

A
PROJECT
ON
“Super Mario 2.0”

Submitted to

Shiksha Mandal's
G. S. COLLEGE OF COMMERCE & ECONOMICS, NAGPUR
(AUTONOMOUS)
In the Partial Fulfillment of

B.Com. (Computer Application) Final Year

Submitted by
Mandar Kawade
Sanket Godhe

Under the Guidance of
Pravin J. Yadao



Shiksha Mandal's
G. S. COLLEGE OF COMMERCE & ECONOMICS, NAGPUR
(AUTONOMOUS)
2021-2022

Shiksha Mandal's

**G. S. COLLEGE OF COMMERCE & ECONOMICS,
NAGPUR
(AUTONOMOUS)**

CERTIFICATE

(2021 - 2022)

This is to certify that Mr. Mandar Kawde & Mr. Sanket Godhe has completed their project on the topic of “Super Mario 2.0” prescribed by G. S. College of Commerce & Economics, Nagpur (Autonomous) for B.Com. (Computer Application) – Semester-VI.

Date:

Place: Nagpur

Pravin J. Yadao

Project Guide

External Examiner

Internal Examiner

ACKNOWLEDGEMENT

We take this opportunity to express our deep gratitude and whole hearted thanks to project guide Prof. Pravin Yadao, Coordinator for his guidance throughout this work. We are very much thankful to him for his constant encouragement, support and kindness.

We are also grateful to our teachers Prof. Rahul Tiwari, Prof. Sushma Gawande, Prof. Preethi Ragnar, Prof. Prajakta Deshpande and Prof. Haresh Naringin for their encouragement, help and support from time to time.

We also wish to express our sincere thanks to Principal Dr. N. Y. Khandi for providing us wide range of opportunities, facilities and inspiration to gather professional knowledge and material without which this project could not have been completed.

1. Mandar Kawde
2. Sanket Godhe

Date:

Place: Nagpur

DECLARATION

We **Mandar Kawade & Sanket Godhe** hereby honestly declare that the work entitled “**Super Mario 2.0**” submitted by us at G. S. College of Commerce & Economics, Nagpur (Autonomous) in partial fulfillment of requirement for the award of B.Com. (Computer Application) degree by Attractant Tukituki Maharaj, Nagpur University, Nagpur has not been submitted elsewhere for the award of any degree, during the academic session 2021-2022.

The project has been developed and completed by us independently under the supervision of the subject teacher and project guide.

1. Mandar Kawde
2. Sanket Godhe

Date:

Place: Nagpur

INDEX

Sr. No	Particular	Page no.	Remarks	Signature
1.	INTRODUCTION	6-9		
2.	OBJECTIVES	10-12		
3.	PROJECT CATEGORY	13-17		
4.	SOFTWARE AND HARDWARE REQUIREMENT SPECIFICATIONS	18-19		
5.	DETAILED SYSTEM ANALYSIS 5.1. Data Flow Diagram 5.2. Structure of Application 5.3 Data Tables			
6.	SYSTEM DESIGN 6.1. Form Design 6.2. Source Code 6.3. Input And Output Screen	20-27		
7.	TESTING AND VALIDATION	28-29		
8.	SYSTEM AND SECURITIES MEASURES	30-31		
9.	IMPLEMENTATION, EVALUATION AND MAINTENANCE	32-35		
10.	FUTURE SCOPE OF PROJECT	36-37		
11.	CONCLUSION	38-39		
12.	BIBLIOGRAPHY	40-41		
13.	APPROVED COPY OF SYNOPSIS	42-43		

INTRODUCTION

Introduction:-

This article is about the video game series. For the character, see Mario. For the media franchise, see Mario (franchise). For other uses, see Super Mario (disambiguation).

Super Mario is a 2D and 3D platform game series created by Nintendo based on and starring the fictional plumber Mario. Alternatively called the Super Mario Bros. Series or simply the Mario series, it is the central series of the greater Mario franchise. At least one Super Mario game has been released for every major Nintendo video game console. There are over twenty games in the series.

