BSC – HGP- Assignment 02
Pictionary Game
UI Design Document

Introduction

This UI Design Document provides a comprehensive overview of the design choices made for the "Pictionary Game" application. The application is a “Pictionary Game” made using PyQt6, which allows two users to play the game.
This application has several functionalities, such as game modes (easy level or hard level), a scoring and point system, and the ability to export and import paintings from the device to the canvas in a range of different formats.
The design decisions outlined below aim to enhance the user experience and maintain a visually appealing interface.

An explanation of working

The Pictionary game is a Python-based application that utilizes PyQt6 for its graphical user interface (GUI). The application features menus for file operations, brush customization, help/about information, and game mode selection. It also incorporates undo and redo functionality, an eraser tool, a dock widget for displaying game information, and support for mouse events and file operations.
The game logic includes player turns, scores, and secret word reveals. Users can select colors using a color dialog or an additional color wheel. The main program creates an instance of the application and initiates the event loop.

Screenshot of the Application

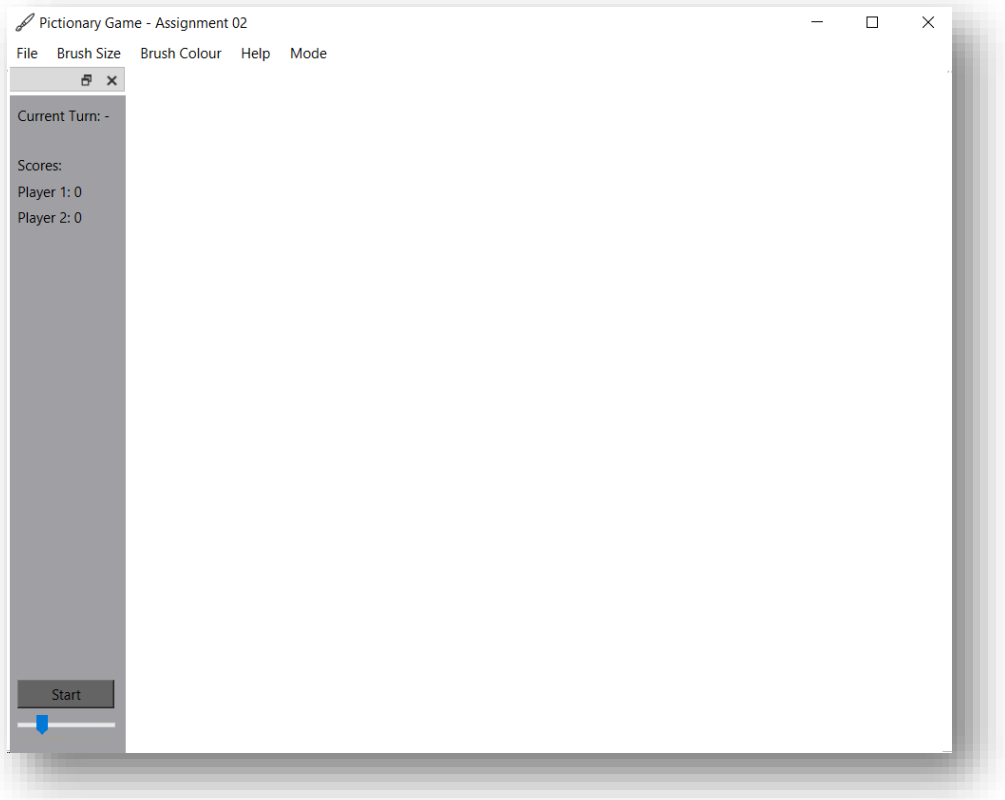


Figure 1- Screenshot of the Application

UI design choices

I have kept my UI design choices simple and straightforward. The UI design for this application is thoughtfully organized and user-friendly.
Firstly, it employs a QMainWindow with clear menus, including standard options like File, Edit, and Mode, ensuring easy access to various functionalities.
Additionally, a QToolBar on the right side enhances user convenience by providing quick access to color selection through a color wheel. The canvas interaction is well handled, with mouse events facilitating drawing.
Users can customize the brush size and color via intuitive menus. The inclusion of standard file operations (Open, Save, Clear, Exit), undo and redo functionalities, and an eraser tool with adjustable size further enhances the user experience.
The design also caters to the specific requirements of a Pictionary game, incorporating turn information, scores, and a game start button within a docked widget. Dynamic word selection from external files adds flexibility to the game.
Overall, the UI design is consistent, well-documented, and includes error handling, contributing to a seamless and enjoyable user experience in the context of a Pictionary game application.

