Process Description

Title : PUZ-BUG (Puzzle Debugging Game)

Introduction

Debugging is an essential part of software testing. It refers to finding the errors in the code and fixing them. As a software developer, this is an important skill that one must possess. To teach this skill, we are planning to develop a puzzle debugging game called 'Puz-Bug'. This game will help the students to learn debugging the code in a new way which will be fun.

Team Organization

Our team consists of 5 members - Juwaria, Panika, Rahul, Yu Hao and Yuan. Hence, for the project, we will allocate tasks as per everyone's capabilities and areas of interest. Since there are 4 Demo's, we will try to follow Agile software development approach and have fixed sprints of 2 weeks each. We also plan to set up weekly meetings to track the progress and discuss and resolve the issues we face. There are 3 main tasks that need to be worked upon - User Interface, Debugger & its Integration and Code Snippets. The task allocation is discussed in detail in a further section.

Outline of Tasks

1. Develop the User Interface
2. Find a debugging platform/plugin to use
3. Develop code snippets for the user using Java
4. Integrate the debugging tool with the UI
5. Ability to check if the user debugged the code correctly
6. Check if the puzzle is correctly solved/arranged
7. Use Jenkins in some way (Maybe for testing using Junit and Jacoco)

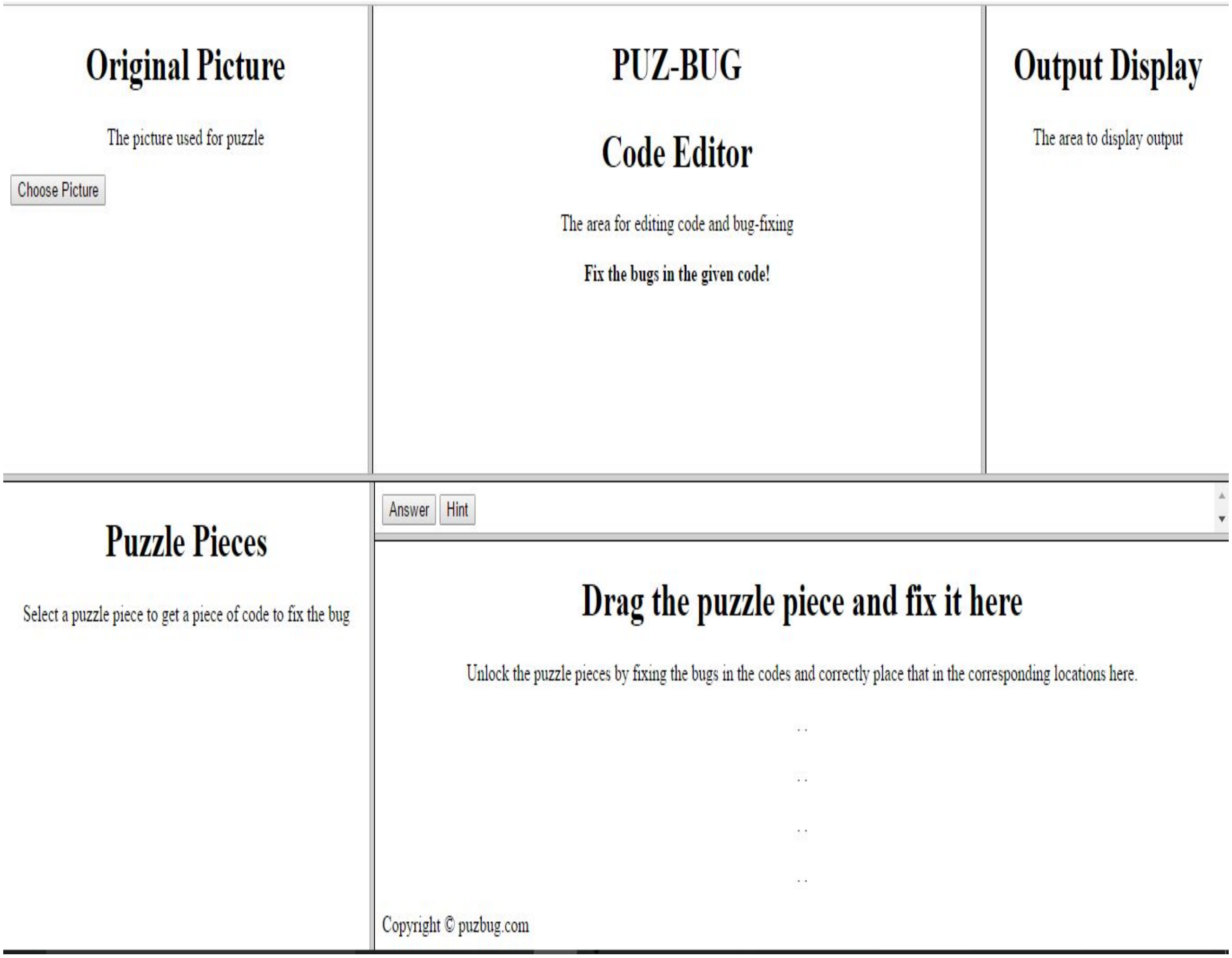
Features

1. Picture Selection: Chose a picture from the games library
2. Level Selection: There will be 3 levels for this game- Easy, Medium and Hard. Each level will correspond to the difficulty level which is directly related to the number of pieces the image is split into – 2, 4 and 6 for easy, medium and hard levels respectively. With each difficulty level, the number of bugs that need to be fixed will be increased.
3. Drag and drop puzzle pieces: The user will have the ability to drag and drop the pieces into the frame.
4. Code view and edit: The user will be given a code once a piece is selected. The code will appear in a section of the user interface and the user will be allowed to edit the code and run it and also to see the results in another section. This will help the user in debugging.