The Super Mario games are set primarily in the fictional Mushroom Kingdom, typically with Mario as the player character. He is usually joined by his brother, Luigi, and often by other members of the Mario cast. As platform games, they involve the player character running and jumping across platforms and atop enemies in themed levels. The games have simple plots, typically with Mario and Luigi rescuing the kidnapped Princess Peach from the primary antagonist, Bowser. The first game in the series, Super Mario Bros., released for the Nintendo Entertainment System (NES) in 1985, established the series' core gameplay concepts and elements. These include a multitude of power-ups and items that give the character special powers such as fireball-throwing and size-changing.

The Super Mario series is part of the greater Mario franchise, which includes other video game genres and media such as film, television, printed media, and merchandise. More than 380 million copies of Super Mario games have been sold worldwide, making it the fourth-bestselling video game series, behind the larger Mario franchise, the puzzle series Tetris, and first-person shooter series Call of Duty.

Super Mario Bros., the first side-scrolling 2D platform game to feature Mario, was released for the Nintendo Entertainment System (NES) in 1985. It was derived through collaboration by Nintendo's Shigeru Miyamoto and Takashi Tezuka as a successor to the 1983 arcade game Mario Bros., which starred two characters: Mario, the titular character that first appeared in Donkey Kong as the original player character and its sequel where he was a final boss, and Luigi, who first appeared in Mario Bros.[3] Super Mario Bros. established many core Mario elements, such as Goombas, Koopa Troopas, Bowser, Peach, and its three power-ups: the Super Mushroom, increasing the character's size and providing an extra hit point, Fire Flower, allowing the character to throw fireballs as weapons, and

Super Star, granting temporary invincibility. The etymology of "Super" in the title came from the integration of the Super Mushroom into the game. The brothers Mario and Luigi must rescue Princess Toadstool/Peach from Bowser/King Koopa in the Mushroom Kingdom. The game consists of eight worlds of four levels each, totaling 32 levels altogether. Though the worlds differ in themes, the fourth level is always a fortress or castle that ends with a fight against Bowser (or one of his minions disguised as him). Super Mario Bros. is one of the best-selling video games of all time. Super Mario Bros. 2 in Japan is the first sequel to the original Super Mario Bros. It uses the Super Mario Bros. Engine, with additions such as whether, character movements, and more complex levels, altogether yielding a much higher difficulty. The game follows the same style of level progression as Super Mario Bros., with eight initial worlds of four levels each. At that time, this sequel was not released outside Japan since Nintendo of America did not want the Super Mario series to be known to players outside of Japan for frustrating difficulty. It remained inaccessible to a steadily broadening market of American video game players, becoming stylistically outdated by the time the Japanese Super Mario Bros. 2 could be eventually delivered to America. The game later debuted

outside Japan in 1993 as “Super Mario Bros.: The Lost Levels” in the compilation game Super Mario All-Stars for the Super Nintendo Entertainment System (SNES).

In Super Mario Bros. 2 (Super Mario USA in Japan), Mario and his companies seek to defeat the evil frog Wart in the Subcon dreamland. Based on a discarded prototype, the game was instead originally released as Yume Kōji: Doki Doki Panic in Japan, and was ultimately converted into a Mario game for the rest of the world as Super Mario Bros 2, before being released in Japan as Super Mario USA as part of Super Mario All-Stars. One of the game’s most defining aspect is the four player characters: not only Mario, but Luigi, Princess Peach and Toad are available for single-player gameplay, each with define character movements: Luigi jumps higher, the princess can hover in the air for a short amount of time, and Toad is the fastest. Characters here also can pluck items from the ground to throw at enemies. This also the first Super Mario game to use a life meter, which allows the characters to be hit up to four times before dying.

Super Mario Bros 3.is divided into eight themed worlds, each with 6-10 levels and several bonus stages displayed as locations on a mapped overworld.

