# *At a Glance*

Eye Tracking Project Documentation

**Penn State Erie, The Behrend College**
HARE Research Group
School of Humanities & Social Sciences
School of Engineering
Office of Research & Outreach

## Documentation Table of Contents

## Project Overview

At a Glance is a multifaceted project that incorporates the essential components of eye tracking, user interface design, application creation, and data collection. The following are the 4 overarching goals of the project:

1. Design a user interface that simplifies the calibration process, launches applications, and saves useful and reliable data.
2. Create a video recording of the user experience, including an overlay of critical eye tracking information.
3. Save and export calibration accuracy, eye tracking, and application data in a usable and meaningful way to researchers.
4. Design applications that fully utilize and incorporate the functionality of the eye tracker.

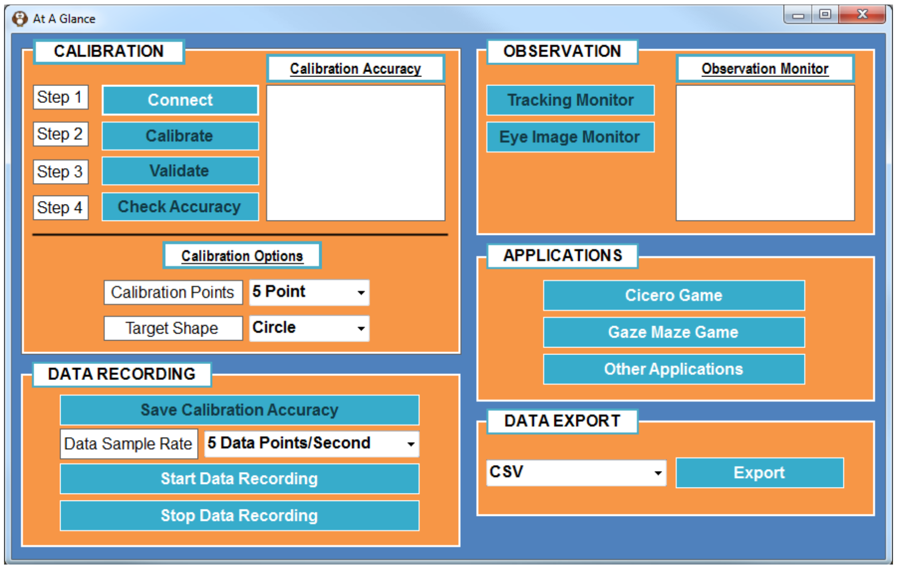
In the following sections below, these goals will be expanded upon, including specific details of implementation and functionality.

## Implementation/Project in Action

### Eye Tracking User Interface

#### Overview

The eye tracking user interface was created in order to incorporate all of the essential elements of the design including a simplified calibration panel, data recording panel, observation panel, applications panel, and data export panel. A screenshot of the graphic user interface (Figure 1) and a detailed description of each panel are below.



Calibration Points: 5 Point
Target Shape: Circle
Data Sample Rate: 5 Data Points/Second
CSV

*Figure 1-Eye Tracking GUI*

## Calibration Panel Functionality

The calibration panel includes the steps needed to set up the eye tracker and calibrate the system with the user's eyes. It is a 4 step process:

1. *"Connect"*- In this step, the eye tracker connects to the server and is ready for the calibration process.
    1. After the connection has been established, the user will see a representation of their eyes or "Tracking Monitor" in the Observation Monitor window of the Observation panel.
    2. Once the "Connect" button is clicked and the connection has been established, the "Calibrate" button will be enabled.
2. *"Calibrate"*- When the user clicks "Calibrate", the calibration window opens, the user hits the "spacebar" on the keyboard, and looks at the calibration points on the screen. There are 2 default settings for this process; the calibration points (set at 5 points) and the target shape set as a circle.
    1. The user may change the default calibration options in the "Calibration Options" portion of the panel.
        1. The "Calibration Points" can be changed from the default 5 points to 1, 2, 5, or 9 points.
        2. The Target Shape" can be changed from the default circle to a cross.
    2. Once the "Calibrate" button is clicked, the "Validate" button will be enabled.
3. *"Validate"*- This will load a new window in which the user looks at 4 points to validate the calibration.
    1. Once the "Validate" button is clicked, the "Check Accuracy" button and the "Save Calibration Accuracy" button (from the "Data Recording Panel") will be enabled.
4. *"Check Accuracy"*- Clicking this button will open a new window with an image of the validation screen. This is a larger version of what is shown in the "Calibration Accuracy" window.

