

Yogev Hassid & Ron Azu

**Mix-It**

# Software Design Document

Ron Azu And Yogev Hassid

Research & Development

Computer Games Development

13/1/2021

# TABLE OF CONTENTS

<b>1. INTRODUCTION</b>	<b>2</b>
1.1 Purpose	2
1.2 Scope	2
1.3 Overview	2
<b>2. SYSTEM OVERVIEW</b>	<b>3</b>
<b>3. SYSTEM ARCHITECTURE</b>	<b>4</b>
3.1 Architectural Design	4
3.2 Decomposition Description	5
3.3 Design Rationale	7
<b>4. DATA DESIGN</b>	<b>7</b>
4.1 Data Description	7
4.2 Data Dictionary	8
<b>5. COMPONENT DESIGN</b>	<b>10</b>
<b>6. HUMAN INTERFACE DESIGN</b>	<b>12</b>
6.1 Overview of User Interface	12
6.2 Screen Images	14
6.3 Screen Objects and Actions	17
<b>7. REQUIREMENTS MATRIX</b>	<b>18</b>

# **1. INTRODUCTION**

## **1.1 Purpose**

The purpose of the document is to describe the architecture and design of the system by analyzing each subsystem into small parts, presenting the objects and components needed for each subsystem by diagrams, and connecting them all together into a complete functioning system.

## **1.2 Scope**

The purpose of project is to develop the capabilities of color combinations. The system's benefits and objectives are: solve a problem of parents trying to expand their children's skills without success and looking for original to do it, fun and unforced ways, children with inability to notice colors and even a financial problem for parents who will not have to buy coloring books and paints. Another goal is a maximum user experience by automatically creating levels and creating a feeling for the user that he is a part of us.

## **1.3 Overview**

This document organizes the entire system formally, individually and addresses each component in the game.

The document details each operation of each process, object and component needed for the system so that diagrams are described that simplify all the connections between all the required processes, entities and objects.

Each diagram is divided into subsystems so that each has an in-depth reference to all the objects and functions of that subsystem.

## **2. SYSTEM OVERVIEW**

Our project is a coloring game that tests the user's color combination skills by matching the colors he sees between a painted painting and a white painting that the user needs to paint with 3 basic colors.

At the beginning of the game, the main menu will be displayed, the purpose of which is to navigate between different processes such as: new game, results table, automatic level creation and exit.

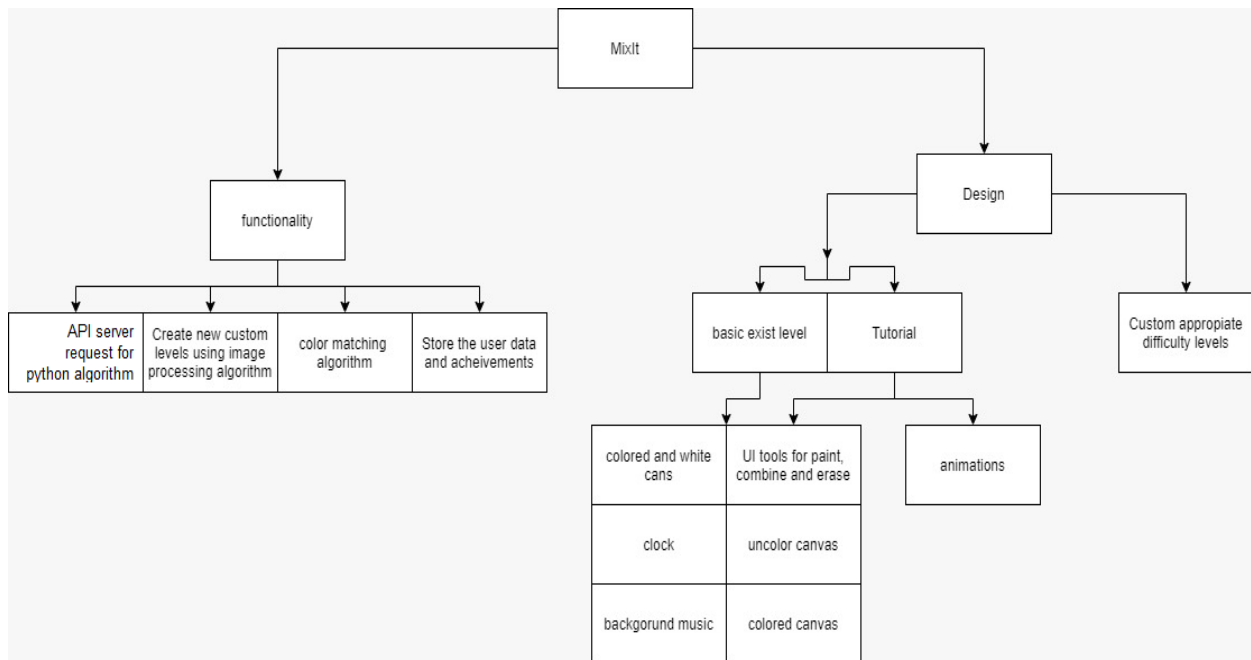
At the beginning of each stage, 2 canvases will be displayed: white and colored drawing, a painting stand that includes: eraser, paintbrush, 3 basic paints and empty cans.

