

AGILE LABORATORY
DRAGON FLY (G1 TEAM 3)
(CSX-326)



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DR. B R AMBEDKAR NATIONAL INSTITUTE OF TECHNOLOGY

JALANDHAR – 144011, PUNJAB (INDIA)

JANUARY-MAY, 2018

SUBMITTED TO
PARAMVIR SINGH
ASSISTANT PROFESSOR

SUBMITTED BY:
TEAM 3 G1
PARTH(15103007)
KARAN(15103014)
HIMAKSHI(15103015)
NANCY(15103016)
ANJALI(15103022)
RAMANDEEP(15103023)
ARSHDEEP(15103025)
GAURAV(15103029)

<u>S.NO</u>	<u>INDEX</u>	<u>REMARKS</u>
1.	About Agile Framework and Basic terminologies of the agile.	
2.	ANT practical program and ANT theory.	
3.	Jenkins practical steps and theory.	
4.	Use-case Diagram of final project.	
5.	Class Diagram of final project.	
6.	Team Division	
7.	Sprint N:- 1. Tasks decided to complete in the sprint 2. task division and allotted time to task in group. 3. Task completed and remaining tasks (not included in the final sprint.)	
8.	Screen shots of the Quick scrum tool and Spread sheet downloaded from the Quick Scrum.	
9	Testing 1.UI and Features Teasting 2.Installation Testing 3. Performance Testing	

OBJECTIVE-1

Aim: Agile Framework and Basic Terminologies of the agile.

Agile Framework:

One of the major handicaps of the traditional water-fall model was that – until the first phase is complete, the application does not move to the other phase. And if by chance there are some changes in the later stage of the cycle, it becomes very challenging to implement those changes, as it would involve revisiting the earlier phases and redoing the changes.

Agile can be used to create high quality products. Agile does this by creating products in small increments, with each individual increment tested before it is considered done. This process builds quality into the product versus inspecting for quality later.

Agile is an umbrella term for several iterative and incremental software development approaches, with each of those variations being its own Agile framework. The most popular Agile frameworks include Scrum, Crystal, Dynamic Systems Development Method, and Feature-Driven Development. Mendix, in particular, subscribes to the Scrum methodology. Any Agile development project involves continuous planning, continuous testing, continuous integration, and other forms of continuous development of both the project and the application resulting from the Agile framework.

Each Agile framework is considered lightweight. Rules and practices are kept to a minimum, especially when compared to traditional waterfall-style development processes, and are designed to be adaptable to all kinds of circumstances. The focus, instead, falls on empowering developers of all kinds to collaborate and make decisions together as a group quickly and effectively. The grand vision behind the Agile development methodology is to create applications in small increments, with each individual increment tested before it is considered complete. This process assures quality is “built” into the product, versus inspecting for quality later.

Manifesto for Agile Software Development :



- *Individuals and Interactions over processes and tools*

- *Working Software over comprehensive documentation*
- *Customer Collaboration over contract negotiation*
- *Responding to Change over following a plan*

Agile software development principles :

The *Manifesto for Agile Software Development* is based on twelve principles :

1. Customer satisfaction by early and continuous delivery of valuable software
2. Welcome changing requirements, even in late development
3. Working software is delivered frequently (weeks rather than months)
4. Close, daily cooperation between business people and developers
5. Projects are built around motivated individuals, who should be trusted
6. Face-to-face conversation is the best form of communication (co-location)
7. Working software is the primary measure of progress
8. Sustainable development, able to maintain a constant pace
9. Continuous attention to technical excellence and good design
10. Simplicity—the art of maximizing the amount of work not done—is essential
11. Best architectures, requirements, and designs emerge from self-organizing teams
12. Regularly, the team reflects on how to become more effective, and adjusts accordingly

Basic Terminologies of Agile :

Acceptance Testing :

Formal testing conducted to determine whether or not a system satisfies its acceptance criteria and to enable the customer to determine whether or not to accept the system.

Backlog Grooming :

Backlog grooming is the process of adding new user stories to the backlog, re-prioritizing existing stories as needed, creating estimates, and deconstructing larger stories into smaller stories or tasks.

Burndown Chart :

A graph that displays the total task hours remaining per day. It shows where the team stands regarding completing the tasks that have been committed to the sprint. The X-axis represents days in the sprint, while the Y-axis is effort remaining.

ContinuousIntegration :

Continuous Integration (CI) is an eXtreme Programming (XP) practice where members of a delivery team frequently integrate their work (e.g. hourly, or at least once daily). Each integration is verified by an automated build, which also performs testing, to detect any integration errors quickly and automatically. The main goal of CI is to avoid what is commonly called integration or merge hell.

Cross-Functional Team :

Team comprised of members with all functional skills and specialties (often called multi-skilled) necessary to complete a project from start to finish.

Extreme Programming :

An agile software development methodology which is intended to improve software quality and responsiveness to changing customer requirements. XP advocates frequent “releases” in short development cycles, which is intended to improve productivity and introduce checkpoints at which new customer requirements can be adopted. Other elements of extreme programming include: pair programming, extensive code reviews, unit testing, Continuous Integration to name a few.

Iteration :

A period (from 1 week to 2 months in duration) during which the Agile development team produces an increment of completed software.

Lean Software Development :

Lean software development or just Lean is focused on reducing waste and optimizing the software production value stream.

Pair Programming :

An agile software development technique in which two programmers work together at one workstation. One types in code while the other reviews each line of code as it is typed in. The person typing is called the driver. and the person reviewing the code is called the observer or navigator. The two programmers switch roles frequently.

Planning Poker :

Planning Poker is a consensus-based technique for estimating, mostly used to estimate effort or relative size of tasks in software development. The team use the fibonacci series or T-shirt sizing to estimate stories during the planning poker game.

Product Backlog :

Product backlog is like a wish list of features that business want to deliver in the long term. It is a collection of stories and tasks the team will work on at some point in the future. The Product Owner maintains this list of product backlog in accordance to business priorities and needs.

Refactoring :

Changing existing software code in order to improve the overall design. Refactoring normally doesn't change the observable behavior of the software; it improves its internal structure.

Retrospective:

A timeboxed meeting held at the end of the sprint in which the team examines its processes to determine what succeeded and what could be improved. The retrospective is key to continuous improvement.

Scrum :

SCRUM is a process in agile methodology which is a combination of Iterative model and incremental model. Scrum is comprised of a series of short iterations – called sprints – each of which ends with the delivery of an increment of working software.

Some of the key characteristics of SCRUM include:

- Self-organized and focused team
- No huge requirement documents, rather have very precise and to the point stories.
- Cross functional team works together as a single unit.
- Close communication with the user representative to understand the features.
- Has definite time line of max 1 month.
- Instead of doing the entire “thing” at a time, Scrum does a little of everything at a given interval
- Resources capability and availability are considered before committing any thing.

Scrum Team :

The scrum team is a cross-functional and self-organizing group that is responsible for delivering the software. The scrum team includes multi-skilled people who understand customer requirements and conduct software design, coding and testing. Additional skills (e.g. UI design,

usability, etc.) may also be included. The scrum team is responsible for all work commitments and outcomes.

ScrumMaster :

The ScrumMaster is responsible for maintaining the Scrum process and the overall health of the team. The ScrumMaster assures that the team is fully functional and productive by removing any obstacles that may be impeding the team's progress. The ScrumMaster also organizes the Scrum ceremonies.

Spike :

A story or task aimed at answering a question or gathering information, rather than implementing product features, user stories, or requirements.

Sprint : In product development, a sprint is a set period of time during which specific work has to be completed and made ready for review. A typical sprint length is usually 2 weeks and normally not any longer than 4 weeks.

Sprint Backlog :

Based on the priority, user stories are taken from the Product Backlog one at a time. The Scrum team brainstorms on it, determines the feasibility and decides on the stories to work on a particular sprint. The collective list of all the user stories which the scrum team works on a particular sprint is called the Sprint backlog.

Sprint Planning :

Sprint planning sessions are held just before the start of a new sprint. In this session the team identifies the tasks that need to be done and decides how many story points they can commit to for the upcoming sprint.

User Story :

A User Story (a.k.a. Story) can be thought of as a requirement, feature which has some business value. Stories describe the work that must be completed to deliver a feature for a product. Stories are the basic unit of communication, planning, and negotiation between the Scrum Team, Business Owners and the Product Owner.

Story Points :

Story points are a quantitative indication of the complexity of a user story. Based on the story point, estimation and efforts for a story are determined. A story point is relative and is not absolute.

Velocity :

Velocity measures how much work a team can complete in an iteration. Velocity is often measured in stories or story points. Velocity may also measure tasks in hours or an equivalent unit.

OBJECTIVE-2

Aim: ANT build tool and theory.

Apache Ant is a Java based build tool from Apache Software Foundation. Apache Ant's build files are written in XML and they take advantage of being open standard, portable and easy to understand. This tutorial should show you how to use Apache ANT to automate the build and deployment process in simple and easy steps. After completing this tutorial, you should find yourself at a moderate level of expertise in using Apache Ant from where you may take yourself to next levels.

On an average, a developer spends a substantial amount of time doing mundane tasks like build and deployment that include:

- Compiling the code
- Packaging the binaries
- Deploying the binaries to the test server
- Testing the changes
- Copying the code from one location to another

To automate and simplify the above tasks, Apache Ant is useful. It is an Operating System build and deployment tool that can be executed from the command line.