## Data Recording Panel Functionality

The data recording panel includes the options for the types of information that the user can save including:

- *"Save Calibration Accuracy"*- This opens a new directory window where the user can save an image (in png format) of the calibration accuracy.
- *"Data Sample Rate"*- This is a dropdown menu that allows the user to choose how many eye tracking data points per second they would like to record.
    - The default is set at 5 data points/second, with options to change it to 1, 7, 10, or 20 data points/second.
- *"Start Data Recording"*- Clicking this button opens a new window that allows the user to input a user ID and click "OK" to start recording data. This will start recording two sets of eye tracking data:
    - The first set of eye tracking data is a video (in avi format and codec information) of the user's screen with a red dot overlaid where the user is looking. This red dot will allow the user to see where they were looking at

on the screen during the video recording. An example screenshot of this is in Figure 2 below.

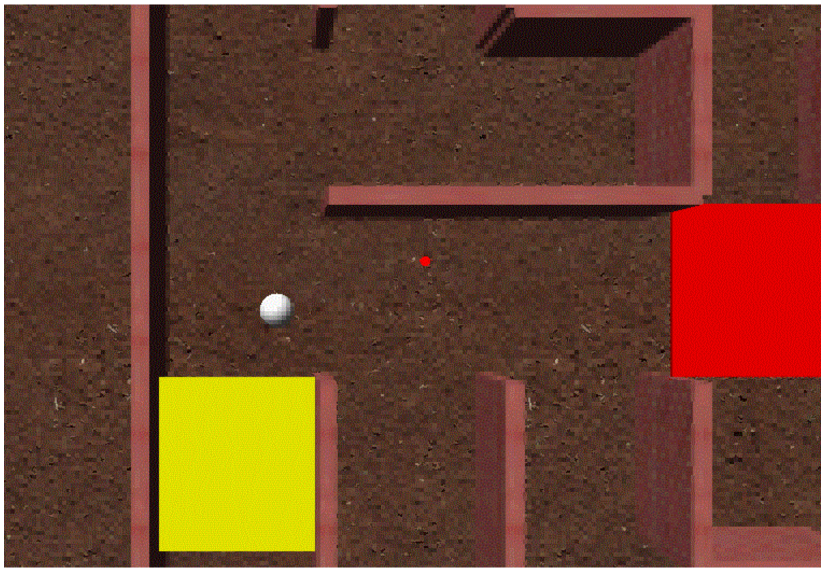


***Figure 2-Eye Tracking Video w/ Red Dot Overlay***

  - The second set of eye tracking data is a recording of the raw eye tracking data. Specifics of this can be found in the Database Design & Implementation section below.
- *"Stop Data Recording"*- Clicking this button opens a new window that lets the user know that the data has been saved to the database.

## Observation Panel Functionality

The observation panel allows the user to monitor the status of their eyes in relation to the eye tracker. There are two main features of this panel:

- *"Tracking Monitor"*- This button loads two white dots (that represents the user's eyes) on a black background in the "Observation Monitor" window. It will also show arrows on the sides of the screen that allows for more information and better eye placement.
- *"Eye Image Monitor"*- This button loads the black and white camera view of the user's face in the "Observation Monitor" window.

## Applications Panel Functionality

This panel houses the applications that can be loaded from the user interface. Specifics of the background and functionality of these applications can be found in the Application Design & Implementation section of this document below. There are currently 3 buttons that function under this panel:

- *"Cicero Game"*- This loads a new window that runs the "Cicero Game" application.

- *"Gaze Maze Game"*- This loads a new window that runs the "Gaze Maze Game" application.
- *"Other Applications"*- Clicking this button opens a directory window where the user can load any exe file.

### Database Panel Functionality

This panel allows the user to export the eye tracking data to a csv file. Specifics of the background and functionality of how the database was designed, what is saved, and the functionality of it can be found in the Database Design & Implementation section of this document below.