OBJECTIVE

Objective:-

1. Provide Platform:- We provide a platform to the user to have access to all three games under a single app its helps to save the time of the user in searching of games and all these games are easy to use.
2. Entertainment:- we have developed this application with the objective of providing a source of entertainment to our user that will not only help to pass their free time but they can also gain some useful knowledge.
3. Quick Access:- The user can easily access the games because all three games are available under one single app. Its interface is easy to use as well as user friendly designed and there is a different variety game that fulfils the user interest.
4. Time Saving:- User can easily save their time by using this app as it will reduce the efforts in searching of game made by the users as in this application user will find more than three games in future.
5. Easy to use:- our application is very easy to use as it does not require any special access to run a game user can easily enjoy it by clicking on its icon as there is no need to download any external application to run it.
6. Good designed:- This application is designed in a way to keep the user need fulfilled and make it easy for the user to access it and keep them attracted towards the application.

7.Improvement:- Improve user support through better customer service as if there is any problem in the application or user need any change in the application this improvement will be made by the application developers.

8.Analytical Skills :- Many games require the players to solve puzzles or problems in order to advance. These mario really challenge the players analytical and problem-solving skills, with some studies suggesting that it may make the players smarter.

PROJECT CATEGORY

PROJECT CATEGORY:-

To create an attractive gaming application, it is necessary to build a effective and accurate design with user friendly environment. This project logic is developed by using java language and designed through XML Code. We used android studio to develop this application.

LANGUAGE USED PYTHON: -

Python is a widely used general-purpose, high level programming language. It was created by Guido van Rossum in 1991 and further developed by the Python Software Foundation. It was designed with an emphasis on code readability, and its syntax allows programmers to express their concepts in fewer lines of code.

History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties

at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++,

Algol-68, Smalltalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

Python Features

Python's features include –

- Easy-to-learn – Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- Easy-to-read – Python code is more clearly defined and visible to the eyes.
- Easy-to-maintain – Python's source code is fairly easy-to-maintained.
- A broad standard library – Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- Interactive Mode – Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- Portable – Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- Extendable – You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- Databases – Python provides interfaces to all major commercial databases.
- GUI Programming – Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- Scalable – Python provides a better structure and support for large programs

than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below –

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

List of why Python is popular:

- The Python framework also has modules and packages, which facilitates code reusability.
- GUI Programming – Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- Scalable – Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below –

- It supports functional and structured programming methods as well as OOP.
 - It can be used as a scripting language or can be compiled to byte-code for building large applications.
 - It provides very high-level dynamic data types and supports dynamic type checking.
 - It supports automatic garbage collection.
 - It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.
- List of why Python is popular:• The Python framework also has modules and packages, which facilitates code reusability.

SOFTWARE AND HARDWARE
REQUIREMENT SPECIFICATIONS

Hardware

To play A Flappy Bird in Real Life you will need a minimum CPU equivalent to an Intel Core 2 Duo Q6867. A Flappy Bird in Real Life system requirements state that you will need at least 512 MB of RAM. The cheapest graphics card you can play it on is an NVIDIA GeForce 7200 GS.

- 4 GB RAM and Above
- 320 GB HARDDISK and Above
- Keyboard • Mouse
- Processor (CPU) with 2 gigahertz (GHz) frequency or Above
- Monitor Resolution 1024 * 768 or Above

