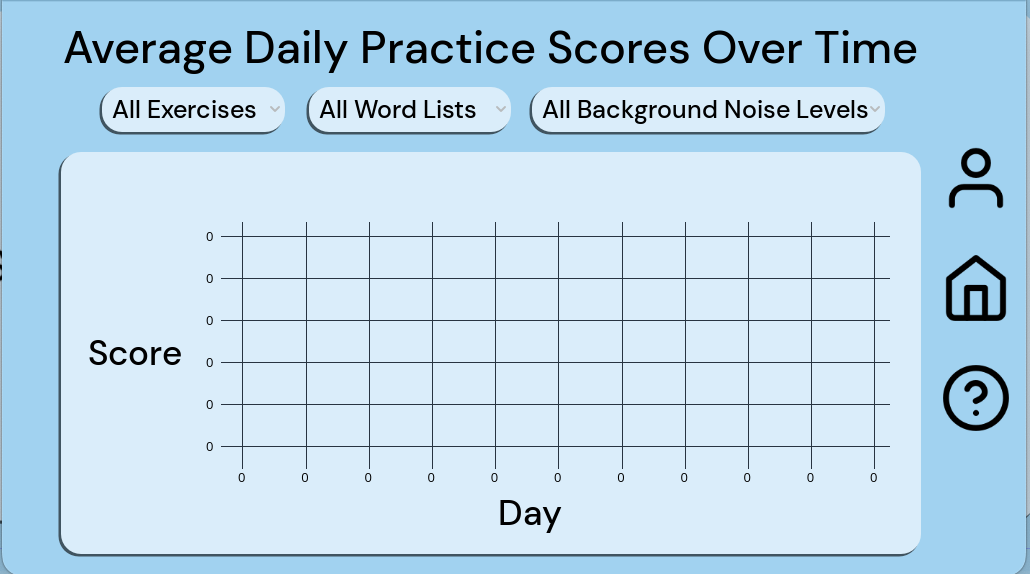
# Statistics

## Overview

The statistics appear on two types of screens: the screen at the end of an exercise and the statistics page. Everything for the exercise post-screen is in the exercise documentation. This section will detail the data used in the statistics as well as the code used in the statistics screen.



Statistics Screen as of 3/10/24

## Code Documentation

## Overview:

The Statistics Testing Module is a component of a larger application developed in the Godot Engine. It is designed to manage statistical data entries, allowing users to search through existing entries and generate random data for testing purposes.

## Files:

### StatisticsTesting.gd

#### Purpose:

* This GDScript file defines the functionality for managing statistical data entries.

#### Functionality:

* It contains methods for searching through existing entries and generating random entries.
* It establishes connections between UI elements and methods to trigger actions based on user interaction.

#### Structure:

* The script extends the Node2D class.
* It includes methods such as searchEntries() and generateRandomEntries() for performing specific actions.
* Connections between button signals and methods are established within this file.

### StatisticsTestingScene.tscn

#### Purpose:

* This scene file defines the structure and layout of the user interface (UI) for the Statistics Testing Module.

#### Components:

##### Nodes:

* + It contains various nodes representing different UI elements such as buttons, labels, and input fields.

##### Connections:

* + Signal connections between UI elements and methods defined in the StatisticsTesting.gd script are set up within this file.

#### Layout:

* The scene is organized hierarchically with nodes nested within each other to create the desired layout for the user interface.
* UI elements are positioned and styled according to the requirements of the module.

## Functionality:

### Searching Entries:

* Users can input search parameters such as date, background noise, sound, and exercise.
* Upon triggering the search action, the module retrieves matching entries from the database and displays them in the UI.
* Entries are displayed in a tabular format, showing date and corresponding scores.

### Generating Random Entries:

* Users can trigger the generation of random entries by clicking on a designated button.
* The module adds a specified number of random entries to the database, which can be used for testing purposes.

Note: Some functionality related to additional entry fields such as time, background noise, sound, and exercise is currently commented out, suggesting that it may be a work in progress or placeholder code for future implementation.

## Scoring System

The scoring system of Auditory Ace uses two metrics when evaluating the user’s score; accuracy and time. A score will be given for each individual word or phrase answered by the user which ranges between 0-1000. Incorrect guesses will result in zero points given. Applying background noise rewards the user with a score multiplier:

| **Background Noise Level** | **Score Multiplier** | **Max Score (For each entry)** |
| --- | --- | --- |
| None | 1.0x | 5000 |
| Low | 1.2x | 6000 |
| Medium | 1.4x | 7000 |
| High | 1.6x | 8000 |

Each exercise has a pre-determined grace period that allows the user to get full credit if they answer within a specific time frame. The scoring algorithm uses the following expression:

**# Subtract from 1000 the minimum between the following expressions:**

**# 5 \* sqrt(time after grace period)**

**# 150 \* log10(time after grace period)**

**score = 1000 - min((5 \* sqrt(timeTaken - gracePeriod)), (150 \* (log(timeTaken - gracePeriod) / log(10))))**

The following table provides sample scores based on time:

| Time (after grace period) | Score (no multipliers) |
| --- | --- |
| 0s | 1000 |
| 1s | 842 |
| 5s | 646 |
| 10s | 500 |
| 16s | 368 |
| 25s | 340 |
| 45s | 302 |
| 90s | 257 |
| 5 minutes | 178 |

# 