1. *"Export"*- This will export the database into a .csv file. Each table will have its own csv file generate with the field names place in the first row and each unique entry placed in subsequent rows. There are currently 3 separate tables in the database: the raw eye tracking information, the Cicero game data, and the Gaze Maze game data.

# Applications Design & Implementation

## Cicero Dicit Fac Hoc (Simon Says) Game

### Overview

In this game, the player must remember a sequence of quadrants, then correctly input the sequence. In a twist, the user will use their eyes to repeat the sequence instead of a mouse or keyboard response. Using the eye-tracker, the player will look for 2 seconds at the quadrant corresponding to the color or pattern. This will act as a "click" similar to a mouse click and they can then move on to the next quadrant that is part of the sequence. There are a variety of options to choose from to change the type of gameplay. Instead of a normal increment of one addition to the sequence after each correct answer, the player has the option to increase that amount for a more difficult challenge. The player may also choose some different styles/patterns of play. Instead of choosing normal mode, there is a color mode and a position mode which are different than the classic game. In each of these two modes, the quadrants will switch around. In color mode, the player will have to remember the correct color of the quadrants that lit up, regardless of position. In position mode, the player will have to remember the correct position of the quadrants that lit up, regardless of color. An example of the menu screen is in Figure 3 below.

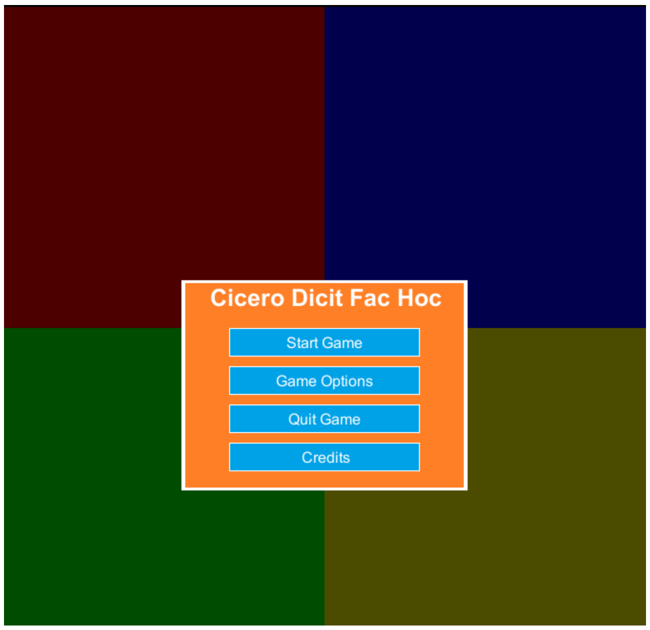


***Figure 3-Cicero Game Menu***

## Functionality

1. *"Start Game"*- This option will begin the game.
2. *"Game Options"*-
    - *"Network Options"*-
        - *"Server IP"*- The IP of the server can be changed here. The server is auto-detected, so this will not need to be changed under normal circumstances.
    - *"Increment"*- This is a difficulty setting. This will determine the amount of new additions to the sequence the player must remember after each correct answer. It can increment by 1, 2, 3, or 4 per round.
        - *"Pattern"*-
        - *"Normal"*- In this play setting, the normal/classic version of the game is presented.
        - *"Color Only"*- In this play setting, the quadrants will be switched around. The player will have to remember the correct colors, regardless of position.
        - *"Position Only"*- In this play setting, the quadrants will be switched around. The player will have to remember the correct position, regardless of color.
    - *"Quit Game*"- This option will close the application.
    - *"Credits"*- This will bring up a short credits page, with information on the creators and sponsors of this game.

## Gaze Maze Game

### Overview

In this maze game, the player controls a ball, navigating it from start to finish with only their eyes. The eye tracker tracks where the player is looking, and that is used as the input. The ball follows the player's line of vision, and the farther that line of vision is from the ball, the faster the ball will roll toward where the player is looking. The game will track how long the player takes to complete the maze, the number of collisions the ball made with the walls of the maze, and the route taken to get to the finish. Below are two screenshots of the game. The image to the left (Figure 4) is the menu screen. The image to the right (Figure 5) is an example of what a maze looks like.