SOFTWARE

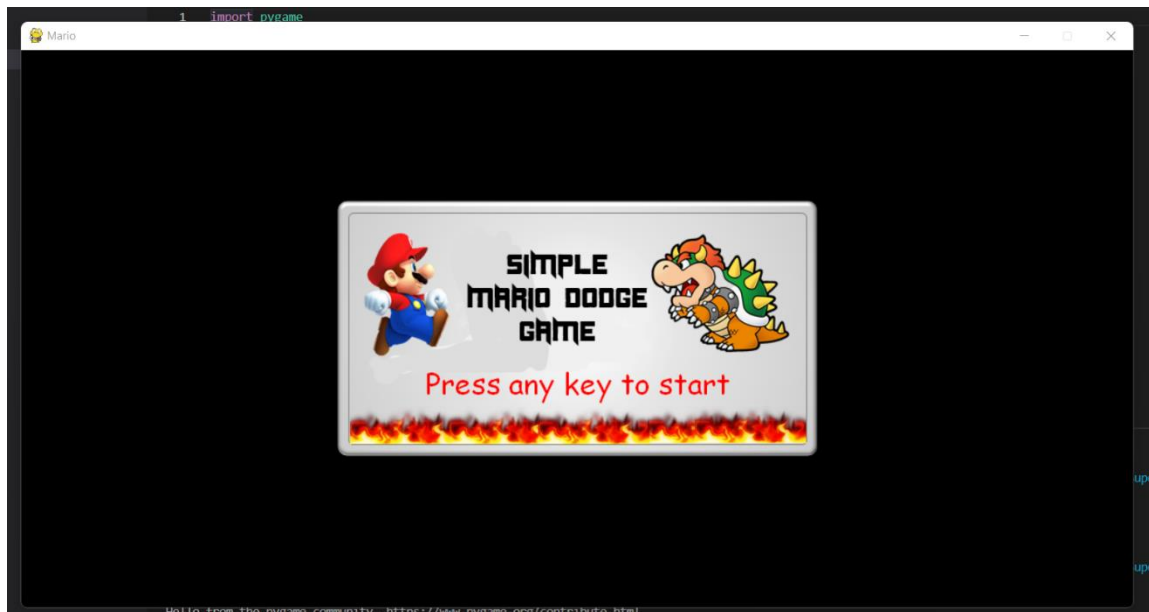
Software can be termed as the group of instruction or command used by the computer to accomplish the given task. In today's world generation of software is ever ending. It is an evolution of dignified technology.

