Language: C#  
Software Used: Unity Engine

|  |  |
| --- | --- |
| Application Link:  <https://github.com/johnrefani/OPrism-and-EPrism/releases/download/Release/OPrism.and.EPrism.apk> | QR Code: |

Development Process:

1. Analysis

The development process begins with gathering and documenting the application requirements, including features needed. This phase involves thoroughly understanding the functional and non-functional requirements to ensure clarity and direction. The outcome is a comprehensive requirement document that guides the entire development process.

2. Design

In this phase, the application’s concept and design are developed. The design focuses on the overall concept and visual appearance, ensuring the direction is aligned with the project’s goals. This phase lays the foundation for development by establishing the visual and functional blueprint of the application.

3. Development

During execution, the application is developed according to the design specifications. This involves coding the frontend and backend, and integrating the components to create a cohesive application. The development team ensures that all features are implemented correctly, and the application functions as intended.

4. Implementation

The application undergoes various testing phases, including unit, integration, and system testing, to identify and fix bugs. Necessary revisions are made to ensure the application meets quality standards and performs optimally across all platforms. The result is a thoroughly tested application that is ready for deployment, ensuring a smooth user experience.

5. Evaluation

In this phase, areas requiring improvement are examined based on the outcomes of the completed sprint, focusing on both development progress and quality assurance. This enables adjustments to be made to the application, ensuring improvement and optimization.

Source Code:

using UnityEngine;

using UnityEngine.UI;

using UnityEngine.Video;

using TMPro;

using UnityEngine.SceneManagement;

public class JeopardyGame: MonoBehaviour

{

public GameObject mainPanel, instructionPanel, contextPanel, questionPanel, hintPanel, solutionPanel, PopUp;

public int scoretoReach;

public TMP\_Text questionText, answerText1, answerText2, answerText3, hintText, hintCredits, solutionText, solutionCredits, trackedScoreText, scoreText;

public Image questionImage, answerImage1, answerImage2, answerImage3, hintImage, solutionImage;

public Button answerButton1, answerButton2, answerButton3;

public VideoPlayer videoPlayer, solutionVideoPlayer;

public QuestionData questionDatabase;

private int hintsShown = 0, score;

private string currentQuestionKey;

private Color disabledColor = new Color32(226, 219, 255, 255);

private Color disabledAnswerColor = new Color32(172, 10, 13, 255);

private Vector3 originalSolutionImagePosition, originalSolutionRawImagePosition;

void Awake()

{

LoadScore();

string sceneName = SceneManager.GetActiveScene().name;

if (PlayerPrefs.GetInt("Accessed\_" + sceneName) == 0) ShowContext(); else CloseInstruction();

}

void Start()

{

InitializeButtonStates();

string sceneName = SceneManager.GetActiveScene().name;

PlayerPrefs.SetInt("Accessed\_" + sceneName, 1);

PlayerPrefs.Save();

}

public void OnQuestionButtonClicked(string questionKey)

{

DisplayQuestion(questionKey);

}

public void PlayVideo()

{

videoPlayer.Prepare();

videoPlayer.Play();

}

public void StopVideo()

{

videoPlayer.Stop();

}

public void ShowMainPanel()

{

EnableAnswerButtons();

solutionVideoPlayer.Stop();

ResetSolutionPanelPosition();

}

public void DisplayQuestion(string questionKey)

{

if (!questionDatabase.questions.ContainsKey(questionKey))

{

Debug.LogError("Question Key does not exist");

return;

}

currentQuestionKey = questionKey;

var (question, questionType, answers, answerType, correctIndex, choicesCount, \_, \_, \_, \_, \_, \_, \_, \_, \_, \_, \_) = questionDatabase.questions[questionKey];

DisableQuestionButton(questionKey);

ResetButtonState(answerButton1);

ResetButtonState(answerButton2);

ResetButtonState(answerButton3);

SetQuestionContent(questionType, question);

SetAnswerButtons(choicesCount, answerType, answers);

questionPanel.SetActive(true);

}

private void ResetButtonState(Button button)

{

CanvasGroup canvasGroup = button.GetComponent<CanvasGroup>();

if (canvasGroup != null)

{

canvasGroup.alpha = 1.0f;

canvasGroup.interactable = true;

canvasGroup.blocksRaycasts = true;

}

else

{

Debug.LogWarning("No CanvasGroup found on button: " + button.name);

}

ColorBlock colorBlock = button.colors;

colorBlock.disabledColor = new Color32(255, 255, 255, 255);

button.colors = colorBlock;

}

private void SetAnswerImage(Image imageComponent, string imageName)

{

if (!string.IsNullOrEmpty(imageName))

{

Sprite answerSprite = Resources.Load<Sprite>(imageName);

if (answerSprite != null)

{

imageComponent.sprite = answerSprite;

}

else

{

Debug.LogError("Answer image not found: " + imageName);

}

}