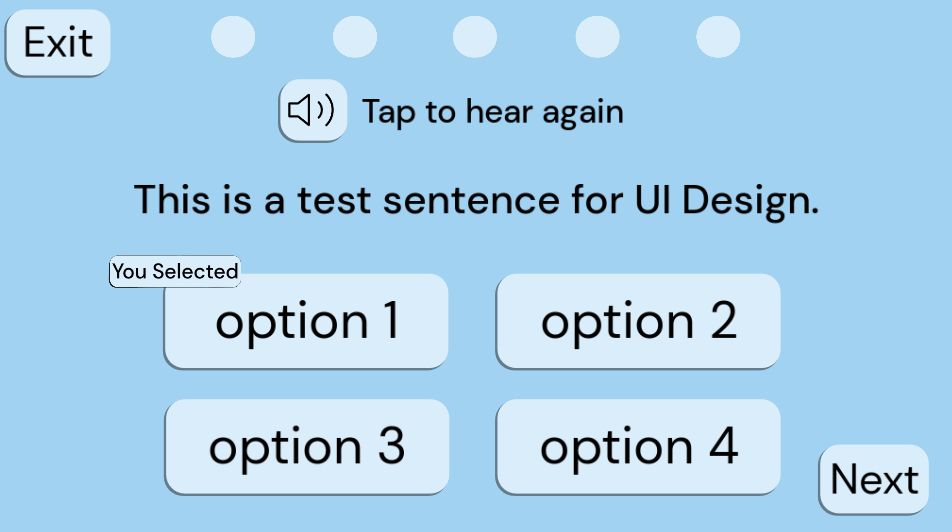
1. Open this scene file in the Godot editor.
2. Customize UI elements and their properties as needed.
3. Ensure that signals from buttons are connected to the appropriate methods in the script (ExerciseOne.gd).
4. Run the scene to allow users to interact with the exercise.
5. This documentation provides an overview of the provided files and their functionalities. Further customization and integration may be required based on specific project requirements.

## Exercise Two

This exercise is a sentence type of exercise. There are a predetermined number of questions in the exercise, and each question contains different words in sentences. The questions will play a sentence sound (which can be replayed) along with displaying the sentence minus a word and will give options as to what the missing word could be. Then the user can select which word matches the missing word’s sound in the sentence. This allows the user to test their hearing for similar-sounding words in sentence structures.

### In-app Description:

Let’s practice hearing and differentiating between similar-sounding words in sentences!



Example of Exercise 2

### Exercise\_two.gd

#### Description:

This GDScript file contains the logic for Exercise Two in the application. It manages the user interface elements, rounds, correct answers, and other game functionalities.

#### Variables:

* buttonOne: Reference to the first button node.
* buttonTwo: Reference to the second button node.
* buttonThree: Reference to the third button node.
* buttonFour: Reference to the fourth button node.
* nextButton: Reference to the next button node.
* starOne to starFive: References to star nodes representing the user's progress.
* roundTimer: Reference to the timer node.
* topText: Reference to the text node displaying instructions or feedback.
* selectedIndicator: Reference to the indicator node displaying the selected button.
* display\_sentence: Reference to the node displaying the current sentence to the user.

#### Other Variables:

* numRounds: Current number of rounds.
* maxNumRounds: Maximum number of rounds.
* correctWord: Current correct word.
* replayMode: Boolean indicating whether the game is in replay mode.
* soundIcon: Preloaded image for the sound icon.
* numCorrect: Number of correct answers.
* sentence: The current sentence being displayed to the user.

### Exercise\_two.tscn

#### Description:

This scene file defines the layout and structure of Exercise Two's user interface. It includes buttons, timer, feedback elements, and other visual components required for the game.

#### Nodes:

* Button: Container node for buttons.
  + Button1 to Button4: Individual buttons for user interaction.
  + NextButton: Button to proceed to the next round.
* Node2D: Container node for visual elements.
  + Star1 to Star5: Star nodes representing the user's progress.
  + Background: Container node for background elements.
    - hear\_again\_text: Node displaying instructions or feedback text.
    - display\_sentence: Node displaying the current sentence to the user.
  + SelectedIndicator: Node indicating the selected button.
* RoundTimer: Timer node for tracking round time.

This scene file serves as the visual representation of Exercise Two and is linked with the corresponding GDScript file (exercise\_two.gd) to provide functionality.