- Ant is the most complete Java build and deployment tool available.
- Ant is platform neutral and can handle platform specific properties such as file separators.
- Ant can be used to perform platform specific tasks such as modifying the modified time of a file using 'touch' command.
- Ant scripts are written using plain XML. If you are already familiar with XML, you can learn Ant pretty quickly.
- Ant is good at automating complicated repetitive tasks.
- Ant comes with a big list of predefined tasks.
- Ant provides an interface to develop custom tasks.
- Ant can be easily invoked from the command line and it can integrate with free and commercial IDEs.

Installing Apache Ant

It is assumed that you have already downloaded and installed Java Development Kit (JDK) on your computer. If not, please follow the instructions here.

- Ensure that the JAVA_HOME environment variable is set to the folder where your JDK is installed.
- Download the binaries from <https://ant.apache.org>
- Unzip the zip file to a convenient location c:\folder. using Winzip, winRAR, 7-zip or similar tools.
- Create a new environment variable called **ANT_HOME** that points to the Ant installation folder, in this case **c:\apache-ant-1.8.2-bin** folder.
- Append the path to the Apache Ant batch file to the PATH environment variable. In our case this would be the **c:\apache-ant-1.8.2-bin\bin** folder.

Verifying Apache Ant Installation

To verify the successful installation of Apache Ant on your computer, type ant on your command prompt.

You should see an output similar to –

```
C:\>ant -version
```

```
Apache Ant(TM) version 1.8.2 compiled on December 20 2010
```

If you do not see the above output, then please verify that you have followed the installation steps properly.

Installing Eclipse

This tutorial also covers integration of Ant with Eclipse IDE. Hence, if you have not installed Eclipse already, please download and install Eclipse

To install Eclipse –

- Download the latest Eclipse binaries from www.eclipse.org
- Unzip the Eclipse binaries to a convenient location, say c:\folder
- Run Eclipse from c:\eclipse\eclipse.exe

For this exercise, create a file called build.xml anywhere in your computer with the following contents in it –

```
<?xml version = "1.0"?>
<project name = "Hello World Project" default = "info">
  <target name = "info">
    <echo>Hello World - Welcome to Apache Ant!</echo>
  </target>
</project>
```

Note that there should be no blank line(s) or whitespace(s) before the xml declaration. If you allow them, the following error message occurs while executing the ant build -

The processing instruction target matching "[xX][mM][lL]" is not allowed.

All build files require the project element and at least one target element.

Program

```
<target name = "deploy" depends = "package">
  ....
</target>

<target name = "package" depends = "clean,compile">
  ....
</target>

<target name = "clean" >
  ....
</target>

<target name = "compile" >
```

```
....  
</target>
```

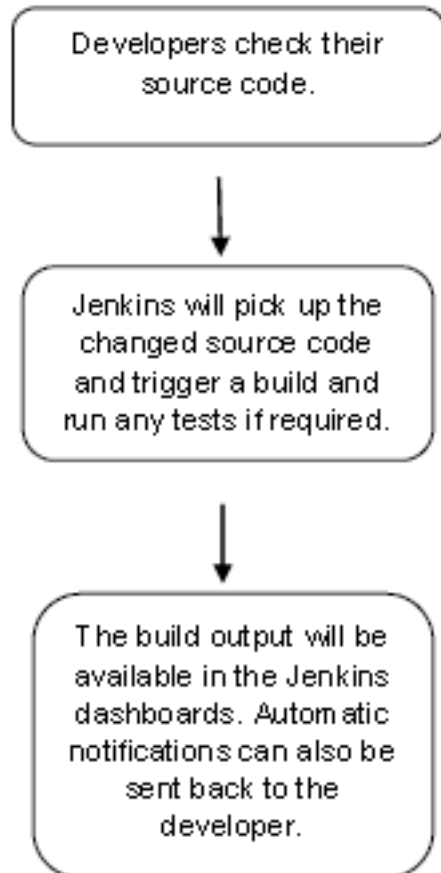
```
<?xml version = "1.0"?>  
<project name = "Hello World Project" default = "info">  
  <property name = "sitename" value = "www.tutorialspoint.com"/>  
  
  <target name = "info">  
    <echo>Apache Ant version is ${ant.version} - You are at ${sitename} </echo>  
  </target>  
</project>
```

```
C:\>ant  
Buildfile: C:\build.xml  
  
info: [echo] Apache Ant version is Apache Ant(TM) version 1.8.2  
       compiled on December 20 2010 - You are at www.tutorialspoint.com  
  
BUILD SUCCESSFUL  
Total time: 0 seconds  
C:\>
```

OBJECTIVE-3

Aim: Jenkins practical steps and theory.

Jenkins is a software that allows **continuous integration**. Jenkins will be installed on a server where the central build will take place. The following flowchart demonstrates a very simple workflow of how Jenkins works.



Along with Jenkins, sometimes, one might also see the association of **Hudson**. Hudson is a very popular open-source Java-based continuous integration tool developed by Sun Microsystems which was later acquired by Oracle. After the acquisition of Sun by Oracle, a fork was created from the Hudson source code, which brought about the introduction of Jenkins.

What is Continuous Integration?

Continuous Integration is a development practice that requires developers to integrate code into a shared repository at regular intervals. This concept was meant to remove the problem of finding later occurrence of issues in the build lifecycle. Continuous integration requires the developers to have frequent builds. The common practice is that whenever a code commit occurs, a build should be triggered.

Download Jenkins

The official website for Jenkins is Jenkins. If you click the given link, you can get the home page of the Jenkins official website as shown below.

By default, the latest release and the Long-Term support release will be available for download. The past releases are also available for download. Click the Long-Term Support Release tab in the download section.



Starting Jenkins

Open the command prompt. From the command prompt, browse to the directory where the jenkins.war file is present. Run the following command

```
D:\>Java -jar Jenkins.war
```

After the command is run, various tasks will run, one of which is the extraction of the war file which is done by an embedded webserver called winstone.

D:\>Java -jar Jenkins.war

Running from: D:\jenkins.war

Webroot: \$user.home/.jenkins

Sep 29, 2015 4:10:46 PM winstone.Logger logInternal

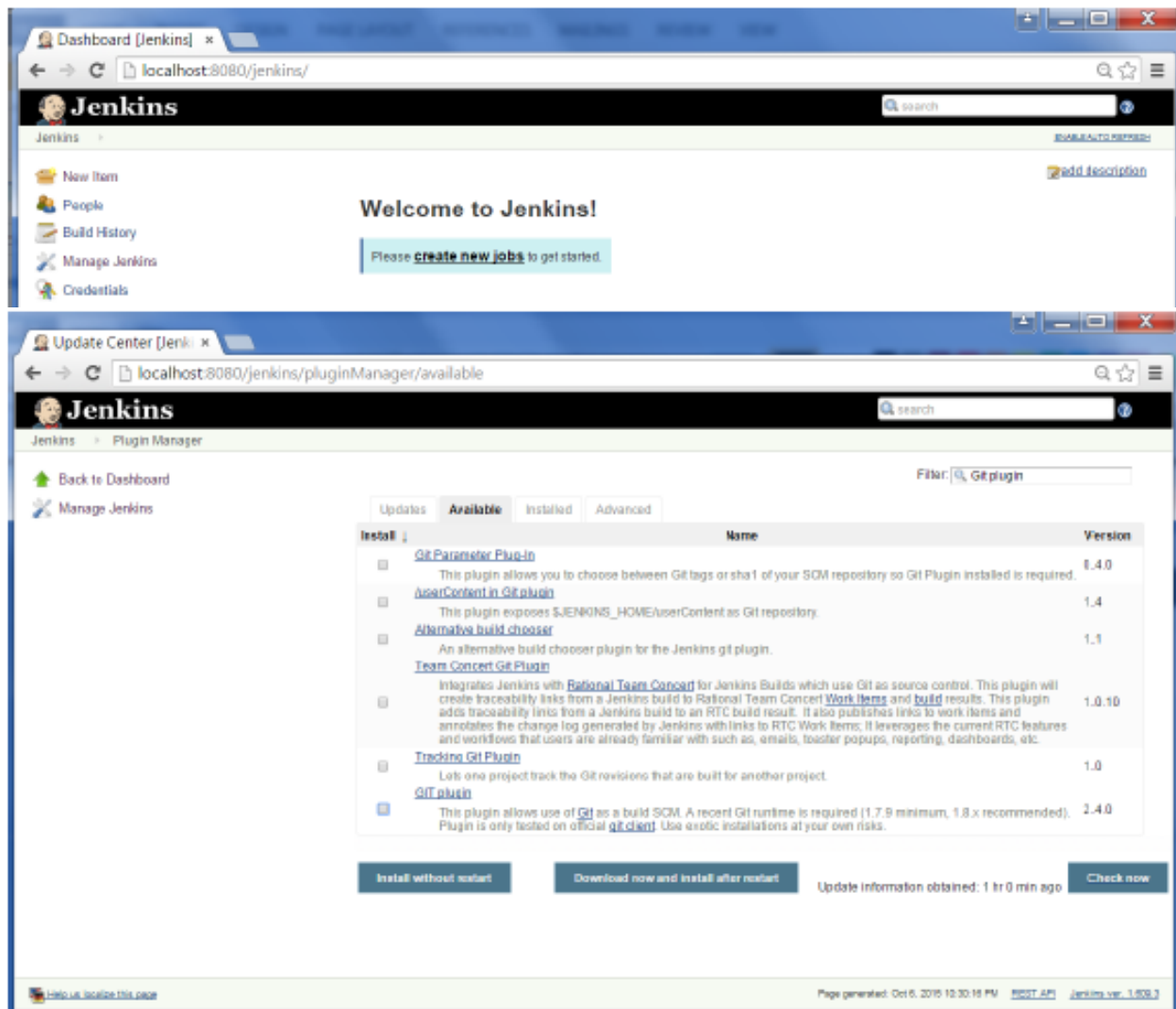
INFO: Beginning extraction from war file

Once the processing is complete without major errors, the following line will come in the output of the command prompt.

