System Architecture – PUZ-BUG

Introduction

Puz-Bug (Puzzle Debugger) is a game that will enable students to learn debugging. In this game, the user will select a puzzle that he wishes to solve. The puzzle will be divided into multiple pieces, each of which will have a code snippet linked to it. Initially, the puzzle pieces will be locked and the user will have to debug the snippet to unlock the puzzle piece. Once all the pieces are unlocked, the user must solve the puzzle to win.

Rationale of the architecture

In our project, each component is a complete unit but we need to integrate them so that one component provides the services to the other one. Each layer(component) performs a function that lets the higher layer (other component) abstract away the complexities. Therefore, we chose Layered architecture for our project.

Our architecture is divided into following layers:

1. *GUI (Graphical User Interface)*

   This is the layer that the users will interact with and it is equivalent to the presentation layer. The main element of the GUI will be the puzzle. The user will have the ability to select any puzzle picture from the given pool of pictures. Once the picture is chosen, the puzzle will be divided into pieces. Each piece will have a code snippet attached to it. On selection, the user will be able to debug the code using this interface.

2. *Debugging Logic*

   This layer is the critical component of our system. This layer coordinates with the UI, Debugger and the code snippets.

3. *Code Snippet Files*

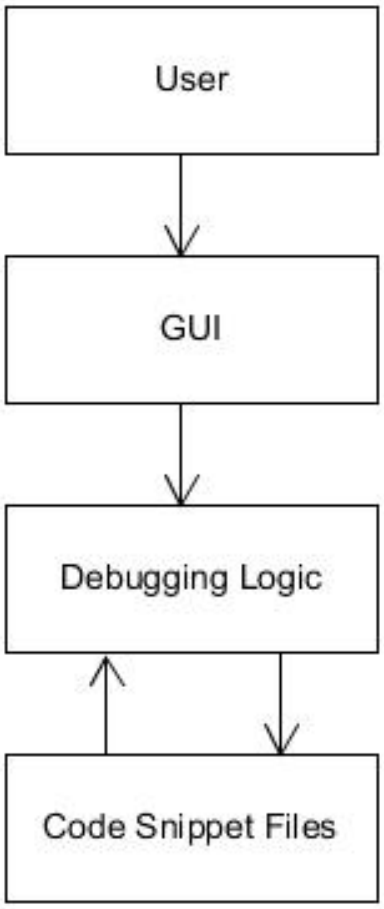
   This layer will be responsible for handling the code snippets and is equivalent to Data layer. The code snippet that the user must debug will be stored and retrieved from this layer. The solutions will also be stored here so that if the user is unable to debug it, he can view the correct solution. Although, there can be many ways to solve a code snippet, we will provide just one correct solution.

Architectural Drivers

The application that we are building requires a clear separation between the user interface, the debugging functionality and how we plan to store our files. This led us to decide our main architectural drivers. They are:

- Clean User Interface
- Debugging Platform
- Data Storage for code snippets

Conceptual Architecture Diagram:



Central concerns for your system

The central concern for our system is integration of the debugging platform. Since we are linking code snippets to each puzzle piece, we need to add a debugging platform into our application to enable the users to edit and run their code and view the number of errors (if any). Once the code is correctly debugged, our application should identify this and enable the locked puzzle piece.