## Pre-Exercise Screens

### Overview

The pre-exercise screens for exercise one and exercise two allow the user to select the word list they want to practice, a text to speech voice, if they want background noise as well as its volume level.

### Node Structure:

Both screens contain nodes for UI elements like buttons, labels, and containers.

The first exercise screen has a more complex structure with option buttons and a VBoxContainer, while the second exercise screen is simpler, focusing on navigation.

### External Resources:

Both screens use external resources for fonts and icon textures.

### Styles and Themes:

The first exercise screen defines custom styles for buttons and backgrounds using StyleBoxFlat resources, while the second exercise screen does not.

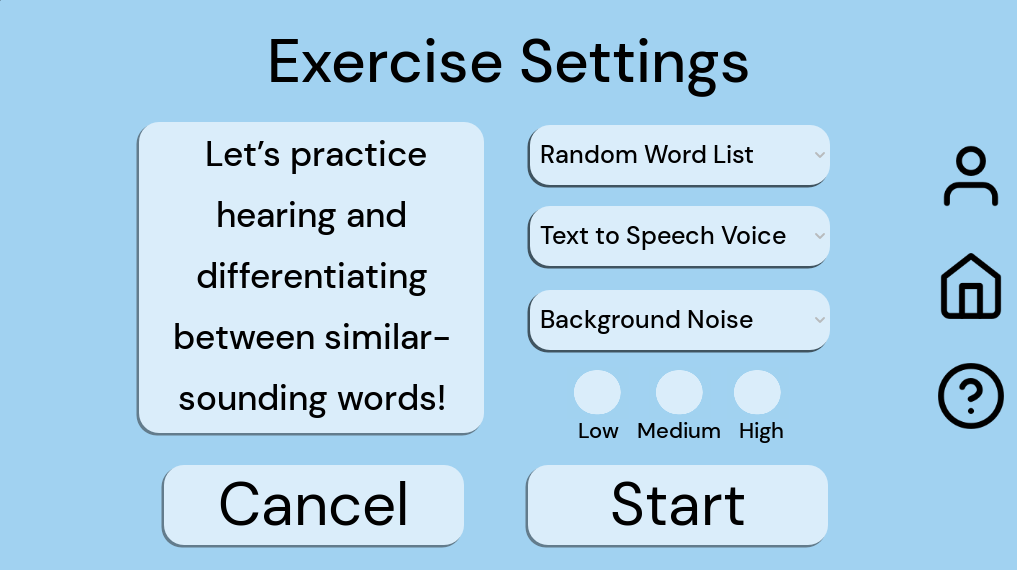
### Functionality:

Both screens provide basic functionality for navigation and button actions

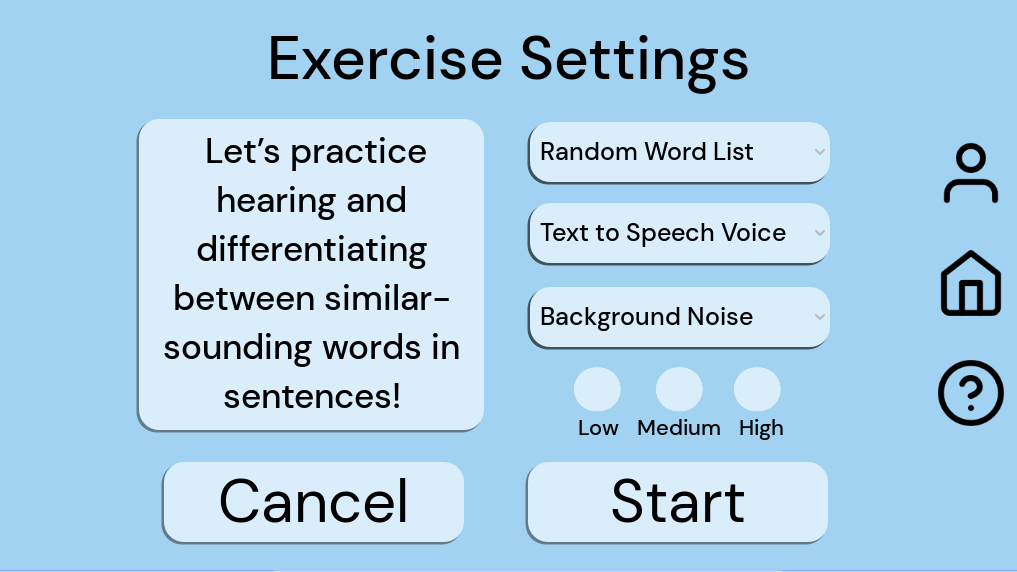
### UI Components:

Both screens utilize buttons, labels, and other basic UI elements.