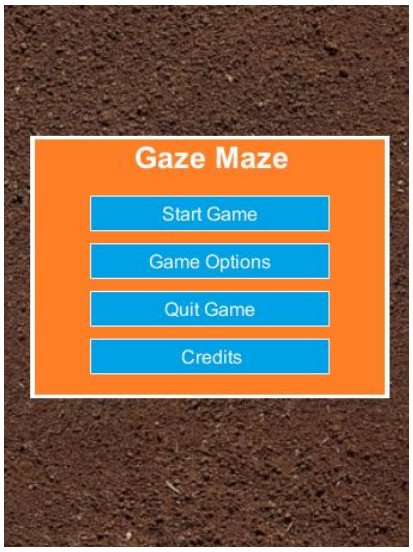


***Figure 4-Gaze Maze Menu***

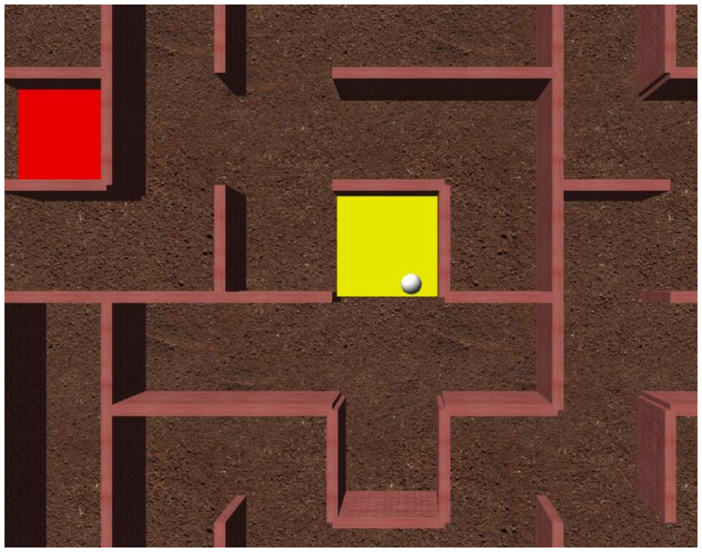


***Figure 5-Gaze Maze Game Play Example***

### Functionality

- *"Start Game"*- This option will begin the game.
- *"Game Options"*-
  - *"Network Options"*-
    - *"Server IP"*- The IP of the server can be changed here. The server is auto-detected, so this will not need to be changed under normal circumstances.
  - *"Difficulty"*- The game's difficulty can be changed here. The more difficult the game becomes, the larger the maze gets and the more zoomed in the camera.
- *"Quit Game"*- This option will close the application.
- *"Credits"*- This will bring up a short credits page, with information on the creators and sponsors of this game.