The system will allow the user to create one level automatically so that he uploads an image, the system will process the image by an image processing algorithm and return a white image divided into areas that he will color as he wishes.

In addition, the system will analyze the user's skills and adjust the custom appropriate difficulty level.

### 3. SYSTEM ARCHITECTURE

#### 3.1 Architectural Design



Our system has several sub-processes that stand on their own individually and connect together into a complete system: game levels, tutorial, creating auto custom level by the user, adjusting different difficulty levels and developing the player's color combination abilities.

**Levels of the game** is the main and most important process in the system, which helps the player develops the ability of his color combinations.

After painting, a color matching algorithm is activated that calculates the level of accuracy in the player's painting.

In addition by the process operation, the system analyzes the player's skill level, determines his level of difficulty and stores the player's achievements in a database.

Another important process is the **simulation game** that instructs the user how to play at each level, and in general the operation of the game.

Another important process is to **create a level automatically**, this process allows the player to develop his creativity.

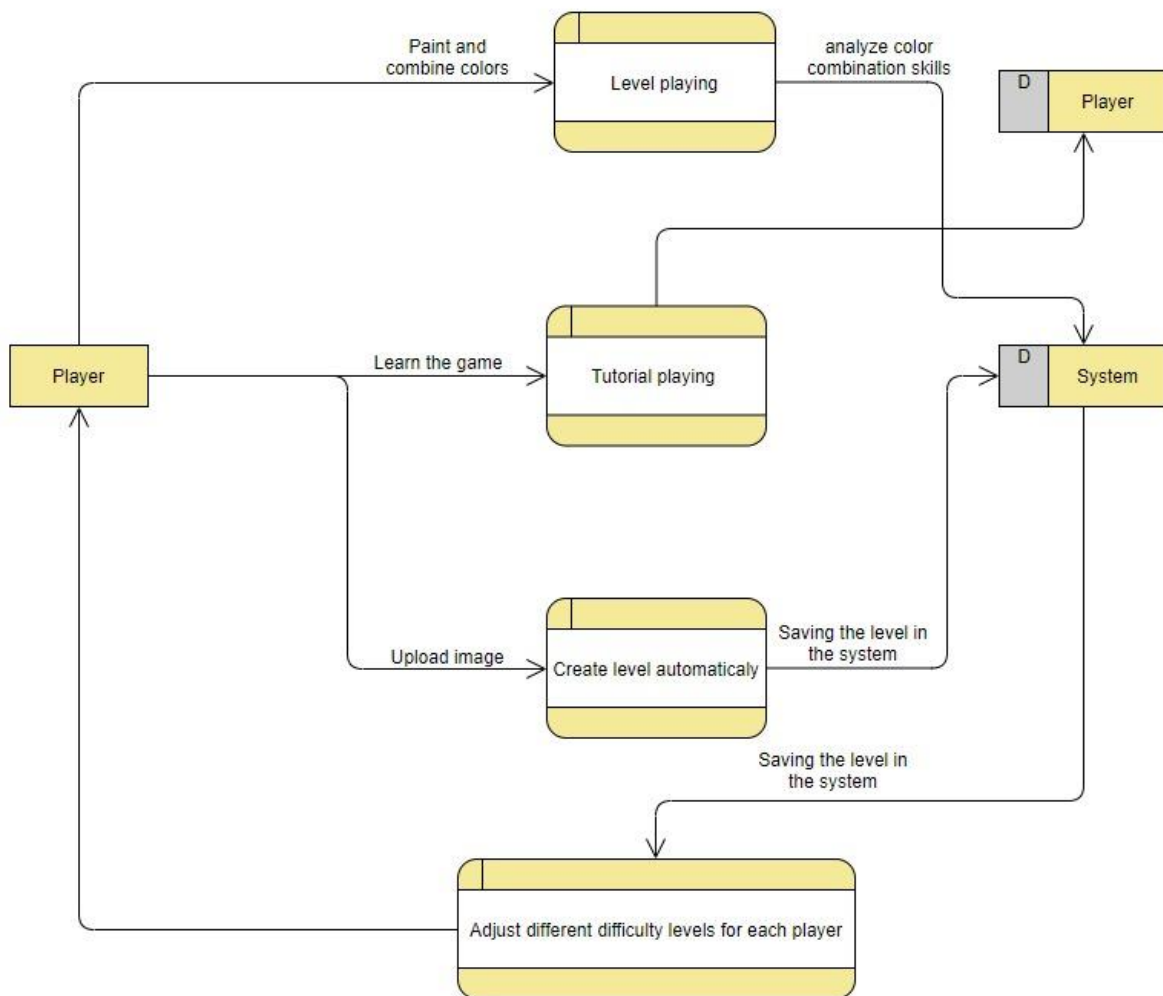
This process is performed by taking the image that the player uploads and running an image processing algorithm which is also a sub-process by itself.