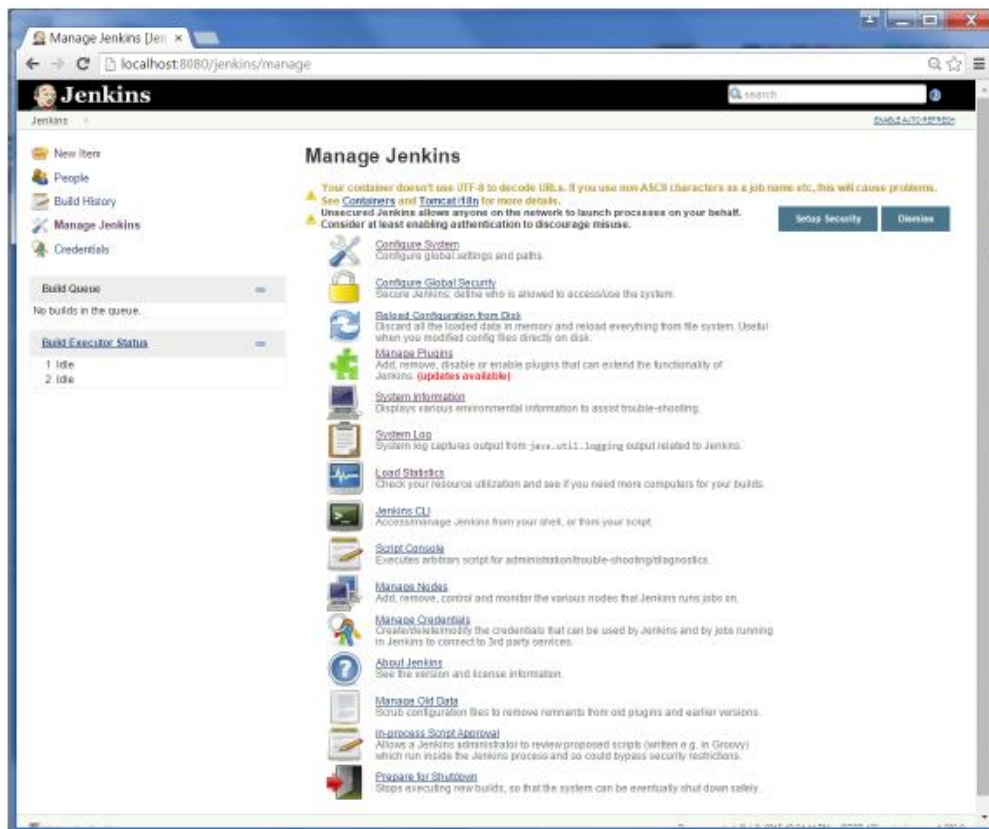
INFO: Jenkins is fully up and running

Accessing Jenkins

Once Jenkins is up and running, one can access Jenkins from the link – **http://localhost:8080**

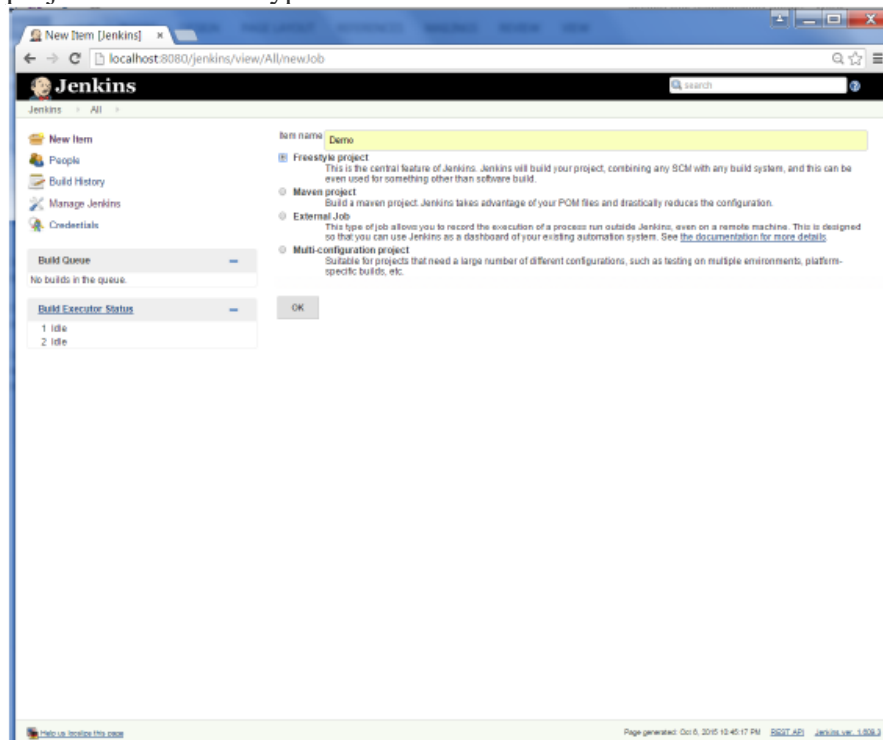


This link will bring up the Jenkins dashboard.



In the next screen, click the Available tab. This tab will give a list of plugins which are available for downloading. In the 'Filter' tab type 'Git plugin'

Once all installations are complete, restart Jenkins by issue the following command in the browser. **http://localhost:8080/jenkins/restart** After Jenkins is restarted, Git will be available as an option whilst configuring jobs. To verify, click on New Item in the menu options for Jenkins. Then enter a name for a job, in the following case, the name entered is 'Demo'. Select 'Freestyle project' as the item type. Click the Ok button.



INTRODUCTION ABOUT PROJECT

Our team has been involved in developing a 2D desktop game which includes a character: Dragon which flies infinitely in sky trying to avoid any kind of collision so that the Dragon achieves its goal of having maximum points. Avoiding the obstacle is the main target in this game otherwise game is over.

We have included controls through the space, pressing space will help dragon jump upwards, and releasing the space bar will make dragon fall downwards. The distance travelled is the score for the game. There are no brakes or moving back and the most thrilling feature will be that the speed increases as the distance increases making it difficult to handle the dragon.

Therefore, no brakes, increasing speed, collisions will be thrilling part of this game.

SRS:

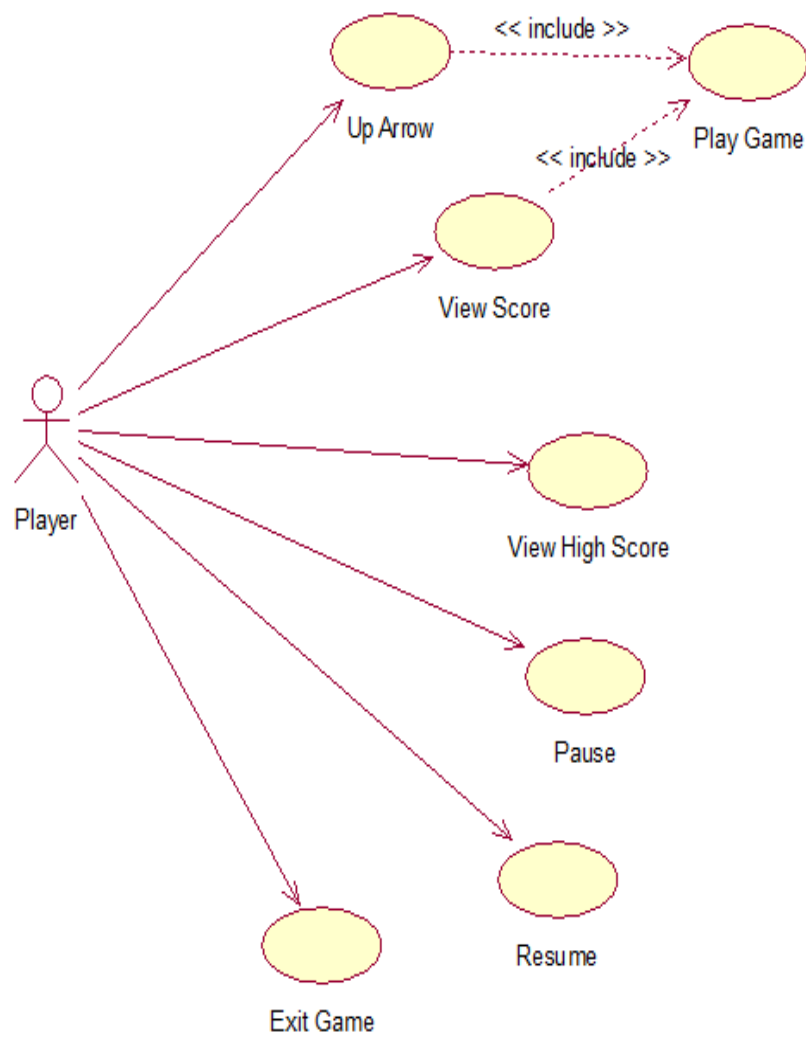
<u>UserStory</u>	<u>Description</u>	<u>Priority Estimate</u>
Desktop game	As a owner, I want 2D game So that player can play it in window os.	high
3rd Person Camera	As a player, I want 3rd person camera	high
Infinte Flying	As a player, I want infinite area to fly So that i can fly infinitely.	high
Dragon	As a player, I want Dragon So that I can fly.	high
Controls	As a player, I want control of actor with arrow key so that i can control it's movement (left or right).	medium
Increase Speed	As a player, I want speed increase periodically of skating So that game can be more challenging.	medium
Score Board	As a player, I want to see my score displaying on screen while playing So that i can see how much distance i have traveled.	medium
Obstacle	As a player, I want obstacle in game so that game can be more challenging.	medium
Pause	As a player, I want pause feature So that i can pause game when i want to.	medium
Game Over	As a player, I want game over when i collide to any obstacle.	medium
Restart	As a player, I want restart feature so that when game is over i can restart the game if i want to.	medium
User Interface	As a player, I want user interface So that I can select option like start, exit, setting etc when i open application.	medium
Start	As a player, I want start button so that i can start game when i click on it.	medium
Resume	As a player, I want resume feature So that i can resume game from where i paused it.	medium
Top - 03 Score	As a player, I want history of last top 03 score So that I can see them whenever i want to.	medium
Setting	As a player, I want setting feature So that i can control background music and sounds.	medium
Exit	As a player, I want exit feature so that i can exit game.	medium
Background Music	As a player, I want Background music.	low
Sound	As a player, I want sound produced when i collide to obstacle.	low

OBJECTIVE-4

Aim: Use-case Diagram of final project.