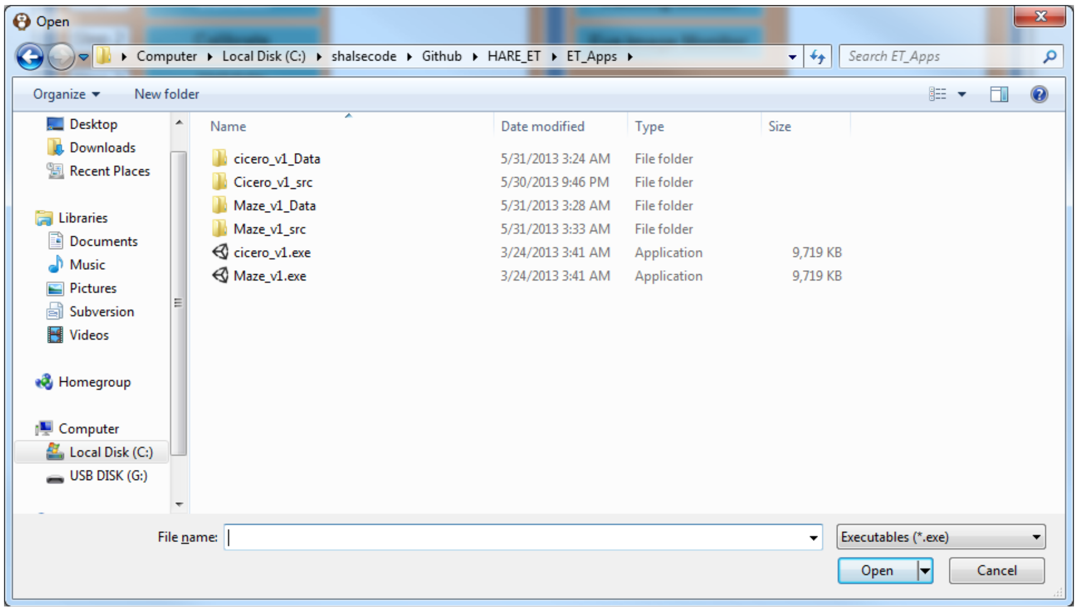
## Other Applications

### Overview

This portion of the eye tracking user interface is designed to allow any additional application to be loaded. This section will allow the user to load most exe files. Once the file is loaded the eye tracker interface will take over mouse control. It will also start recording the eye tracking data such as gazeX and gazeY positions.

### Functionality

- When the "Other Applications" button is click a file browser dialog will load which will allow the user to browse to an .exe file. An example of this browser screen is in Figure 6 below.
- Once an .exe file is selected and the "Open" button is clicked the eye tracker interface will launch the .exe and take control of the mouse until an escape character (Key.Escape in this design) key is pressed.



| Name | Date modified | Type | Size |
|---|---|---|---|
| cicero_v1_Data | 5/31/2013 3:24 AM | File folder | |
| Cicero_v1_src | 5/30/2013 9:46 PM | File folder | |
| Maze_v1_Data | 5/31/2013 3:28 AM | File folder | |
| Maze_v1_src | 5/31/2013 3:33 AM | File folder | |
| cicero_v1.exe | 3/24/2013 3:41 AM | Application | 9,719 KB |
| Maze_v1.exe | 3/24/2013 3:41 AM | Application | 9,719 KB |

***Figure 6-Other Applications Browser***

# Mouse Handler Design & Implementation

## Mouse Handler Design

### Overview

Since the games and other applications are designed to work with traditional plug and play devices like a mouse or keyboard, it was important to consider how the eye tracker will work in this context. The mouse handler was created to utilize the eye tracker in place of a mouse.

### Functionality

- To stop the eye tracker interface from controlling the mouse, or to gain mouse control at any time press the escape hotkey (current escape key used in our design is Key.Escape).
- Current Games- In the "Cicero Game" and the "Gaze Maze", when the user clicks "Start Game from the application, the mouse will be disabled and the eye tracker will take over as the mouse cursor.
  - In the "Cicero Game" the user will gaze in the quadrant for a given length of time (default is set to two seconds) at which time it will act as a click from a mouse and the user can look at the next quadrant.
  - In the "Gaze Maze Game", the user will look where they want the ball to go and the ball will follow.
- Other Applications- For additional applications once the .exe file has been launched via the "Other Applications" button the eye tracking interface will take over mouse control automatically until the escape hotkey is pressed.

# Database Design & Implementation

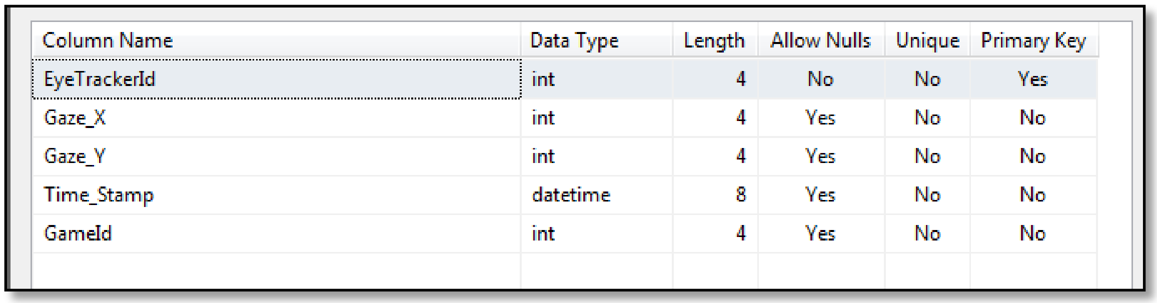
## Database Design

### Overview

In order to save data for later use and across instances we have chosen to use an internal database. This allows us to keep track of all our data but doesn't create an internet or network dependency that is found with traditional databases. Data is saved to a file called AlphaDB.sdf. This file can also be used to import data into a traditional online database at a later time.