*In conclusion*, there are design processes of the game - the game stages, the simulation game and the difficulty level adjustment are combined with the development of the player's color combination skills, and in addition, the functional processes of creating the automatic stages, image processing algorithms and data

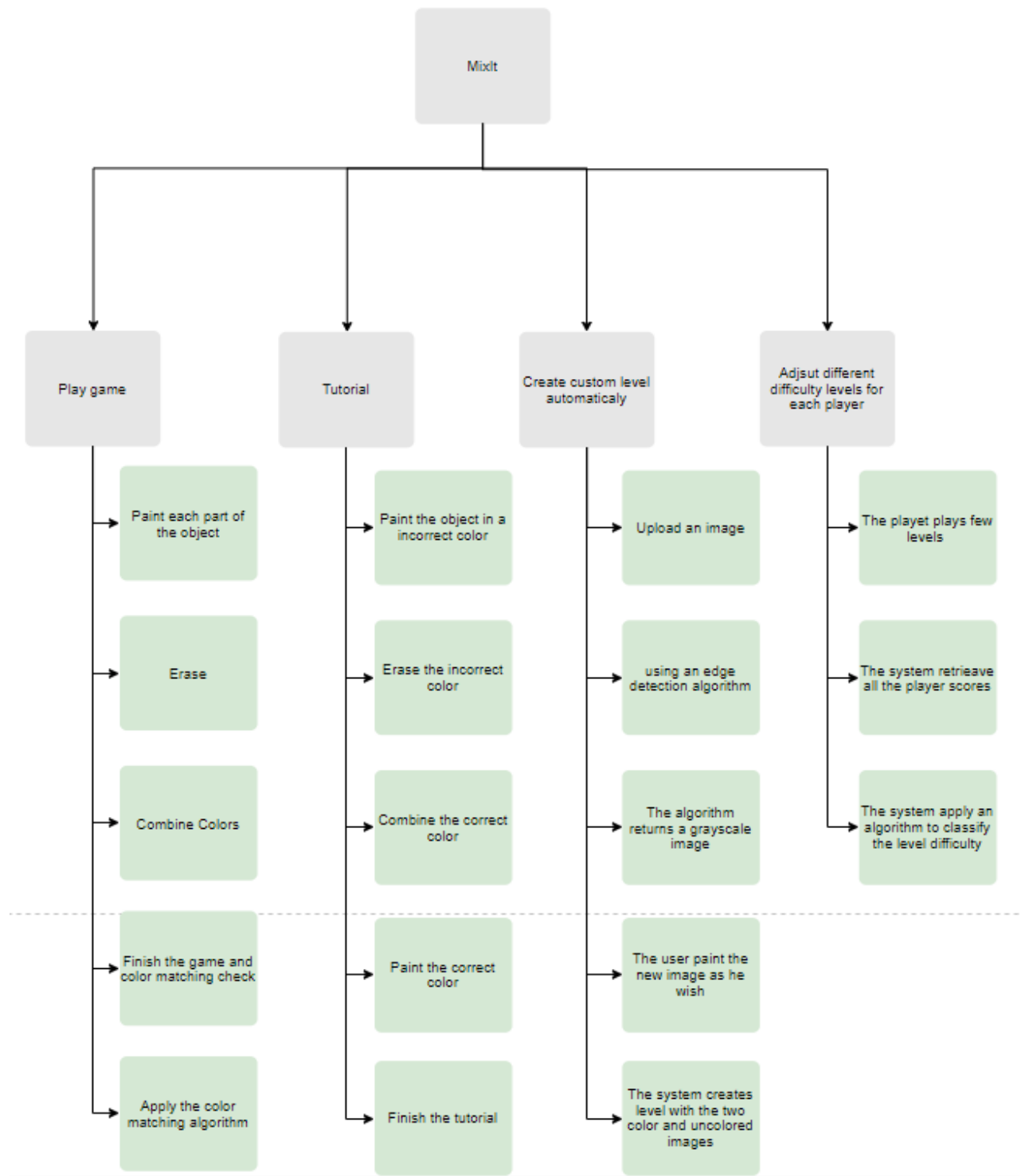
storage contribute to the player's independence and provide a challenge to each player in particular.