Use-case diagram helps us determine the functionality and features of the software from users prospective. A use case describes how a user interacts with the system by defining steps required to accomplish a specific goal.

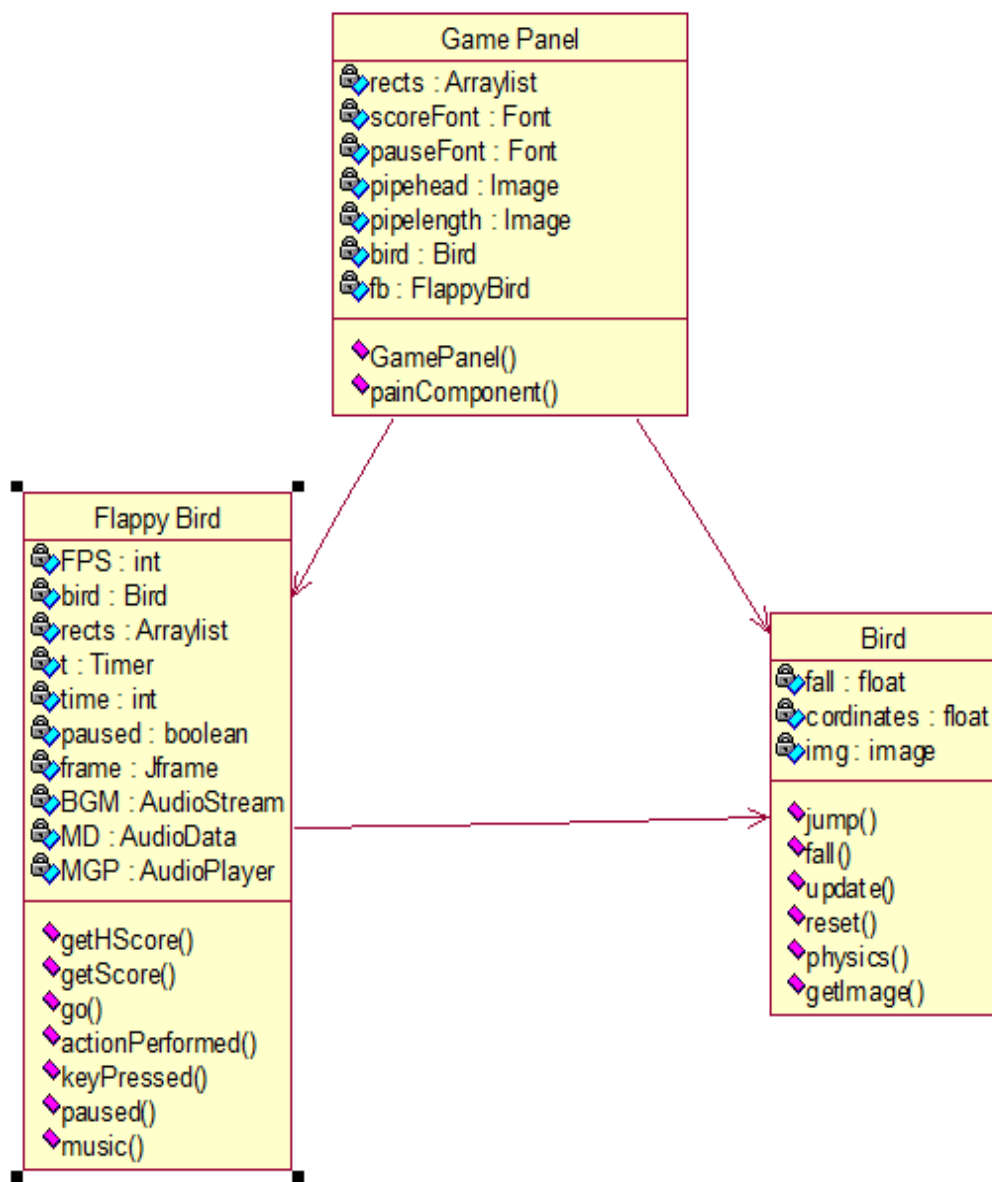
Use Case selection	Description
Use Case name	Adding Player and recording the score.
Primary Actor	Player: wants to Play the game.
Pre-condition	Player should know how to play and have read all the instructions.
Post-condition	New score matched with high score and maximum score is recorded as high score.
Main success scenario	<ol style="list-style-type: none">1. Player clicked the up arrow.2. Player Played the game.3. Player can see his score when the game is over.



OBJECTIVE-5

Aim: Class Diagram of final project.

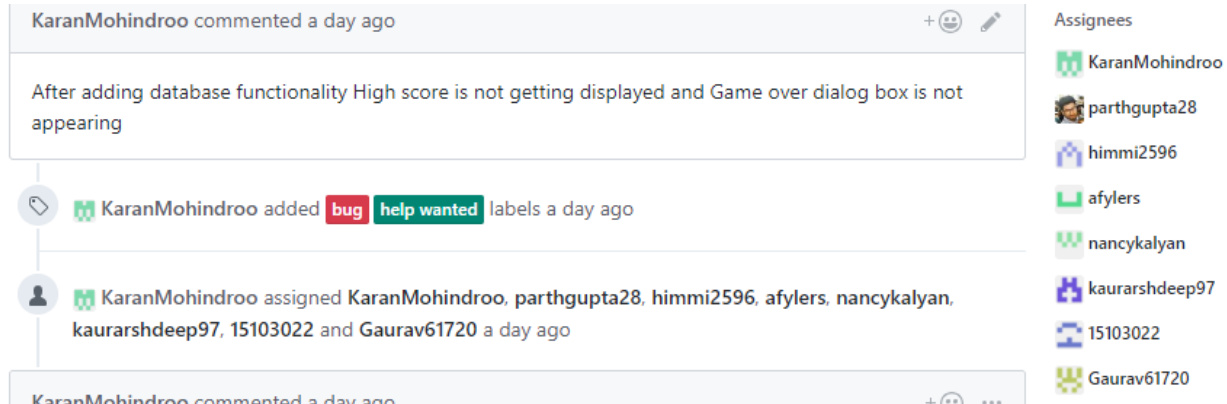
The class diagram is the main building block of object-oriented modeling. It is used for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code.



OBJECTIVE-6

Aim: Team Division for the project.

The development for its own easiness is divided into three teams; Programming team, Graphics team and Testing team.



PROGRAMMING TEAM

Programming team has been responsible for developing all the functions of the character as well as the environment.

The programming team would be responsible to make the dragon move, increase the speed, obstacles appear.

MEMBERS

1. Karan Mahindroo
2. Parth Gupta
3. Himakshi Salhotra

GRAPHICS TEAM

Graphics team has been responsible for developing a good and tasty visual effects. The team has been involved in developing sky, obstacle, background, dragon.

MEMBERS

1. Anjali
2. Arshdeep
3. Nancy

TESTING TEAM

Testing team has been responsible for figuring out any errors or bugs found in the program. They have been involved in testing the game from every way possible for a friendly user experience.

1. Ramandeep Singh
2. Gaurav Singh

OBJECTIVE-7

Aim: Sprint Division for the project.

Objective

To create functions needed for the production of 2D Dragon fly Game following agile methodology for software development.

Goals

To chain graphics team work with the programming team work. Here we import the graphical modules and map them to the functionalities created by the programming team.

Project Outline

Here we have divided the work in three sprints. In each of the sprints it has been primary job is to make the character run forward infinitely, create some obstacle, make the character go up and down to avoid obstacles and increase the speed gradually. We have used Eclipse as an Environment providing tool to program the game in java.

SPRINTS

The game was completed in 3 sprints each sprint of 1 week.