### Functionality

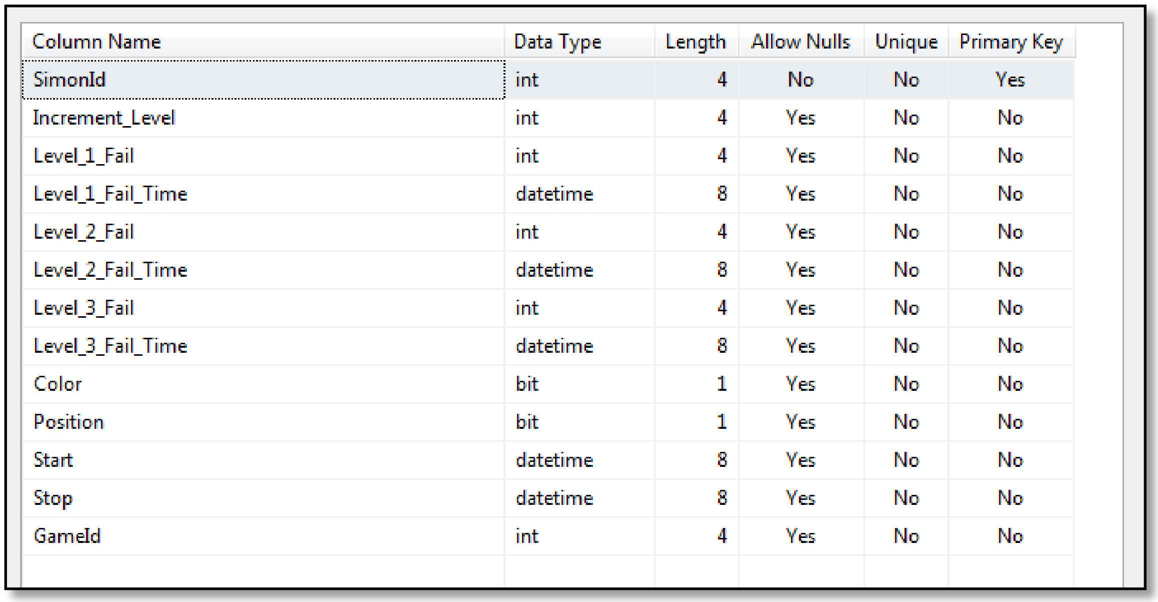
- To enable data saving the user should click "Start Data Recording". This button will set a flag to true that starts data recording after an application is launched and started.
- At the moment we currently have following three data tables structures examples :
  - *Eye Tracker-*
    - This is used in any application used, it records eye tracking data at a selected data rate
    - It uses the following database schema (Figure 7 below):



| Column Name | Data Type | Length | Allow Nulls | Unique | Primary Key |
|---|---|---|---|---|---|
| EyeTrackerId | int | 4 | No | No | Yes |
| Gaze_X | int | 4 | Yes | No | No |
| Gaze_Y | int | 4 | Yes | No | No |
| Time_Stamp | datetime | 8 | Yes | No | No |
| GameId | int | 4 | Yes | No | No |