The first exercise screen includes additional components like option buttons, making it more feature-rich compared to the second exercise screen.



Exercise One Pre-Exercise Screen



Exercise Two Pre-Exercise Screen

## Post-Exercise Screen

### GDScene File: post\_exercise\_screen.gd

This GDScript file defines the behavior and functionality of the post-exercise screen in a game or application. It extends the Node2D class and includes methods for handling button presses and initializing UI elements with data. Here's a breakdown of its components:

#### Variables:

* backscene: This variable stores a reference to the previous scene, likely for navigation purposes.

#### Methods:

* \_on\_cancel\_pressed(): Handles the press event of the "Cancel" button, changing the scene to pre\_exercise\_screen.tscn.
* \_on\_done\_pressed(): Manages the press event of the "Done" button, changing the scene to main\_menu.tscn.
* \_on\_profile\_pressed(): Deals with the press event of the "Profile" button, changing the scene to profile.tscn.
* \_on\_help\_pressed(): Manages the press event of the "Help" button, changing the scene to help.tscn.
* \_on\_home\_pressed(): Handles the press event of the "Home" button, changing the scene to main\_menu.tscn.
* \_ready(): Initializes UI elements with data such as score, word list text, background volume level, and accuracy.

### GDScene File: post\_exercise\_screen.tscn

This Godot scene file defines the layout and appearance of the post-exercise screen. It includes various UI elements such as buttons, labels, and containers. Here's an overview of its structure:

#### Nodes:

* Exercise 1 Settings: A node possibly holding settings related to Exercise 1.
* Background: Node containing visual elements like color rectangles.
* UI: Node containing UI elements like labels and buttons.
* Buttons: Node containing button elements.
* Globals: Node possibly storing global variables or references.

#### Connections:

* Establishes signal connections between UI elements and methods defined in the associated script file.

#### UI Elements:

* Labels: Display information such as game title, score, accuracy, word list practiced, and background sound level.
* Buttons: Perform actions like canceling, finishing, accessing profiles, getting help, and returning to the main menu.

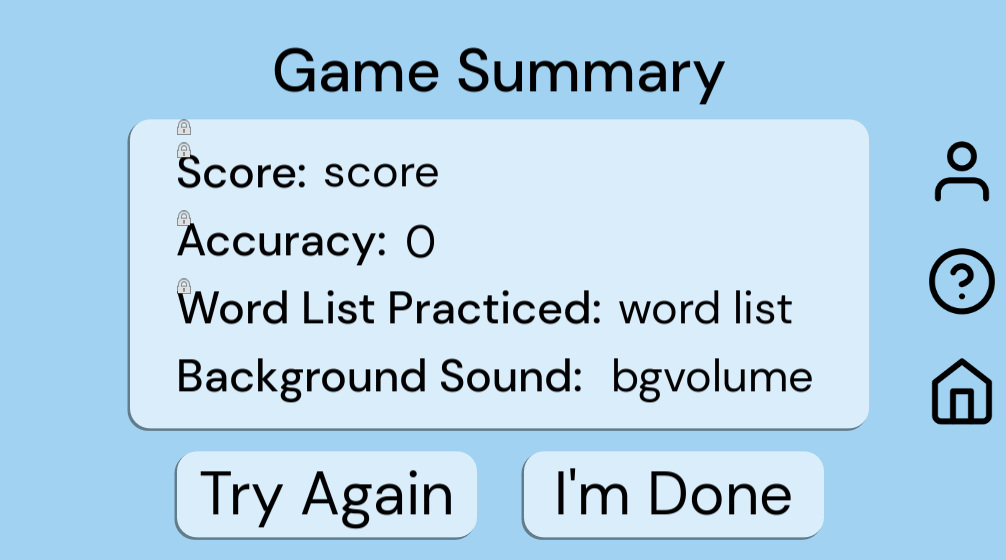
#### Styling:

* Style boxes are defined to customize the appearance of UI elements, including colors, corner radius, and shadows.

#### Layout:

* UI elements are organized within containers such as VBoxContainer for proper alignment and spacing.

This combined documentation provides insight into both the functionality and visual layout of the post-exercise screen in the game or application. It describes how the script interacts with UI elements to manage user interactions and scene transitions.



Example of post exercise screen