- OPERATING SYSTEM: Windows 10
- LANGUAGES (FRONT END): Python

SYSTEM DESIGN

1.FORM DESIGN

GAME WINDOW



2.SOURCE CODE

```
import pygame
import sys
pygame.init()

WINDOW_WIDTH = 1200
WINDOW_HEIGHT = 600
FPS = 20
BLACK = (0, 0, 0)
GREEN = (0, 255, 0)
ADD_NEW_FLAME_RATE = 25
cactus_img = pygame.image.load('img/cactus_bricks.png')
cactus_img_rect = cactus_img.get_rect()
cactus_img_rect.left = 0
fire_img = pygame.image.load('img/fire_bricks.png')
fire_img_rect = fire_img.get_rect()
fire_img_rect.left = 0
CLOCK = pygame.time.Clock()
font = pygame.font.SysFont('forte', 20)

canvas = pygame.display.set_mode((WINDOW_WIDTH, WINDOW_HEIGHT))
pygame.display.set_caption('Mario')

class Topscore:
    def __init__(self):
        self.high_score = 0
    def top_score(self, score):
        if score > self.high_score:
            self.high_score = score
        return self.high_score

topscore = Topscore()

class Dragon:
    dragon_velocity = 10

    def __init__(self):
        self.dragon_img = pygame.image.load('img/dragon.png')
        self.dragon_img_rect = self.dragon_img.get_rect()
        self.dragon_img_rect.width -= 10
        self.dragon_img_rect.height -= 10
        self.dragon_img_rect.top = WINDOW_HEIGHT/2
        self.dragon_img_rect.right = WINDOW_WIDTH
        self.up = True
        self.down = False
```

```
def update(self):
    canvas.blit(self.dragon_img, self.dragon_img_rect)
    if self.dragon_img_rect.top <= cactus_img_rect.bottom:
        self.up = False
        self.down = True
    elif self.dragon_img_rect.bottom >= fire_img_rect.top:
        self.up = True
        self.down = False

    if self.up:
        self.dragon_img_rect.top -= self.dragon_velocity
    elif self.down:
        self.dragon_img_rect.top += self.dragon_velocity

class Flames:
    flames_velocity = 20

    def __init__(self):
        self.flames = pygame.image.load('img/fireball.png')
        self.flames_img = pygame.transform.scale(self.flames, (20, 20))
        self.flames_img_rect = self.flames_img.get_rect()
        self.flames_img_rect.right = dragon.dragon_img_rect.left
        self.flames_img_rect.top = dragon.dragon_img_rect.top + 30

    def update(self):
        canvas.blit(self.flames_img, self.flames_img_rect)

        if self.flames_img_rect.left > 0:
            self.flames_img_rect.left -= self.flames_velocity

class Mario:
    velocity = 10

    def __init__(self):
        self.mario_img = pygame.image.load('img/mario.png')
        self.mario_img_rect = self.mario_img.get_rect()
        self.mario_img_rect.left = 20
        self.mario_img_rect.top = WINDOW_HEIGHT/2 - 100
        self.down = True
        self.up = False

    def update(self):
        canvas.blit(self.mario_img, self.mario_img_rect)
        if self.mario_img_rect.top <= cactus_img_rect.bottom:
            game_over()
```

```
    if SCORE > self.mario_score:
        self.mario_score = SCORE
    if self.mario_img_rect.bottom >= fire_img_rect.top:
        game_over()
        if SCORE > self.mario_score:
            self.mario_score = SCORE
    if self.up:
        self.mario_img_rect.top -= 10
    if self.down:
        self.mario_img_rect.bottom += 10

def game_over():
    pygame.mixer.music.stop()
    topscore.top_score(SCORE)
    game_over_img = pygame.image.load('img/end.png')
    game_over_img_rect = game_over_img.get_rect()
    game_over_img_rect.center = (WINDOW_WIDTH/2, WINDOW_HEIGHT/2)
    canvas.blit(game_over_img, game_over_img_rect)
    while True:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                pygame.quit()
                sys.exit()
            if event.type == pygame.KEYDOWN:
                if event.key == pygame.K_ESCAPE:
                    pygame.quit()
                    sys.exit()
                game_loop()
        pygame.display.update()

def start_game():
    canvas.fill(BLACK)
    start_img = pygame.image.load('img/start.png')
    start_img_rect = start_img.get_rect()
    start_img_rect.center = (WINDOW_WIDTH/2, WINDOW_HEIGHT/2)
    canvas.blit(start_img, start_img_rect)
    while True:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                pygame.quit()
                sys.exit()
            if event.type == pygame.KEYDOWN:
                if event.key == pygame.K_ESCAPE:
                    pygame.quit()
                    sys.exit()
                game_loop()
        pygame.display.update()
```



```
def check_level(SCORE):
    global LEVEL
    if SCORE in range(0, 10):
        cactus_img_rect.bottom = 50
        fire_img_rect.top = WINDOW_HEIGHT - 50
        LEVEL = 1
    elif SCORE in range(10, 20):
        cactus_img_rect.bottom = 100
        fire_img_rect.top = WINDOW_HEIGHT - 100
        LEVEL = 2
    elif SCORE in range(20, 30):
        cactus_img_rect.bottom = 150
        fire_img_rect.top = WINDOW_HEIGHT - 150
        LEVEL = 3
    elif SCORE > 30:
        cactus_img_rect.bottom = 200
        fire_img_rect.top = WINDOW_HEIGHT - 200
        LEVEL = 4

def game_loop():
    while True:
        global dragon
        dragon = Dragon()
        flames = Flames()
        mario = Mario()
        add_new_flame_counter = 0
        global SCORE
        SCORE = 0
        global HIGH_SCORE
        flames_list = []
        while True:
            canvas.fill(BLACK)
            check_level(SCORE)
            dragon.update()
            add_new_flame_counter += 1

            if add_new_flame_counter == ADD_NEW_FLAME_RATE:
                add_new_flame_counter = 0
                new_flame = Flames()
                flames_list.append(new_flame)
        for f in flames_list:
            if f.flames_img_rect.left <= 0:
                flames_list.remove(f)
```

```
    SCORE += 1
    f.update()

for event in pygame.event.get():
    if event.type == pygame.QUIT:
        pygame.quit()
        sys.exit()
    if event.type == pygame.KEYDOWN:
        if event.key == pygame.K_UP:
            mario.up = True
            mario.down = False
        elif event.key == pygame.K_DOWN:
            mario.down = True
            mario.up = False
    if event.type == pygame.KEYUP:
        if event.key == pygame.K_UP:
            mario.up = False
            mario.down = True
        elif event.key == pygame.K_DOWN:
            mario.down = True
            mario.up = False

score_font = font.render('Score:'+str(SCORE), True, GREEN)
score_font_rect = score_font.get_rect()
score_font_rect.center = (200, cactus_img_rect.bottom + score_font_rect.height/2)
canvas.blit(score_font, score_font_rect)

level_font = font.render('Level:'+str(LEVEL), True, GREEN)
level_font_rect = level_font.get_rect()
level_font_rect.center = (500, cactus_img_rect.bottom + score_font_rect.height/2)
canvas.blit(level_font, level_font_rect)

top_score_font = font.render('Top Score:'+str(topscore.high_score), True, GREEN)
top_score_font_rect = top_score_font.get_rect()
top_score_font_rect.center = (800, cactus_img_rect.bottom + score_font_rect.height/2)
canvas.blit(top_score_font, top_score_font_rect)

canvas.blit(cactus_img, cactus_img_rect)
canvas.blit(fire_img, fire_img_rect)
mario.update()
for f in flames_list:
    if f.flames_img_rect.colliderect(mario.mario_img_rect):
        game_over()
    if SCORE > mario.mario_score:
        mario.mario_score = SCORE
pygame.display.update()

CLOCK.tick(FPS)
```

```
start_game()
```

