**Introduction**

This report presents an analysis on my Ruby-based music player application that utilizes the Gosu library. The primary objective of the application is to facilitate the management and playback of music albums, providing a user interface for album and track navigation and control.

**Overview**

The code comprises several crucial components:

* The **Genre** module defines music genres.
* The **ZOrder** module organizes the drawing order.
* The **Album** class manages album details, including tracks.
* The **Track** class encapsulates individual track information.
* The **MusicPlayerMain** class serves as the primary application window and controller.

**Code Analysis**

1. **Initialization and Window Setup**:
   * The **MusicPlayerMain** class initializes the application window and sets up essential variables, including data structures for albums and tracks.
2. **Album and Track Loading**:
   * The **load\_albums** method parses album data from a designated text file (**albums.txt**), creating **Album** objects with associated **Track** instances.
3. **User Interface**:
   * Drawing methods (**draw\_albums**, **draw\_tracks**, etc.) handle the graphical user interface, displaying albums, tracks, playback details, progress bar, and control buttons.
   * Event-handling methods (**handle\_scroll\_left**, **handle\_mouse\_click**, etc.) govern user interaction, such as button clicks and navigation.
4. **Playback and Controls**:
   * The **play\_track** method manages track playback, while **update** controls track progress and enables repeat mode.
5. **Usability and Error Handling**:
   * The application offers user-friendly features, including mouse interaction, track selection, and navigation buttons. However, it lacks robust error handling for invalid user input or file loading issues.

**Observations**

* **Strengths**:
  + Well-structured code architecture, organized into distinct classes and methods.
  + Effective use of Gosu library for graphical rendering and user interface elements.
  + Functional implementation of album loading, track selection, and playback controls.
* **Areas for Improvement**:
  + Enhancements required in error handling and input validation to ensure a smoother user experience.
  + Code documentation and comments could enhance readability and comprehension.

Structure Chart