I have created methods such as displaySecretWordPopup() for displaying a popup box to reveal secret words to the current player and selectMode() method that prompts the user to select game mode between 'Hard Mode' and 'Easy Mode'.
The workings of the above are shown in the snippets below.

1) selectMode():

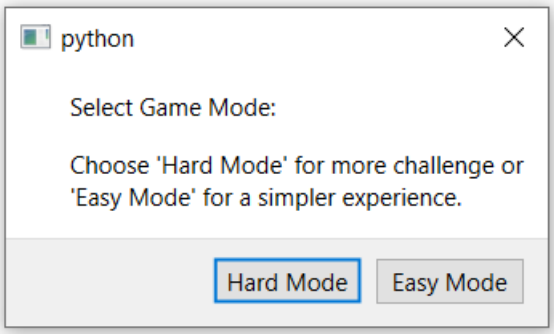


Figure 2- Displays a QMessageBox with option to select between Easy and Hard game modes.

2) displaySecretWordPopup():

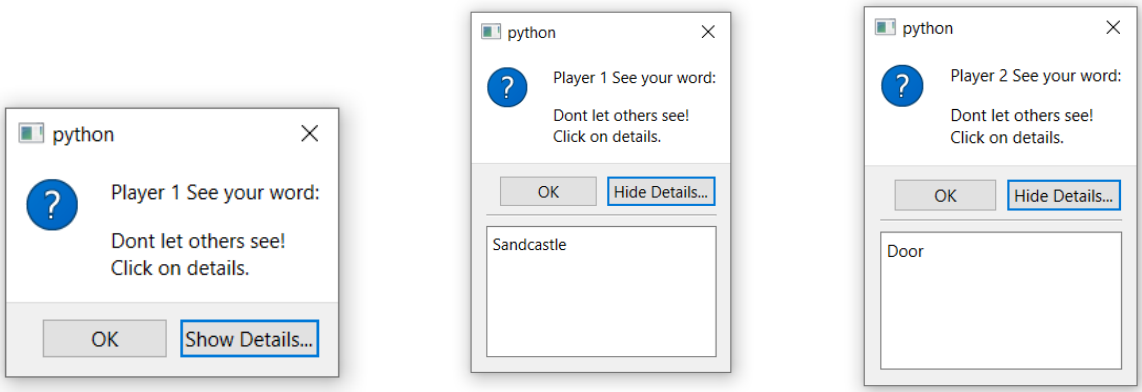


Figure 3- Screenshots displaying the word to draw and the players; player-1 and player-2

Extra Features- 1

1. Color Wheel-

Description:
In addition to the core functionality, the Pictionary Game application incorporates several advanced features to enhance the user experience. One notable feature is the integration of a **QColorWheel**, providing users with an expanded and intuitive color selection mechanism.

Integrating a color wheel expanded the color options beyond the basic palette, offering users a more extensive range of creative expression. It is seamlessly incorporated into the toolbar under the Brush color menu for quick access.

Overall, it enhances user creativity and flexibility in color selection.

Implementation:
The colorDialogMenu method triggers the QColorDialog, allowing users to choose custom colors. The selected color is set as the brush color if it is valid.

Benefits:
Dynamic Selection: Users can choose from a spectrum of colors for a personalized drawing experience.
User-Friendly: Integrated into the toolbar for easy access, enhancing the overall workflow.
Precision and Creativity: Supports comprehensive drawing tools for precise color choices.