5. Hints: For each puzzle piece the user will be given two hints. The hints option will be available in the form of a button and the user can click it to see the hints. This button will be available above the grid where the puzzle is to be assembled.
6. Give up button: We will have another button so that the user can give up if he/she is unable to debug and the correct code will be shown to the user with the corrected part highlighted.
7. Score: A score will also be maintained based on the number of code snippets that are correctly

debugged.

8. Timer: The game will also have a timer which will keep track of the time that is taken to solve each code snippet. The user will be awarded extra points if the code is debugged in a specific amount of time.

Concept Sketch



Original Picture

The picture used for puzzle

PUZ-BUG

Code Editor

The area for editing code and bug-fixing

Fix the bugs in the given code!

Output Display

The area to display output

Puzzle Pieces

Select a puzzle piece to get a piece of code to fix the bug

Drag the puzzle piece and fix it here

Unlock the puzzle pieces by fixing the bugs in the codes and correctly place that in the corresponding locations here.

. .

. .

. .

. .

Copyright © puzbug.com

Task Allocation

Task No.	Task Description	Team Member
1	Develop User Interface	Panika Valecha
2	Search a debugging platform/plugin to use	Rahul Rajendran
3	Develop code snippets for the user using Java	Juwaria Adil, Yuan Xu
4	Integrate the debugging tool with the user intereface	Juwaria Adil, Panika Valecha, Rahul Rajendran, Yuhao Ian, Yuan Xu
5	Ability to check if the user debugged the code correctly	Juwaria Adil, Panika Valecha, Rahul Rajendran, Yuhao Ian, Yuan Xu
6	Check if the puzzle is correctly solved/arranged	Juwaria Adil, Panika Valecha, Rahul Rajendran, Yuhao Ian, Yuan Xu
7	Use Jenkins in some way (Maybe for testing using Junit and Jacoco)	Juwaria Adil, Panika Valecha, Rahul Rajendran, Yuhao Ian, Yuan Xu

Schedule

Demo 1

We will develop the user interface. The basic frames will be added. In addition to this, we will implement the ability to load the pictures for puzzle selection.

Demo 2

We will give the user option to select the level which will split the image into 2, 4 and 6 pieces according to the type of level selected. Next, we will get all code snippets for each pieces. We will try to implement the hint and give up button for each pieces.