***Figure 7-Eye Tracker Database Example***

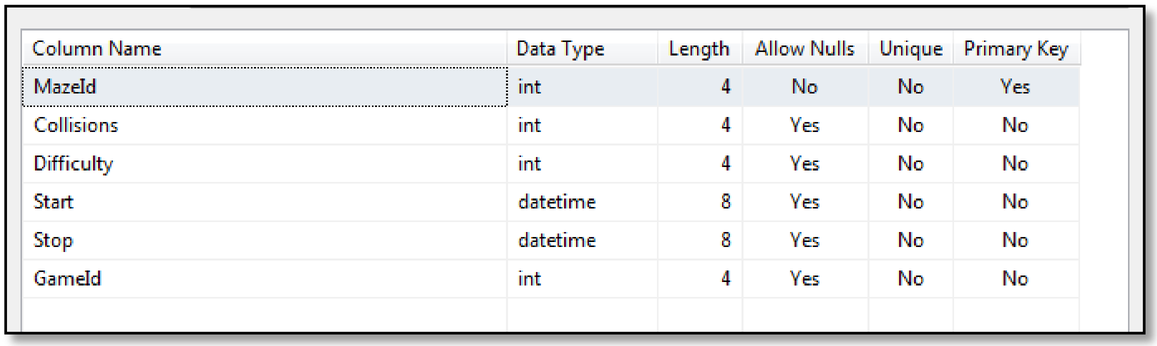
- *"Cicero Game"*-
  - This table is used to store data sent to the eye tracking interface once an instance of the Cicero game has been completed
  - It uses the following database schema (Figure 8 below):



| Column Name | Data Type | Length | Allow Nulls | Unique | Primary Key |
|---|---|---|---|---|---|
| SimonId | int | 4 | No | No | Yes |
| Increment_Level | int | 4 | Yes | No | No |
| Level_1_Fail | int | 4 | Yes | No | No |
| Level_1_Fail_Time | datetime | 8 | Yes | No | No |
| Level_2_Fail | int | 4 | Yes | No | No |
| Level_2_Fail_Time | datetime | 8 | Yes | No | No |
| Level_3_Fail | int | 4 | Yes | No | No |
| Level_3_Fail_Time | datetime | 8 | Yes | No | No |
| Color | bit | 1 | Yes | No | No |
| Position | bit | 1 | Yes | No | No |
| Start | datetime | 8 | Yes | No | No |
| Stop | datetime | 8 | Yes | No | No |
| GameId | int | 4 | Yes | No | No |

***Figure 8-Cicero Game Database Example***

- *"Gaze Maze Game"*-
  - This table is used to store data sent to the eye tracking interface once an instance of the maze game has been completed
  - It uses the following database schema:



| Column Name | Data Type | Length | Allow Nulls | Unique | Primary Key |
|---|---|---|---|---|---|
| MazeId | int | 4 | No | No | Yes |
| Collisions | int | 4 | Yes | No | No |
| Difficulty | int | 4 | Yes | No | No |
| Start | datetime | 8 | Yes | No | No |
| Stop | datetime | 8 | Yes | No | No |
| GameId | int | 4 | Yes | No | No |

***Figure 9-Gaze Maze Game Database Example***

## Future Directions & Summary

This project has brought together an interdisciplinary team of psychology, computer science, computer programming, and game designers to create a strong representative product. Given the nature of the project, the researcher team will continue to tweak the games/applications, add new applications, and perform sufficient usability testing on the eye tracking gui, applications, and database. The team will be continuing past this project deadline to improve upon our designs and implement these additions and revisions for the use in psychological, physiological and gaming research studies at Penn State Erie, the Behrend College

# Research Team

Team Name:  HARE Research Group
Organization:  Penn State Erie, The Behrend College

**Dr. Heather Lum**
Faculty Advisor
4951 College Drive
Erie, PA 16563
(407) 443-8045
hcl11@psu.edu

**Deanna Pettigrew**
Instructional Designer/Documenter
31 Coal Hill Road
Greenville, PA  16125
(814) 455-5895
drp5147@psu.edu

**Shane Halse**
Lead Software Engineer
1413 E 35th Street
Erie, PA  16504
(407) 443-8045
seh297@psu.edu

**Peter Huizar**
Software Developer
3538 Cherry Street First Floor
Erie, PA 16508
(814) 746-8048
pjh5159@psu.edu

**Tyler Ewing**
Database Designer
9 Park Drive
Fairmont, WV  26554
(304) 276-8345
tre5033@psu.edu

**Zekarias Bekele**
Software Developer
5086 Station Road
Erie, PA 16510
(702) 372-2544
zyb5037@psu.edu

**Matthew Kenny**
Lead Game Designer
243 W 29th Street Apt 2
Erie, PA  16508
(814) 969-6777
mtk5138@psu.edu

**Jeffrey Knapp**
Software Developer
4951 College Drive
Erie, PA 16563
(814) 898-6190
jak5616@psu.edu