### 3.2 Decomposition Description

#### DFD



#### Structural decomposition diagram



### 3.3 Design Rationale

We have chosen this architecture to provide the user maximum convenience and independence, both in terms of developing his color combination skills and in creating the levels automatically.

In our opinion, the user will enjoy the experience of creating the levels, he will feel part of the building and development of the game, he will be part of us.

In addition, using the difficulty level adjustment process, we believe we will be able to retain a lot of users who will not be aware that we have adjusted the difficulty level for them so that they do not get bored quickly of the game or alternatively not give too high a difficulty level to new players.

We chose to do a simulation game to give a "soft landing" to new players who are unfamiliar with the game and its operation.

## **4. DATA DESIGN**

### **4.1 Data Description**

We store in the database all the player's data at each level, the player's personal details and also achievements.

The main entity that will be stored is the player, this entity will be characterized by a number of fields: personal details, level of classification, list of achievements, last level passed, etc.

In creating a step automatically, the system will store the image uploaded by the player in the database for algorithmic calculations and retrieval of the newly created level.

Using these fields, the system will analyze the player's achievement data and thus classify it to an appropriate level of difficulty.

The database will be organized into tables: a level table, a players table, a difficulty level table, an achievement table, an automatic level table.

Each entry to the game, the system will retrieve the player's current information from the database, and display the game according to its results and achievements.

We will store a player type item, a classification level item, an achievement type item, a last level item, etc.

### **4.2 Data Dictionary**



We used a functional description in 3.2 and in this diagram we have 4 main processes:

### **1. Play game:**

- Clicking on one of the 3 base colors, or alternatively after combining the colors, pressing the color button will change the mouse icon to the brush, change the color to the color the player chose, and allow it to paint inside the object. The parameters are the collider of the color we want to choose, the player's current color, and the appropriate area for coloring.
- Clicking on an object and absorbing the click by colliders and placing a correct color on the area that the player clicked, where the parameters will be the collider of the object on which it is clicked and the corresponding color obtained from the color combination by the player.
- Pressing the erase button will change the mouse icon to the eraser, change the current color to white, and allow the player to erase the color from a specific area of the object so that the parameters are the collider of the area to be deleted.
- The color combination button will move the mouse icon to a pointer and will open to the player 3 sliders below the base colors, which the player will be able to slide and combine the color he wants. The parameters are the intensity of the 3 base colors that the player has combined.

### **2. Tutorial:**

- A script with several functions that convey in an organized order the flow of game operation in an animated manner and allow the player to understand the game in the best way. Function parameters are the list of animations.
- The rest of the functions in the same way as the functions of "Play Game" process.