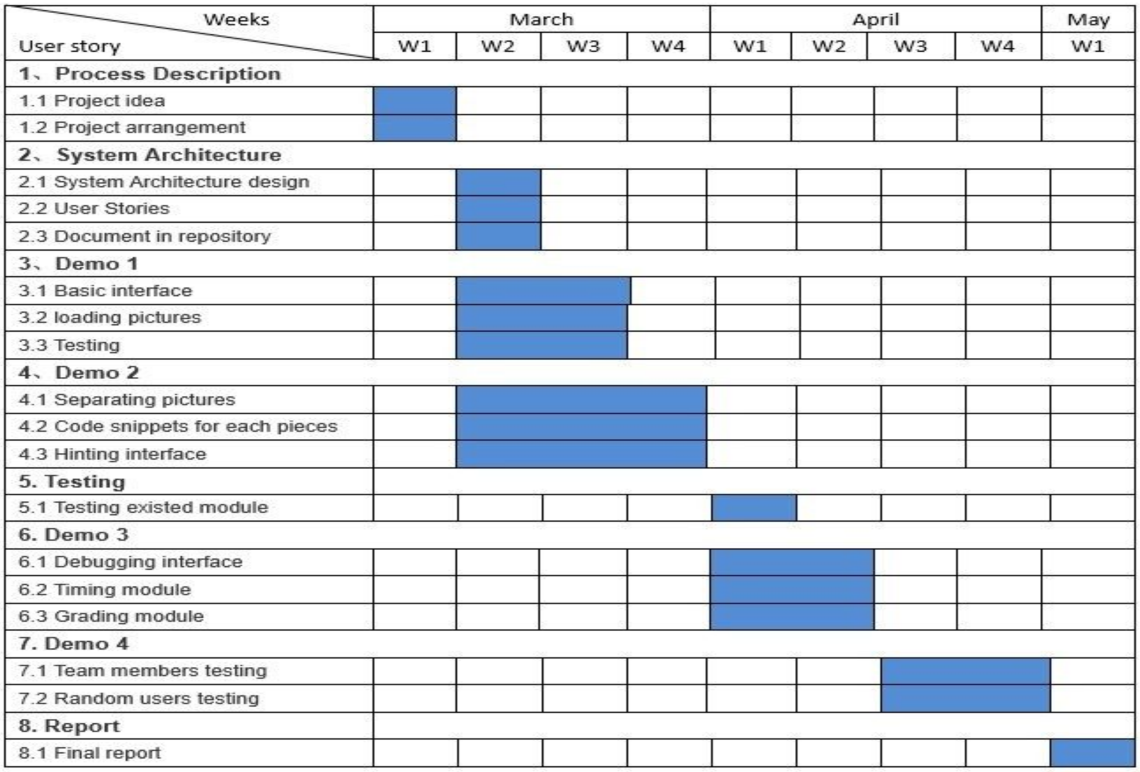
Demo 3

We will implement the functionality of how to debug the code given for each piece. We are going to add the time and score as the standard measure for the progress of the user. And we will also perform some refactoring.

Demo 4

We will perform testing and demo the final version with improvements that are needed and by making the code cleaner. We will play this game during the team members firstly, then ask others to play this game. Then we will improve this game based on some feedback.

The team timeline for the entire semester is as follows:



Weeks	March				April				May
User story	W1	W2	W3	W4	W1	W2	W3	W4	W1
1、Process Description									
1.1 Project idea	■								
1.2 Project arrangement	■								
2、System Architecture									
2.1 System Architecture design		■							
2.2 User Stories		■							
2.3 Document in repository		■							
3、Demo 1									
3.1 Basic interface		■	■						
3.2 loading pictures		■	■						
3.3 Testing		■	■						
4、Demo 2									
4.1 Separating pictures		■	■	■					
4.2 Code snippets for each pieces		■	■	■					
4.3 Hinting interface		■	■	■					
5. Testing									
5.1 Testing existed module					■				
6. Demo 3									
6.1 Debugging interface					■	■			
6.2 Timing module					■	■			
6.3 Grading module					■	■			
7. Demo 4									
7.1 Team members testing							■	■	
7.2 Random users testing							■	■	
8. Report									
8.1 Final report									■