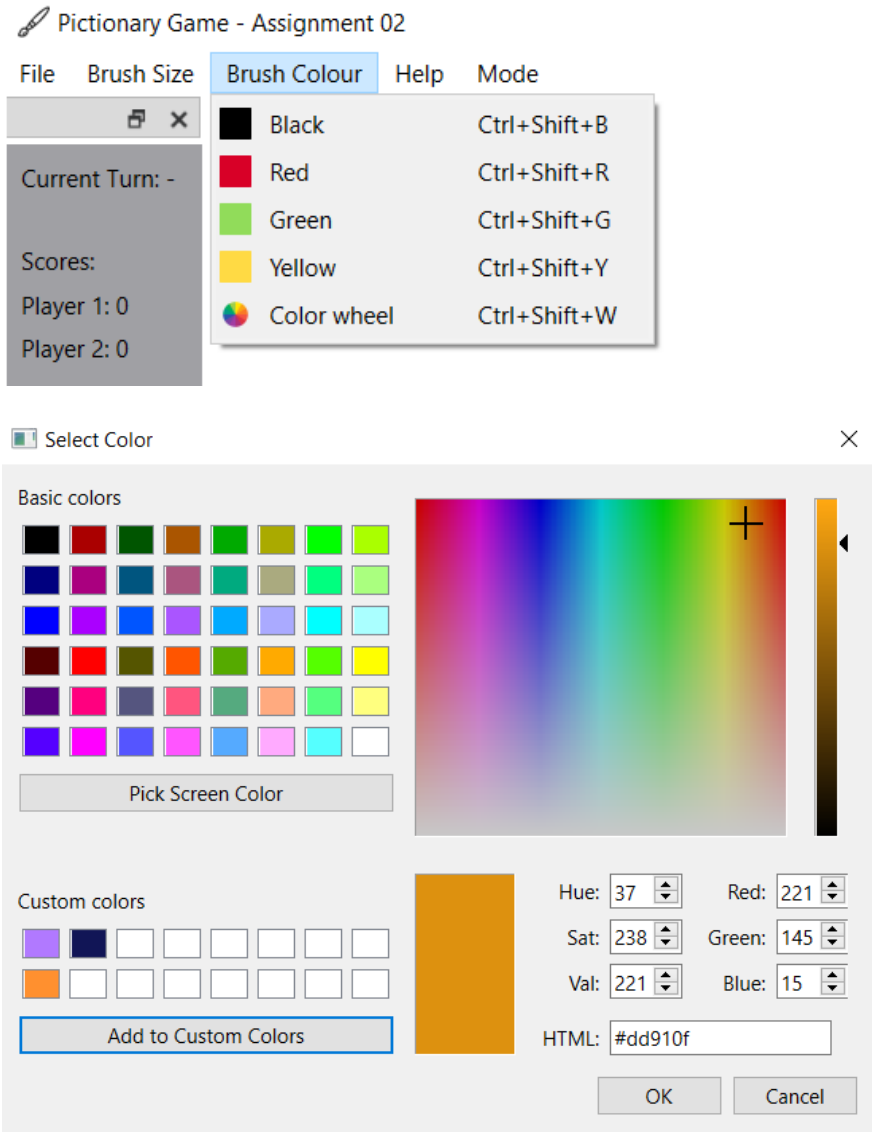


Figure 4- Color Wheel

Extra Features- 2

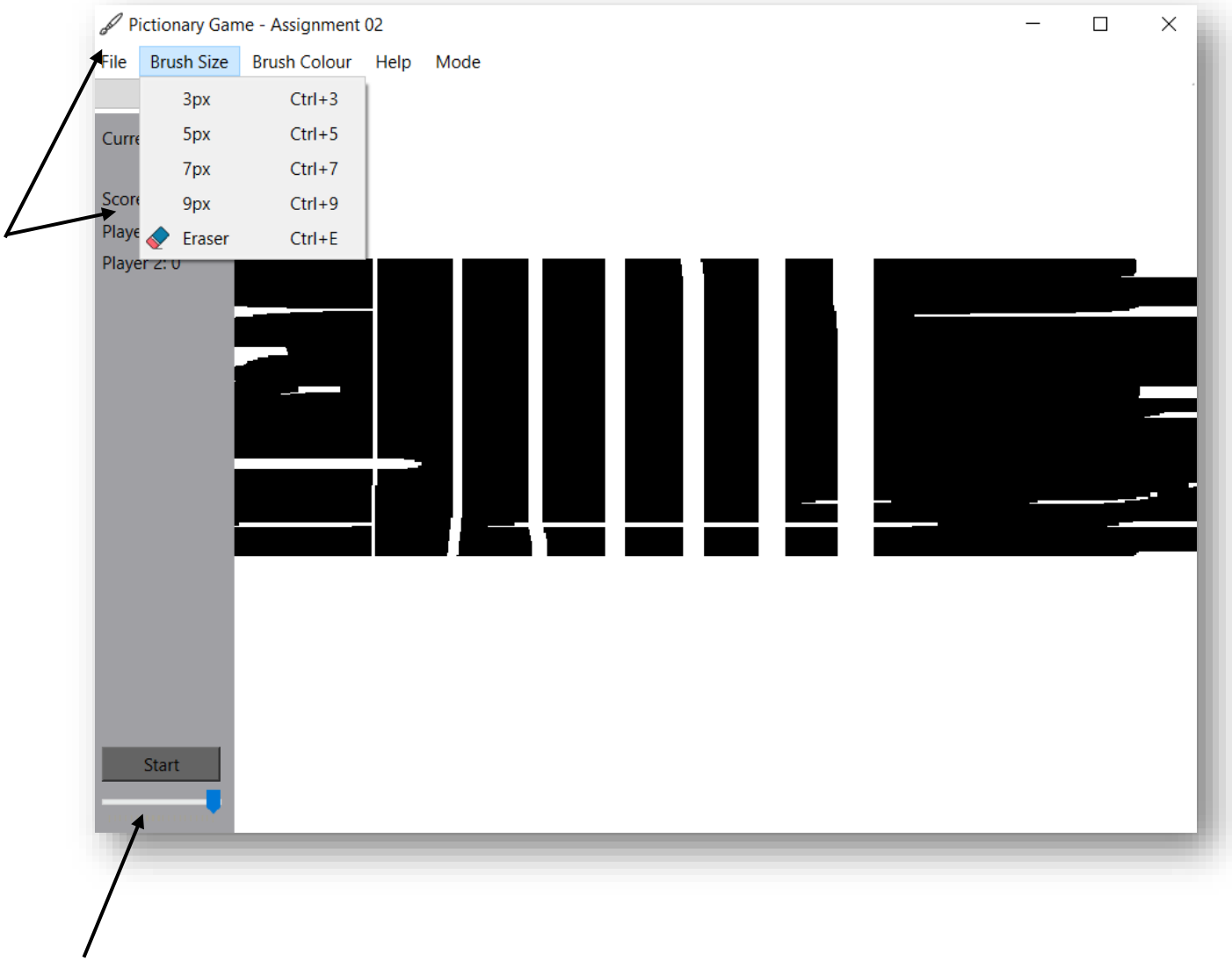
2. Eraser Tool-

Description:
The Pictionary Game introduces an eraser tool for precise content removal. It is seamlessly incorporated into the toolbar in the ‘Brush Size’ menu. It provides users with the ability to erase content with varying brush sizes, including an adjustable eraser size slider for enhanced control.

Implementation:
The eraser tool is activated through the eraser method, dynamically adjusting the brush color and size for effective content erasure. The presence of an eraser size slider allows users to customize the tool's size according to their editing needs, which is just beneath the ‘Start’ button.

Benefits:
Precision Erasure: Users can remove content with varying eraser sizes, allowing for detailed editing.
Efficient Workflow: Integrated into the toolbar, ensuring quick and easy access during drawing sessions.
Enhanced Editing: Supports comprehensive editing tools, including an eraser size slider for fine-tuning adjustments to drawings.

The following snippet shows how the eraser can be used, starting from the minimal size of the eraser to the maximum size, which is shown on the canvas below:



Eraser Slider Widget

Extra Features- 3

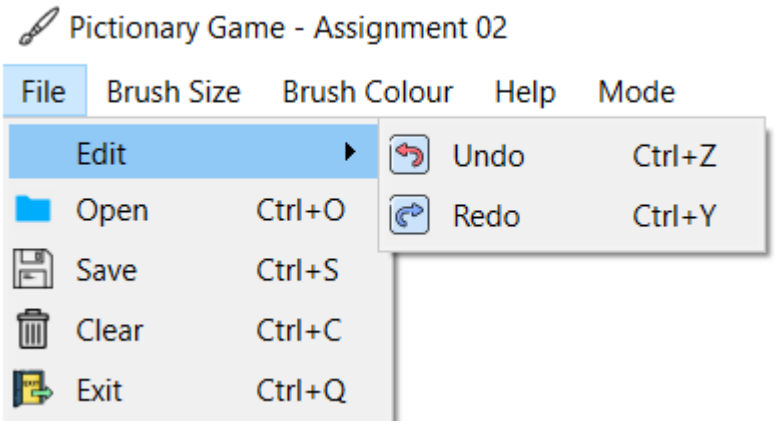
3. Undo and Redo-

Description:
The third additional feature that Pictionary Game incorporates is robust ‘Undo’ and ‘Redo’ functionality to empower users in their editing journey. This feature allows users to undo and redo their drawing actions, providing flexibility and control over the creative process. The shortcuts are also provided.

Implementation:
The ‘Undo’ and ‘Redo’ actions are seamlessly integrated into the 'Edit' menu. Triggered through the undo and redo methods, they navigate through the drawing history, enabling users to revert or reinstate changes. The history is managed with an array, ensuring efficient storage and retrieval.

Benefits:
Flexible Editing: Easily navigate through drawing states for corrections and experimentation.
Improved Workflow: Integrated into the menu for a straightforward approach to undo and redo actions.
Error Prevention: Acts as a safety net, allowing users to revert unintended changes and maintain drawing integrity.

The following snippet shows the design of this functionality:



Conclusion

In conclusion, the "Pictionary Game" UI Design Document reflects a user-centric approach with a simple and intuitive interface using PyQt6. The design prioritizes user-friendliness, featuring clear menus, a convenient QToolBar, and well-implemented canvas interactions. For the future, I aim to enhance the game by adding more icons, including animations, making it mobile-friendly, and incorporating audio and sound effects.

Student Name: Udayy Singh Pawar
Student Number: 3085192