SPRINT 1 :

Description	Sprint	Status
As a user i want control of dragon with arrow keys So that i can move my dragon left or right to avoid obstacles.	Sprint 1	Completed
<u>Obsatcles</u>	Sprint 1	Completed
As a player, I want actor to be dragon.	Sprint 1	Completed
As an owner, I want 2D game So that player can play it in Windows OS	Sprint 1	Completed
As a player, I want dragon to be visible from top view.	Sprint 1	Completed

SPRINT 2 :

Work Description	Sprint	Status
Background <u>musics</u> and other sounds.	Sprint 2	Completed
Increase in speed with the distance covered.	Sprint 2	Completed
Exception <u>handeling</u> when the bird is loaded	Sprint 2	Pending
<u>Animation</u> of dragon	Sprint 2	Completed
Displaying Score at the end.	Sprint 2	Completed
overall Integration of the work	Sprint 2	Completed

SPRINT 3 :

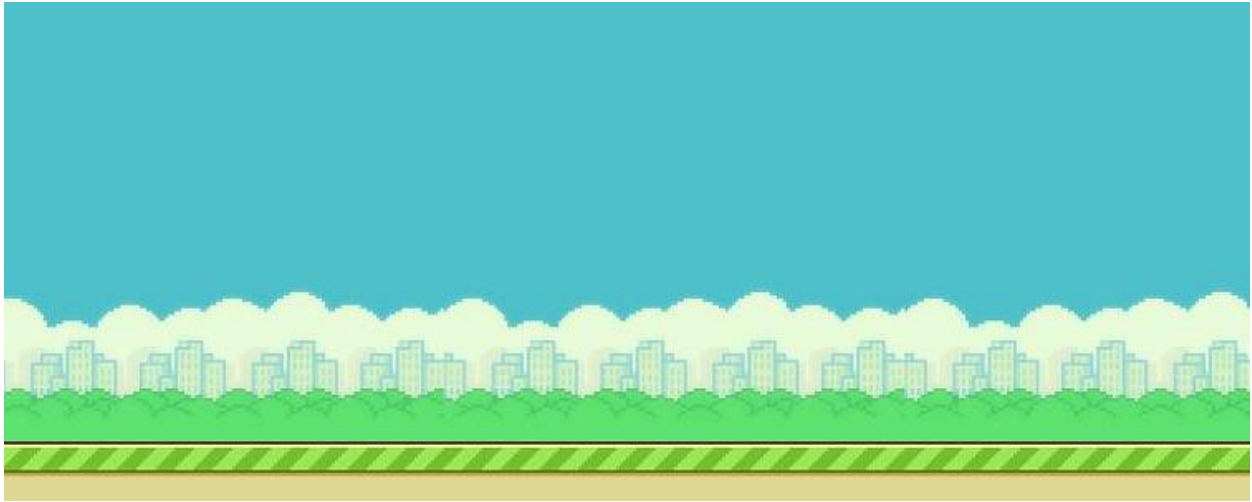
Work Description	Sprint	Status
Database Functionality for top 10 <u>highscore</u>	Sprint 3	Completed
Restart option when game over	Sprint 3	Pending
Performance Testing	Sprint 3	Completed
UI and Feature Testing	Sprint 3	Completed
Installation Testing	Sprint 3	Completed
Documentation	Sprint 3	Completed
Addition Of Background, Dragon Import	Sprint 3	Completed

OBJECTIVE-8

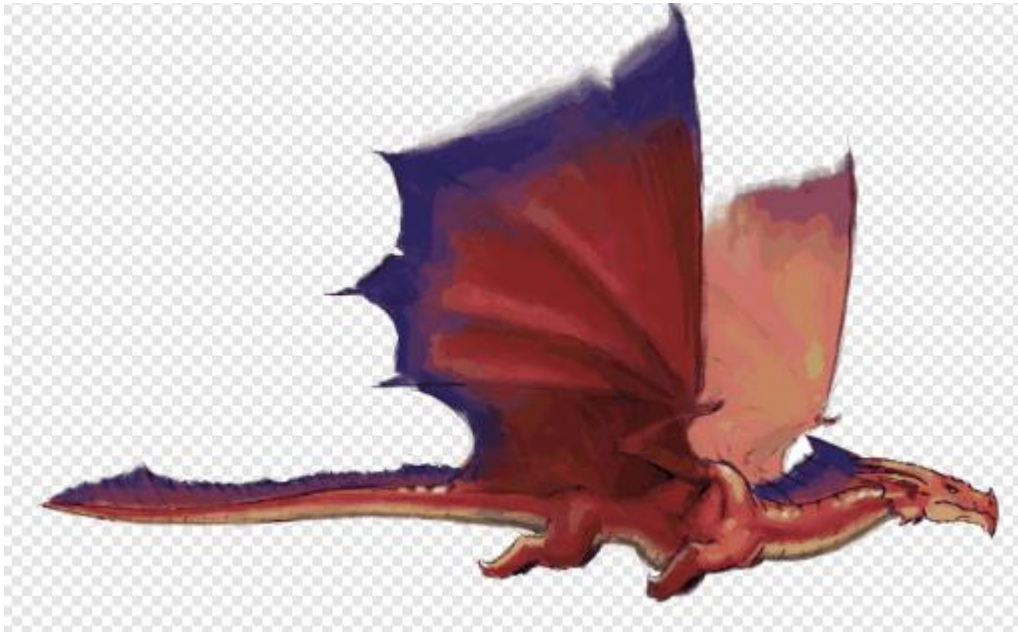
Aim: Screen shots of the Quick scrum tool and Spread sheet downloaded from the Quick Scrum.

GAME SCREENSHOTS:

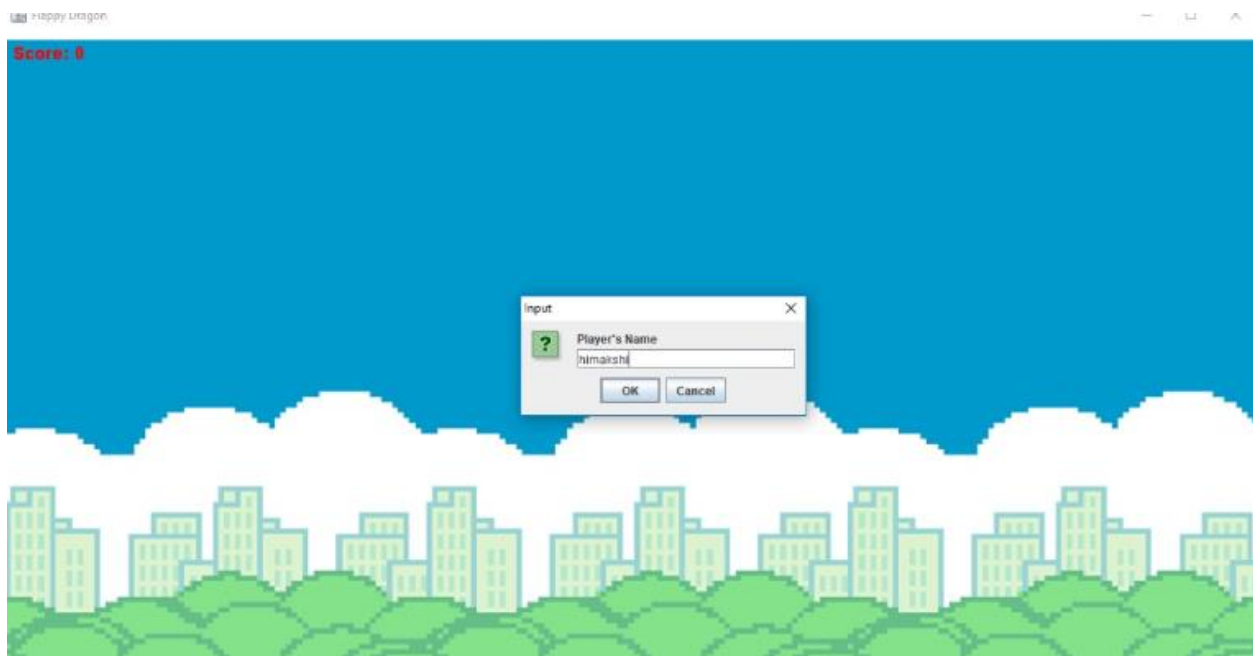
Background:



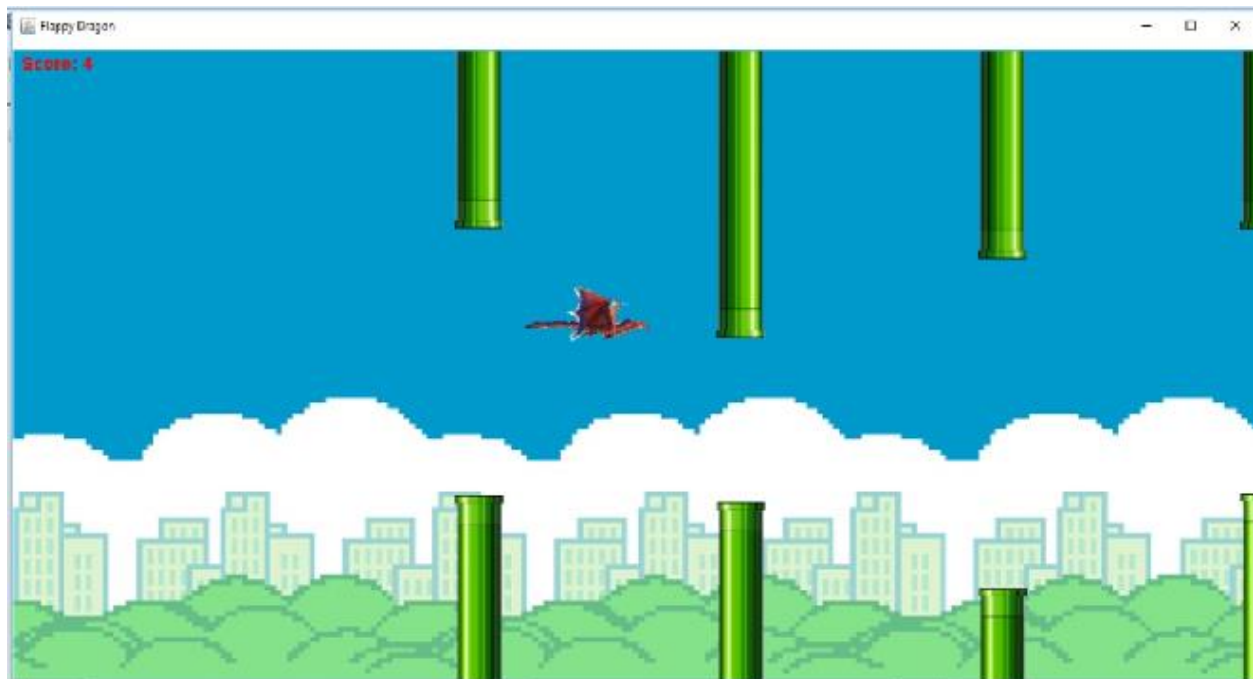
Dragon:



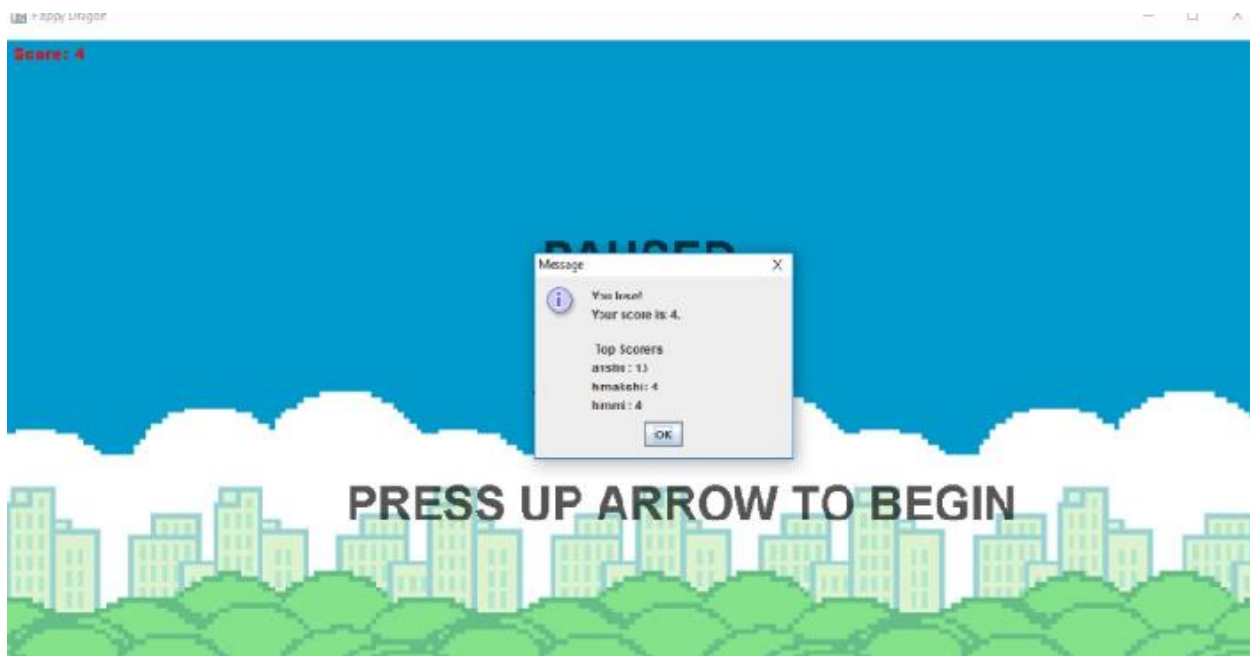
Start Screen:



While playing:



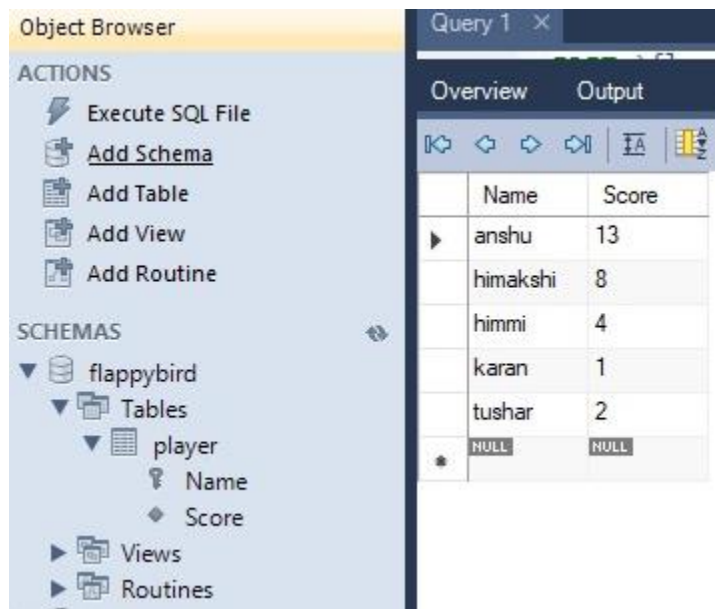
Game Over:



Enter Name to save scores:

	Name	Score
▶	anshu	13
	himakshi	8
	himmi	4
	karan	1
	tushar	2
*	NULL	NULL

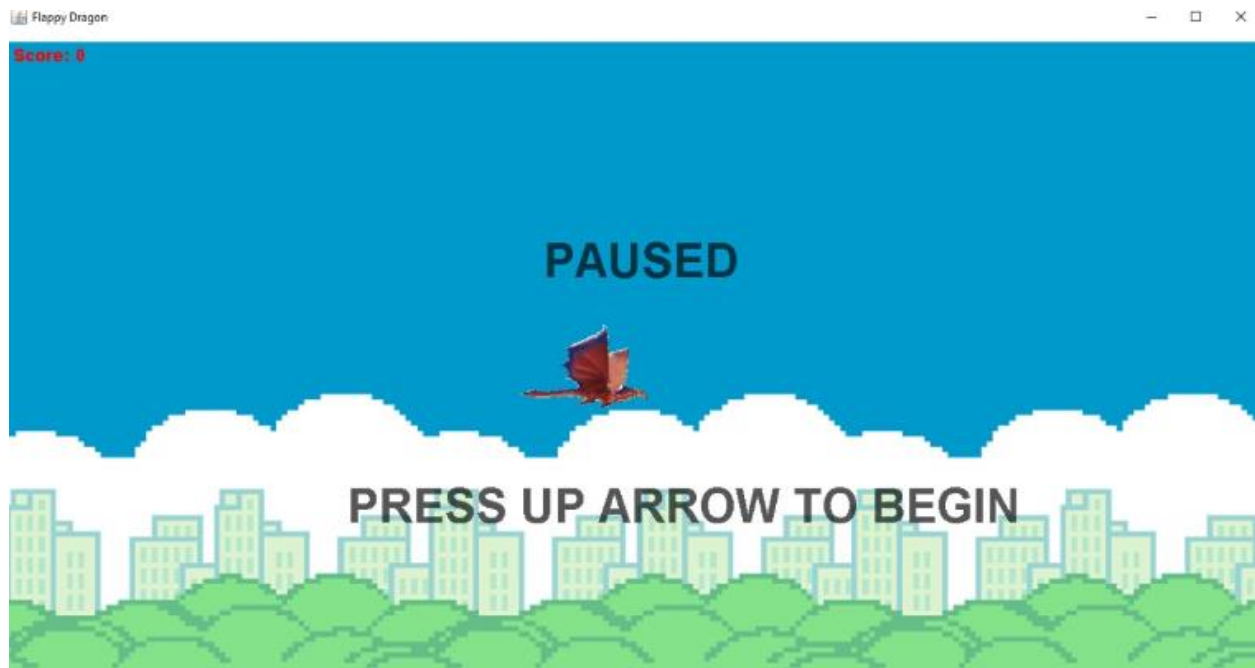
High Scores :