### **3. Create custom level automatically:**

- When you click on the Upload button, the player will be able to upload an image that will be inserted to the system and the database, and the parameters are the image that the player uploaded, and the name of the player.
- Perform an object recognition algorithm and find all edges on the image uploaded by the player, identify all areas of the object and create a new unpainted image. The parameters are just the image.
- Selecting colors using the 3 base colors and creating new colors by combining them. By clicking on colliders the player will paint the resulting drawing as he wishes. The parameters are the intensity of the 3 base colors and the appropriate color for coloring.
- After all the above steps, the system will save the 2 created images: the unpainted image, and the image that the player painted. The system will automatically create a level by placing the unpainted image on the blank canvas and the painted image on the other side, collecting all the scripts in each scene of a step, and create the scene entirely. The parameters are the 2 images.

#### **4. Adjust different difficulty levels:**

- After each level, an matching level algorithm will be automatically executed that pulls all the player's achievements from the database, calculates the total errors in terms of the colors he combined, in terms of time in each level and in terms of the last level reached and finally classifies the player by difficulty level. The parameters are all player achievements including: player score at each stage, time he performed the last level and level number, and player name.
- When the player enters the game, the system will enter the player's result into the database, and then retrieve all its results from the database and run the algorithm that calculates its classification level.
- When entering the main menu of the game, the player will be able to play in levels according to the classification of the levels we compute, and in order to do this we use the player's classification level parameter which is extracted from the database.

## 5. COMPONENT DESIGN

### 1. Play Game:

- Paint:
  1. take the brush after the user clicked on the brush button
  2. take the current color the player selected
  3. take the appropriate area the player clicked on
  4. paint this area with the current color
- Erase:
  1. take the eraser after the user clicked on the eraser button
  2. change the current color to white
  3. take the appropriate area the player clicked on
  4. erase this area by painting it with the current color
- Combine:
  1. display the sliders after the user clicked on the combine button
  2. retrieve the sliders values as CMY color channels and create the new color
  3. if the player click on the 'uncolor' cans:  
Change the color of the can to the new color

### 2. Tutorial:

- Animation Activate:
  1. initial the first animation and display it to the user
  2. while: there are at least more animation to active  
if the user complete this animation's instruction:  
Play the next animation
  3. end the tutorial

### 3. Create custom level automatically:

- Upload image:
  1. get the image from the user as a 3d matrix
  2. upload the image matrix to the database
- Create unpainted image:
  1. apply edge detection algorithm for extract all the edges from the image.
  2. replace all strong edges to black color and the others to white
  3. apply area detection algorithm for extract each area from the black & white image
  4. save each area separately in the database
- Paint Black-White image:
  1. Get the unpainted image and all its areas from the database
  1. Display the sliders after the user clicked on the combine button
  2. Retrieve the sliders values as CMY color channels and create the new color
  3. If the player clicked on the 'uncolor' cans:  
Change the color of the can to the new color
  4. Save the painted image into the database
- Create a level:
  1. Retrieve the black & white and the painted image from the database
  2. put both images side by side on 2 canvases
  3. define threshold and time for this level
  4. create new scene with all the correct scripts

4. Adjust different difficulty levels:

- Classify player difficulty level:
  1. Get the user name
  2. Retrieve from the database all the user achievements
  3. initial sum = 0
  4. for each achievement:
    1. Compute the achievement weight
    2. Add this variable to sum
  5. classify the player difficulty level using sum variable
  6. insert the player difficulty level to the database
  
- Display player difficulty level:
  1. get the player difficulty level from the database
  2. display the correct levels according this difficulty level

## **6. HUMAN INTERFACE DESIGN**

### **6.1 Overview of User Interface**

Describe the functionality of the system from the user's perspective. Explain how the user will be able to use your system to complete all the expected features and the feedback information that will be displayed for the user.

When the user enters the game, he will immediately be shown the main menu screen with: new game, tutorial, automatic level creation, results table and exit.

In addition, the side of this screen will have a volume icon, and appropriate music will play in the background.

New game - This screen will display all the levels that the player has gone through and can play, and locked levels that he has not gone through.

The player can choose any level he has managed to pass and repeat the level , and can choose to continue to the next level and advance in the game.