TESTING AND VALIDATION
CHECK

Testing and validation check: Validation testing in software engineering is in place to determine if the existing system complies with the system requirements and performs the dedicated functions for which it is designed along with meeting the goals and needs of the organization. The process of evaluating software during the development process or at the end of the development process to determine whether it satisfies specified business requirements. Validation Testing ensures that the product actually meets the client's needs. It can also be defined as to demonstrate that the product fulfills its intended use when deployed on appropriate environment. Whenever any particular software is tested then the main motive is to check the quality against the defects being found. The developers fix the bugs and the software is rechecked to make sure that absolutely no bugs are left out in that. This not only shoots the product's quality but also its user acceptance.

- To ensure customer satisfaction
- To be confident about the product
- To fulfil the client's requirement until the optimum capacity
- Software acceptance from the end-user

Client-side validation is an initial check and an important feature of good user experience; by catching invalid data on the client-side, the user can fix it straight away. If it gets to the server and is then rejected, a noticeable delay is caused by a round trip to the server and then back to the client-side to tell the user to fix their data.

SYSTEM SECURITY MEASURES

System Security Measures:-

System security measures: Security of a computer system is a crucial task. It is a process of ensuring confidentiality. A system is said to be secure if its resources are used and accessed as intended under all the circumstances, but no system can guarantee absolute security from several of the various threats and unauthorized access. Security measures will be taken:

- **Strong passwords:** This first measure is taken that users may use special characters in their passwords and password length must be 8 characters.
- **Confidentiality:** If any user is sharing their personal details in login form it will be secure safely as only users can access such information.

IMPLEMENTATION , EVOLUTIONS
AND MAINTENANCE

IMPLEMENTATION , EVOLUTIONS AND MAINTENANCE

Implementation The software implementation stage involves the transformation of the software technical data package (TDP) into one or more fabricated, integrated, and tested software configuration items that are ready for software acceptance testing. The primary activities of software implementation include the:

- Fabrication of software units to satisfy structural unit specifications.
- Assembly, integration, and testing of software components into a software configuration item.
- Prototyping challenging software components to resolve implementation risks or establish a fabrication proof of concept.
- Dry-run acceptance testing procedures to ensure that the procedures are properly delineated and that the software product (software configuration items (CIs and computing environment) is ready for acceptance testing.

-

Evaluation:-

After the implementation phase, another stage in project development is Evaluation. Evaluation during a program's implementation may examine whether the program is successfully recruiting and retaining its intended participants, using training materials that meet standards for accuracy and clarity, maintaining its projected timelines, coordinating efficiently with other ongoing programs and activities, and meeting applicable legal standards. Evaluation during program implementation could be used to inform mid-course corrections to program implementation (formative evaluation) or to shed light on implementation processes (process evaluation). After keeping the project in the working condition for the some Time, all the errors that are showing in the computer program should be Removed. The programmer needs to correct them so that same errors should Not be repeated. We should also get the feedback from the user which are Using it and ask them whether, it is user friendly or not. After evaluating the Program and satisfying the needs of the user the program is maintained fully