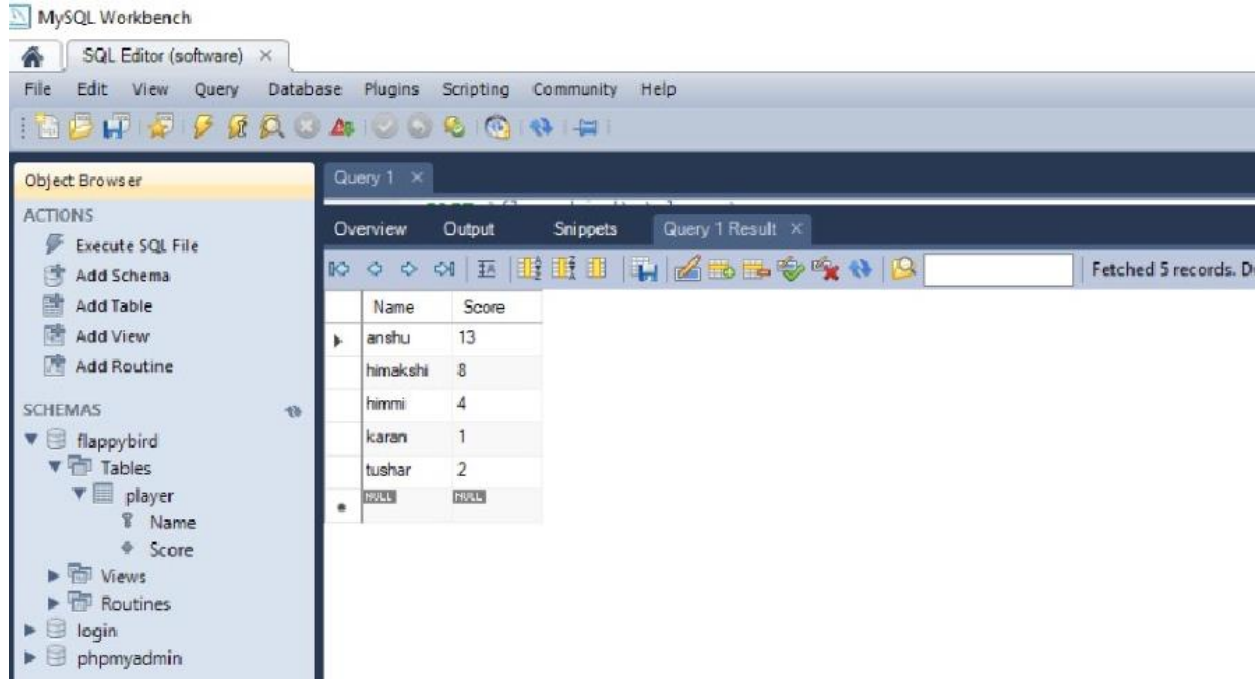
The screenshot shows a database application window. On the left is the 'Object Browser' pane with 'ACTIONS' (Execute SQL File, Add Schema, Add Table, Add View, Add Routine) and 'SCHEMAS' (flappybird, Tables, player, Views, Routines). The 'player' table is expanded, showing 'Name' and 'Score' fields. On the right is the 'Query 1' window with 'Overview' and 'Output' tabs. The 'Output' tab displays a table of high scores.

Name	Score
anshu	13
himakshi	8
himmi	4
karan	1
tushar	2
NULL	NULL

Restart:







Database:



Github Commits:


himmi2596 display 3 top scorers now		Latest commit 884cf9a a day ago
78px-Pipe.png	Resources	7 days ago
Bird.java	changed the value of fall variable	6 days ago
Database.java	display 3 top scorers now	a day ago
Dragon.gif	Resources	7 days ago
FlappyBird.java	Song pauses with game	a day ago
GamePanel.java	Update GamePanel.java	2 days ago
README.md	Update README.md	6 days ago
pipe_part.png	Resources	7 days ago
snowp.png	Add files via upload	5 days ago
README.md		


Issues Raised:


<input type="checkbox"/>	4 Open ✓ 1 Closed	Author ▾	Labels ▾	Projects ▾	Milestones ▾	Assignee ▾	Sort ▾
<input type="checkbox"/>	High score and Game over bug help wanted #5 opened a day ago by KaranMohindroo						2
<input type="checkbox"/>	No Database Connectivity help wanted #4 opened 2 days ago by himmi2596						1
<input type="checkbox"/>	Background image is not covering full game screen help wanted #3 opened 5 days ago by KaranMohindroo						5
<input type="checkbox"/>	Collision bug #1 opened 7 days ago by KaranMohindroo						2


Background image is not covering full game screen #3


Open KaranMohindroo opened this issue 3 days ago · 5 comments

 KaranMohindroo commented 3 days ago



 KaranMohindroo assigned KaranMohindroo, parthgupta28, himmi2596, afylers, nancykalyan, kaurarshdeep97, IS103022 and Gaurav61720 3 days ago

 KaranMohindroo added the **help wanted** label 3 days ago


 KaranMohindroo commented 3 days ago · edited


we can change the code to

```
g.drawImage(img,0,0,1000,600,null);
```


on

<https://github.com/KaranMohindroo/Flappy-Bird/blob/master/GamePanel.java#L55>




 himmi2596 commented 3 days ago

Yeah that is why we have prevented the maximize of the window

 KaranMohindroo commented 3 days ago

Now it looks much better what's say?

 himmi2596 commented 3 days ago

Is the quality of picture preserved?

On May 4, 2018 15:35, "KaranMohindroo" <notifications@github.com> wrote:

Assignees

- KaranMohindroo
- parthgupta28
- himmi2596
- afylers
- nancykalyan
- kaurarshdeep97
- IS103022
- Gaurav61720

Labels

- help wanted**

Projects

- None yet









Milestone

- No milestone

Notifications

- Unsubscribe**
- You're receiving notifications because you were assigned.

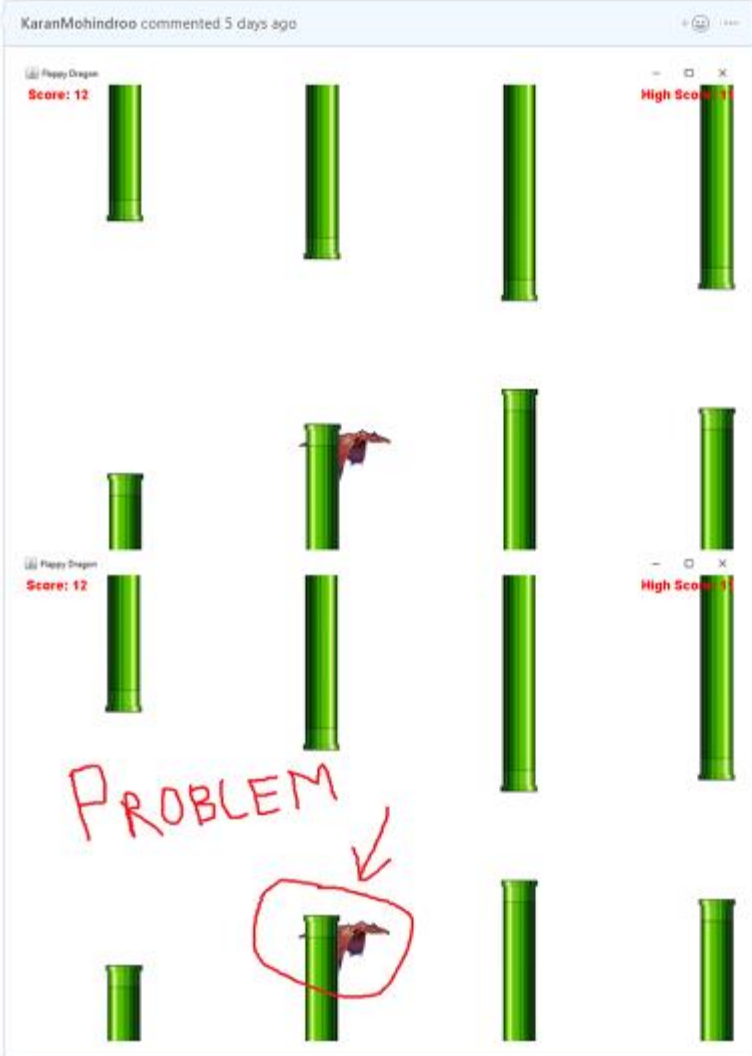
Participants

-        

Lock conversation

KaranMohindroo added the **bug** label 5 days ago

KaranMohindroo commented 5 days ago



PROBLEM

KaranMohindroo removed their assignment 5 days ago

himmi2596 commented 4 days ago

I have modified the collision code maybe it is working better than before

```

if(r.contains(bird.x+60, bird.y+50)) { JOptionPane.showMessageDialog(frame, "You lose!\n" + "Your score was: " + time/60 + "."); game = false; if(hs<time/60) { hs=time/60; } } if(r.contains(bird.x-60, bird.y-50)) { JOptionPane.showMessageDialog(frame, "You lose!\n" + "Your score was: " + time/60 + "."); game = false; if(hs<time/60) { hs=time/60; } }

```

<https://github.com/KaranMohindroo/Flappy-Bird/blob/master/FlappyBird.java#L81>

Issues Fixed :

2 Open 1 Closed

Author Labels Projects

Drogon Crosses the Upper/Lower boundary Fixed

#2 by KaranMohindroo was closed 5 days ago

OBJECTIVE-9

Aim: Screen shots of the Quick

TESTING

UI And Feature Testing:

Introduction

Test plan documents the strategy that will be used to verify and ensure that a product or system meets its design specifications and other requirements. This document contains guidelines and direction that will assist designated staff and personnel involved in testing in completing their task.

Unit testing

Unit testing, also known as component testing, refers to tests that verify the functionality of a specific section of code, usually at the function level. In an object-oriented environment, this is usually at the class level, and the minimal unit tests include the constructors and destructors.

These types of tests are usually written by developers as they work on code (white-box style), to ensure that the specific function is working as expected. One function might have multiple tests, to catch corner cases or other branches in the code.

Integration testing

Integration testing is any type of software testing that seeks to verify the interfaces between components against a software design. Software components may be integrated in an iterative way or all together ("big bang"). Normally the former is considered a better practice since it allows interface issues to be located more quickly and fixed.

Component interface testing

Component interface testing is a variation of **black-box testing**, with the focus on the data values beyond just the related actions of a subsystem component.

The practice of component interface testing can be used to check the handling of data passed between various units, or subsystem components, beyond full integration testing between those units. Tests can include checking the handling of some extreme data values while other interface variables are passed as normal values. Unusual data values in an interface can help explain unexpected performance in the next unit.

System testing

System testing, or end-to-end testing, tests a completely integrated system to verify that it meets its requirements. For example, a system test might involve testing a logon interface, then creating and editing an entry, plus sending or printing results, followed by summary processing or deletion (or archiving) of entries, then logoff.