Once the player starts a certain level, he will be presented with 2 canvases with the painted canvas to the right and the unpainted canvas to the left that needs to be painted.

At the bottom of the unpainted canvas will be a painting surface that includes: paint cans of the 3 base colors, and empty cans to the right that he will need to combine colors.

On the right side of the screen there will be buttons for erase, coloring and color combinations and finishing the level.

On the top left, the time limit for this level will appear.

At the bottom of the screen on the left there will be a home button so that the player can return to the main menu.

Each action of the player will play a suitable sound, and at each level background music will be played.

Once the player has finished painting and presses the finished button, he will receive feedback of whether he has moved on to the next stage or not reached the desired level of accuracy and will be able to start the level again.

Tutorial - This screen will be displayed almost as in a normal level, except for the player's free action.

The system will navigate the player through animations and help him paint the canvas perfectly step by step, in order to teach the player the operation of the game in the best way.

At any step that certain buttons will not affect the simulation operation, they will be turned off and the player will not be able to use them.

Automatic level creation - in this screen the user will press the upload button to upload an image, and after a few seconds of algorithm calculation - the player will be presented with the new image in black and white and will be shown the 3 base colors with which he can paint the unpainted image.

Once he has finished painting, he will be able to click on the finished button and thus complete the step creation process automatically.

Results table - In this screen the user will be able to choose which table will be displayed - a personal results table or a general results table of all users.

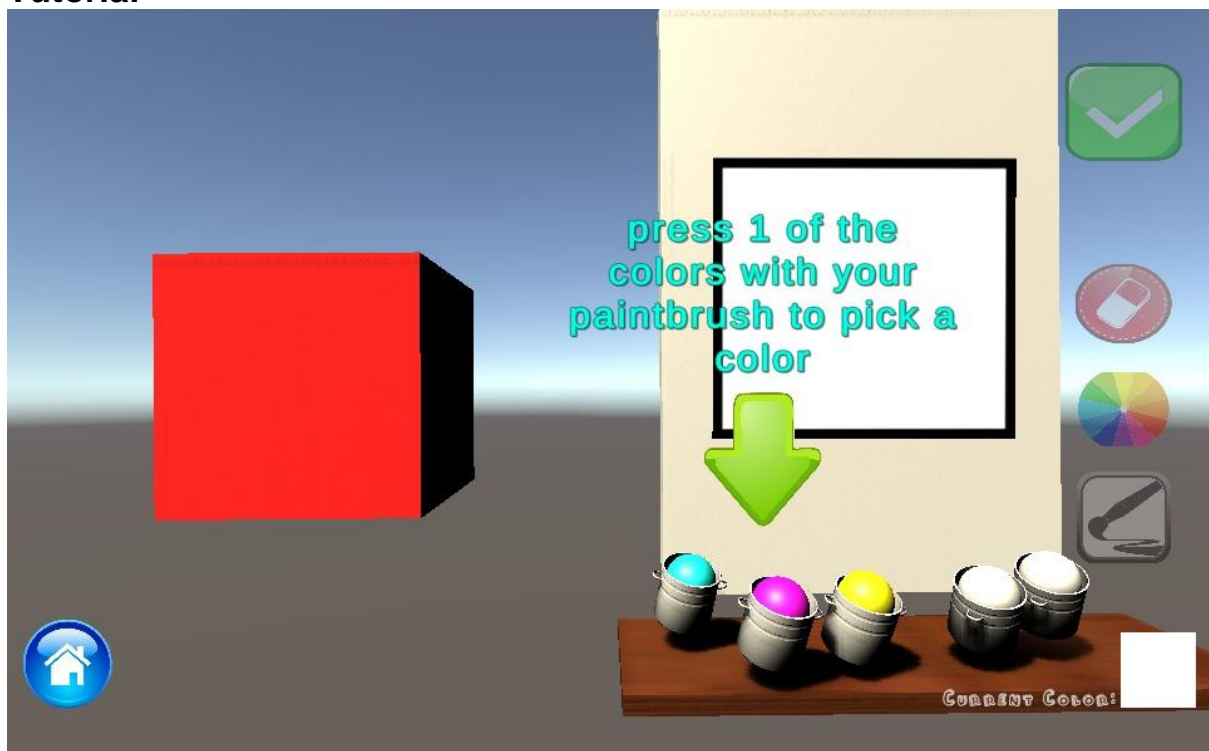
In the personal results table the user will see his highest results from all the levels he passed, and in the general results table the user will see the highest levels and the highest results that all users have reached.