Maintenance:-

Maintenance of software can include software upgrades, repairs, and fixes of the software if it breaks. Software applications often need to be upgraded or integrated with new systems the customer deploys. It's often necessary to provide additional testing of the software or version upgrades. During the maintenance phase, errors or defects may exist, which would require repairs during additional testing of the software. Monitoring the performance of the software is also included during the maintenance phase. Once the system is deployed, and customers start using the developed system, following 3 activities occur

Bug fixing – bugs are reported because of some scenarios which are not tested at all.

Upgrade – Upgrading the application to the newer versions of the Software.

Enhancement – Adding some new features into the existing software.

FUTURE SCOPE OF PROJECT

FUTURE SCOPE OF PROJECT:-

1. **Upload globally:-** This application is currently available for some of the users as it's not uploaded on any global platform till now so wide range of the user can not access it. So, looking towards this issue will try to upload on a global platform like google play store so that we can attract wide range of users.
2. **Addition of more game:-** Currently this application only contains limited numbers of game. It only contains 3 game currently but in near future developer will add many more additional games like car racing, puzzle, chess, etc. in it.
3. **Design Improve design:-** Our application is well designed and developed but if user wants to change the design theme of the gaming application, we will try to improve the design of the application as per the user satisfaction.
4. **Compatibility: -** We will make our project more effective and compatible with other android devices whose android version is less than lollipop 5.0 as for this project is only accessible for the devices with android lollipop 5.0 or more.
5. **Platform Independent: -** This project can only run on an android device in current situation so it is platform dependent for now, so as a future scope of the project we will try to make it platform independent and run it on other operating system like IOS, Windows.

CONCLUSION

CONCLUSION:-

- **Premium Quality:** The main motive behind developing this project is to provide good quality software.

- **Fulfilment of User Satisfaction:** The basic concern behind developing this software is to provide good quality of game and provide high level of satisfaction to the user.

- **Refreshing and Enhances Mind Power:** This project is made basically to refresh the mind of people as well as it also enhances the concentration power of the user.

- when games are played in moderation and with mindfulness, they are a viable source of stress relief as well as a catalyst for mental health improvement and development of social skills.

- We have created this application using python language which is very popular.

- Our application also has many pros and cons which will be improved as per the user's preferences.

BIBLIOGRAPHY & REFERENCES

Bibliography & References:-

- 1) <http://www.tutorialspoint.com/>
- 2) Stack Flow
- 3) www.Google.co.in
- 4) YouTube

**A
PROJECT SYNOPSIS
ON**

“Mario 2.0”

Submitted to

**G. S. COLLEGE OF COMMERCE & ECONOMICS, NAGPUR
AUTONOMOUS**

In the Partial Fulfillment of

B.Com. (Computer Application) Final Year

Synopsis Submitted by

Sanket Godhe

Mandar Kawade

Under the Guidance of

Pravin J. Yadao



**G. S. COLLEGE OF COMMERCE & ECONOMICS, NAGPUR
AUTONOMOUS
2021-2022**

1. Introduction:

Mario 2.0 is an upgraded version of the vintage game super mario bros. The game is made for entertainment purposes. It will have several levels in which the main character has to clear several levels which will contain obstacles and reach till the end. The main idea of the game lies in saving the princess from the demon which has captured her.

2. Objectives of the project:

To relive the childhood memories of playing the vintage game.

For getting entertained & Acts a source of relaxation

Having numerous levels will keep excitement high and will increase brain activity.

Easy controls and freely available.

Can be played on all platforms.

1. Project Category: **Game**

2. Tools/ Platform/ Languages to be used: PYTHON

3. Scope of future application:

Multi players.

Daily challenges.

Weakly tournament.

Season update.

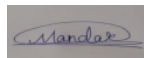
Submitted by,
Mandar Kawade
Sanket Godhe

Approved by,

Prof. Pravin Yadao
Project Guide

Name and Signature of the student

Mandar Kawade



Sanket Godhe