Roles and Responsibilities

Project Owner

The Project Owner is in charge of recruiting, staff supervision, and staff training. The Project Manager will also be responsible for test budgeting and test planning and the cohesive integration of test and development activities. Any acquisition of hardware and software for test environment must also be approved by the Project Manager. The Project Manager is also required to coordinate meetings and keep track of the progress of the testing as well as ensuring that test-product documentation is complete.

Lead Designer

The Lead Designer design and develop all testing scenarios and procedures. The Lead Designer will be in charge of training new testers the procedures for bug and status reporting. The Lead Designer will also need to be able to identify the best ways to leverage a test tool on the project and to review test reports.

Lead Programmer

The Lead Programmer is in charge of the technical aspect of leadership for the testing. The Lead Programmer will need to be able to verify the quality of the requirements, including testability, requirement definition, test design, test-script and test-data development, test automation, test-environment configuration; test-script configuration management, and test execution. The Lead Programmer will help train new testers to use existing test tools.

Designer

Designers will execute automated test cases using test scripts designed by the programmers as well as manual tests. Designers will also need to prepare test reports which will be reviewed by the Lead Designer.

Programmer

Programmers will be in charge of maintaining test environment and creating automated scripts. Programmers will also be responsible for executing security, load and performance stress test. Programmers will also be in charge of preparing test reports which will be reviewed by the Lead Programmer.

Public Testers

Public Testers will be introduced to the project to provide external feedback from the consumers and usability testing.

Team Members:

- Parth
- Karan
- Himakshi
- Nancy
- Anjali
- Ramandeep
- Arshdeep
- Gaurav

Features To Be Tested

Check for background music and sound effects	Synchronization of sound & background music
----------------------------------------------------------	---------------------------------------------

User Interface	Check for animation, movement of character, graphics.
	There should not be any clipping (cutted background)
	Test whether one object overlaps with another
	Character should not move out of the screen/specified area
	Check for screen title
	Check for message title, message description, label (should be appropriate)
	Font displayed (color, size etc)
	Check other objects too (ex –Bottom pipe ,Top pipe etc)
Performance	Check the loading time of a game
	Make sure that any action is not taking considerable time, game flow should be fast
Score	score calculation
	Check the score registration functionality
Multitasking	Switch b/w different apps and play game , check for sound, score, UI, time-out etc
Functionality	Check game area, game logic
	play till game over.
	Synchronization of Dragon jump with the key when pressed.
	Check high score.
	Check whether scores are incremented with the distance
	Randomization of appearance of top pipe and bottom pipe.

Features Not To Be Tested

Interruption	If game is in running mode, then Check the behavior of interruption like Bluetooth, Infra red and CALL/SMS/MMS.
Upgrade the game / Battery effect	Upgrade of Games to the latest version and while migration all data should persist [score, user profile etc]
	What if Battery goes down/switched of the cell while playing, Whether the score will get saved?
Check for time format	Change the device time , format etc

Why are they not tested.

- Somethings are not required to be re-tested again (i.e when patches are done or expansions to an existing product.

Risk and Mitigation

Data corruption / Loss	Low	- High scores are updated whenever a new score is added.
System crash	High	-Game does not crash but sometimes certain frames do not load
Hardware malfunction or breakdown	High	- Have spare hardwares for temporary use while it gets repaired. - If no spare, get it replaced and have it back to running immediately. - Softwares installation files are kept for quick hardware restoration.
Task Delay	High	- Pinpoint problem origin and solve it immediately - Require tester to sign task completion report form
Unexpected absence of staff	Medium	- Enlist a temporary substitute employee through specialist or freelance. - If unable, director are to temporarily fill in the work. - All staff are to weekly document their work in preparation for their substitutes.
Insufficient tester	Low	- Project leads are to temporarily fill in the work - Hire 3rd party tester or freelance
Insufficient skilled tester	Low	- Make sure testers go through tester training briefing

INSTALLATION TESTING:

Objective: To test and report the bug and issues to the graphics team and programming team regarding models and elements needed for the production of 2D Dragon Fly Game for G1 agile lab.

Goals: To achieve a nearly bug free game by testing all the requirements and functionalities of the game.

Outline: Infinity 2D Game is a project being made during the course of Agile Lab. The group is of 3 teams consisting of Testing, Graphics, and Programming team. The testing team is required to test the game functionalities and all the other things. We have used jenkins and manual testing to fulfil the requirements.

Installation Test Plan

Product Name:	Dragon fly 2D Game		
Product Version or Build:	Dragon fly 2D Game v1		
Product Owner :	Parth Gupta		
Created On:	2 May 2018		
Review On:	3 May 2018		
Review By:	Sanehadeep Mam		
Current Version:	Version1		
Change Details:	Form a .exe file of the file		
Current Status:	Pending		
Signing Off Authority:	Name	Position	Signature, Date
	Parth Gupta	Product Owner Team Manager	Parth (3/05/2018)

<p>(1) Scope of Installation Test Plan:</p> <p>a) What does this document entail?</p> <p>b) What is being tested?</p> <p>c) What is the overall objective of this plan?</p>	<p>This documents entail about the installation testing result.</p> <p>It will be checked that the game installed on system successfully and working well as expected.</p> <p>User will not get any type of trouble while installing the game.</p>
<p>(2) Approach: Briefly provide the high-level description of the testing approach that enables to cost effectively meet the expectation stated in the Scope section.</p>	<p>In this we have generally done the manual testing and by manual testing we observe and record the report about the installation of the game.</p>
<p>(3) Installation Goals:</p> <p>1) What is response time for loading the game for the first time?</p> <p>2) What is acceptable response time?</p> <p>3) Should acceptable or not?</p>	<p>At most 2 seconds but in case Dragon object does not load,it takes restarting the application.</p> <p>5-10 sec</p> <p>yes</p>
<p>(4) Installation Testing Process, Final Report:</p> <p>Test team to baseline the load as under:</p>	

<p>1) Response Time each time starting the game</p> <p>2) Any web or database server errors as reported in the data log.</p> <p>The Final Report:</p>	<p>5-10 second(changes system to system) and if dragon does not load needs restarting.</p> <p>No error reported yet. The game is working smoothly once started.</p> <p>The executable file of the game is working smoothly.</p>
(5) Bug Reporting and Regression Instructions:	Restart manually if the dragon object does not load.
<p>(6) System / Test Environment:</p> <p>Window :</p> <p>Graphics :</p> <p>Software :</p> <p>Ram :</p>	<p>Windows 10</p> <p>NVIDIA GeForce GT 540M graphics card.</p> <p>Eclipse</p> <p>8GB</p>
<p>(7) Exclusions:</p> <p>a) Compatibility Testing</p>	Compatible with all type of Operating System upon which Eclipse run smoothly.
<p>(8) Test Deliverables:</p> <p>1) Final report</p>	Programmers have made .exe file.
<p>(9) Team Members</p> <p>Responsibilities :</p>	<p>Ramandeep Singh</p> <p>To do the installation Testing of the game</p>

PERFORMANCE TESTING:

Performance Test Plan

Product Name:	Dragon fly 2D Game		
Product Version or Build:	Dragon fly 2D Game v1		
Product Owner :	Parth Gupta		
Created On:	2 May 2018		
Review On:	3 May 2018		
Review By:	Sanehadeep Mam		
Current Version:	Version1		
Change Details:	Form a .exe file of the file		
Current Status:	Pending		
Signing Off Authority:	Name	Position	Signature, Date
	Parth Gupta	Product Owner Team Manager	Parth (3/05/2018)

(1) Scope of Load & Performance Test Plan: a) What does this document entail? b) What is being tested? c) What is the overall objective of this plan?	 This documents entail about the performance testing result. The performance of the game is being tested. How much memory it consumed. How much requirement it needs. How stable is it. Overall objective is to give the product owner a report about this testing and to give them a fine game.
(2) Approach: Briefly provide the high-level description of the	In this we have generally done the manual testing and by manual testing we observe and record the

testing approach that enables to cost effectively meet the expectation stated in the Scope section.	report about the performance of the game.
3) Performance / Capability Goals: 4) What is response time for loading the game for the first time? 5) What is acceptable response time? 6) Should acceptable or not?	5-10 Sec (Changes system to system) 5-10 Sec Acceptable
(4) Load Testing Process, Final Report: The internal test team has executed all created scripts. These Scripts will be generated and executed against the system at least three times. Test team have executed these scripts again, after subsequent hardware, software, or other fixes are introduced. Test team to baseline the load as under: 3) Response Time each time starting the game 4) Any web or database server errors as reported in the data log. 5) Memory consumption by the game. 6) Stability of the game 7) Speed	5-10 second(changes system to system) No Normal Moderate (Game is unable to load dragon object sometimes when it is started.) Speed goes on increasing until dragon hits any obstacle and the game is over.

The Final Report:	We have found the graphics related error and some errors during restarting the game.
(5) Bug Reporting and Regression Instructions:	When the game is started, sometimes it is unable to load the dragon object and when it is restarted the same error occurs. Try to resolve these errors.
(6) Testing Tools Used: 1) Build	Jenkins(For build the file) We have done the manual testing.
(7) System / Test Environment: Window : Graphics : Software : Ram :	Windows 10/8.1 NVIDIA GeForce GT 540M graphics card. Eclipse 8GB

(8) Exclusions: a) Compatibility Testing	Compatible with all type of Operating System upon which Eclipse run smoothly.
(9) Test Deliverables: 1) Performance related error 2) Functional bug reports 3) Final report	Memory and speed related errors. Speed was not increasing according to distance. All the issues regarding speed were resolved. The game runs smoothly fulfilling all the functionalities.
(10) Team Members Responsibilities :	Gaurav Singh To do the Performance Testing of the game