## **6.2 Screen Images**

## Menu



## Tutorial



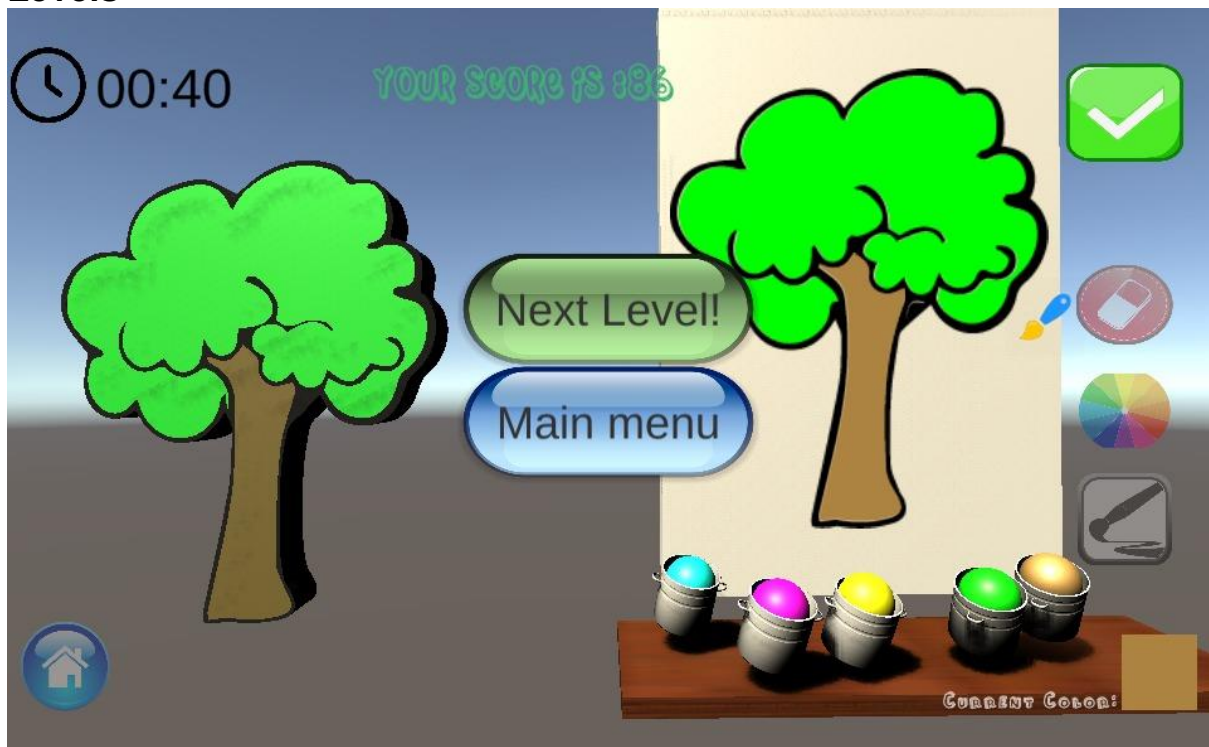
## Before Level

### Levels:

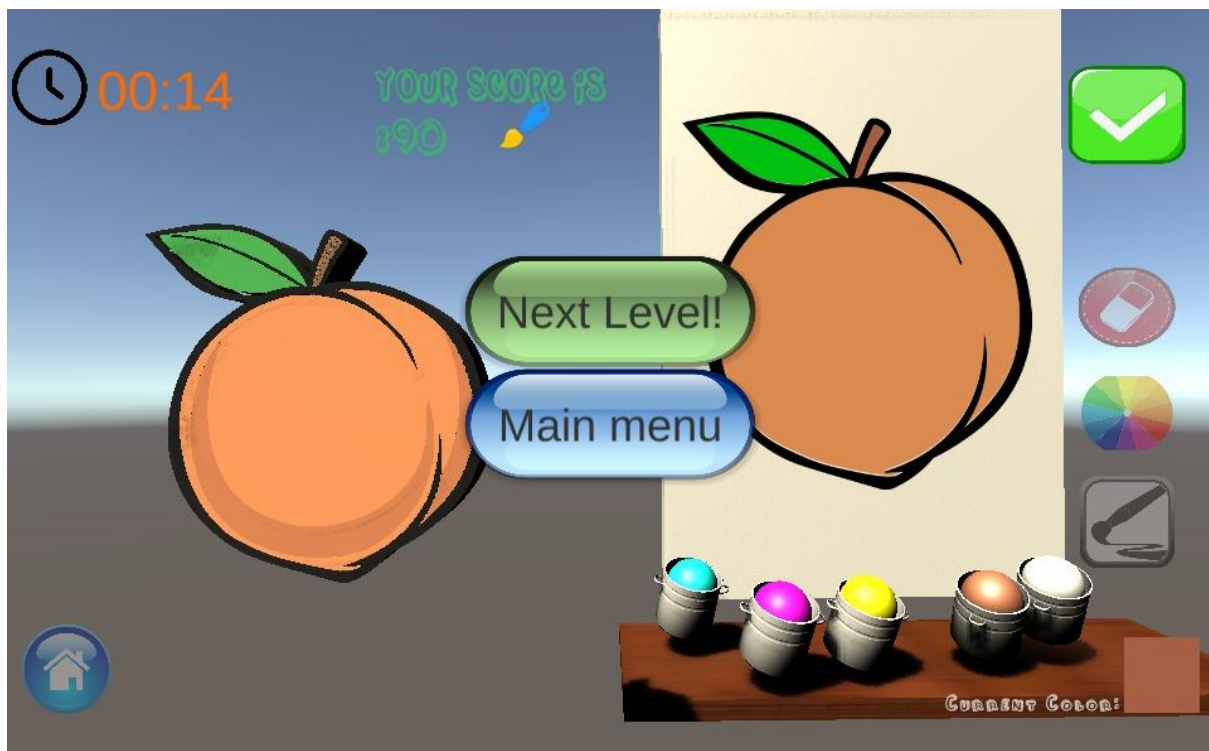
Level 1 88	Level 2 92	Level 3 85	Level 4 83	Level 5 82
Level 6 86				
Level 11	Level 12	Level 13	Level 14	Level 15



### Levels







## Automatic level creation

Create level:

step 1: upload an image of any object



This is how it looks in gray scale:




Paint the image

## Score table

My score level

World score level

NAME	LEVEL	SCORE	TIME
RON	9	23%	4:25
DANA	5	60%	3:10
OMRI	3	55%	2:55
aviv	2	59%	2:10
nathan	2	40%	1:57
YOGEV	1	57%	1:47



## 6.3 Screen Objects and Actions

The new game screen will have objects such as cans, images , brush, eraser and clock.

The user will be able to select a color's can and paint with the brush the unpainted picture.

If the user is not satisfied with the color created, he can take the eraser and erase the color.

In addition, there will be a clock that will be displayed to the player that will show the player how much time he has left for finish.

The tutorial screen will have the same objects as the new game screen.

The automatic level creation screen will have an image upload button that will use the user to upload an image-type object and then run an algorithm to create the image into black and white and display it as an object.

The user will then have the option to click a button that will move it to the object's color screen, just without time and without a painted image on the left.

The results table screen will have a table-type object with strings of the players' highest achievements.

## 7. REQUIREMENTS MATRIX

Requirement	Type	Components
The system will not exceed 1 GB of storage	Functionality	All 2D Scenes (instead of 3D that needs more memory)
The system will store all the player data at each level in a database and will allow access to this data at a constant runtime	Functionality	Player Name as a string is unique (primary key)
All screens will be clear to read and operate	Usability	All objects in each scene (UI unity components)
Every 3 minutes the system will save the player's data	Reliability	Data Base and achievement object
The matching check that will determine the drawing score will be performed in between 3 and 6 seconds	Performance	Color matching algorithm
Each stage of the game will be loaded with a speed of up to 3 seconds	Performance	All 2D Scenes
It will be possible to use the system online on the Internet with Web-GL	Supportability	Unity Builder