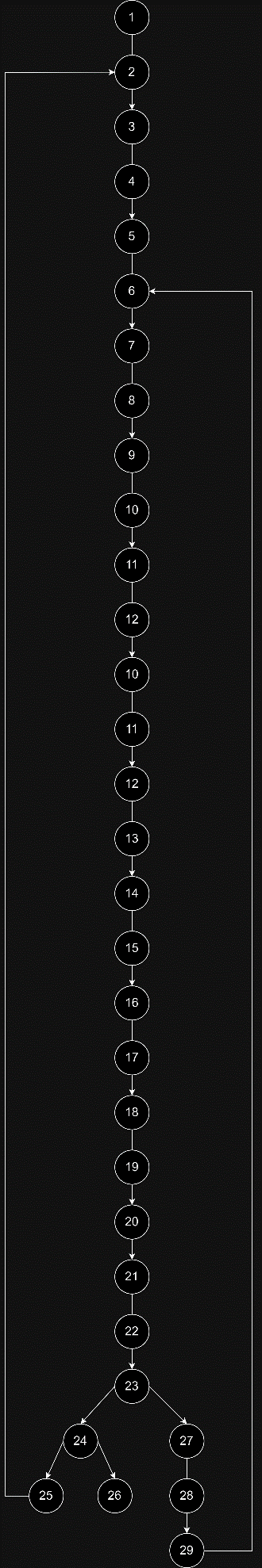
1. White Box

|  |  |  |
| --- | --- | --- |
| **No** | **Code** | |
|  | public class Quiz : MonoBehaviour{ | |
|  | [Header("Questions")] | |
|  | [SerializeField] TextMeshProUGUI questionText; | |
|  | [SerializeField] List<QuestionsSO> questions = new List<QuestionsSO>(); | |
|  | QuestionsSO currentQuestion; | |
|  | [Header("Answers")] | |
|  | [SerializeField] GameObject[] answerButtons; | |
|  | bool hasAnsweredEarly = true; | |
|  | int correctAnswerIndex; | |
|  | [Header("Button Sprite Colors")] | |
|  | [SerializeField] Sprite defaultButtonSprite; | |
|  | [SerializeField] Sprite CorrectButtonSprite; | |
|  | | [Header("Timer")] |
|  | | [SerializeField] Image timerImage ; |
|  | | [Header("Score")] |
|  | | [SerializeField] TextMeshProUGUI displayScoreText; |
|  | | Timer timer; |
|  | | Score score; |
|  | | [Header("Progress Bar")] |
|  | | [SerializeField] Slider progressBar; |
|  | | public bool isComplete; |
|  | | void Update() { |
|  | | timerImage.fillAmount = timer.fillFraction; |
|  | | if(timer.LoadNextQuestion){ |
|  | | if(progressBar.value == progressBar.maxValue){                  isComplete = true;                  return;              } |
|  | | hasAnsweredEarly = false;              nextQuestion();              timer.LoadNextQuestion = false; |
|  | | }else if(!hasAnsweredEarly && !timer.isWaitingForAnswer){ |
|  | | displayCorrectAnswer(-1); |
|  | | setButtonState(false);          }      }  } |

Selanjutnya dilakukan penggambaran dari setiap alur *script* atau disebut dengan Basis *Path Test,* Basis *Path Test* dapat dilihat pada gambar di bawah ini :



##### Gambar 4.26 *Basis Path Test*

Menghitung *Cyclomatic Compliexity* : kompleksitas siklomatis pengukuran kuantutatif dari grafik alir dapat diperoleh dengan perhitungan:

1. V(G) = E-N+2

V(G) = 30-29+2

V(G) = 3

Keterangan:

V(G): Cyclomatic complexity untuk flow graph G

E : Panah (edge)

N : Lingkaran (node)

Dari perhitunan diatas, kompleksitas siklomatis yang dihasilkan adalah 3 dan dari gambar diagram alir dan kompleksitas siklomatis diatas dapat ditentukan independent path. Terdapat 2 independent path yang diperoleh yaitu :

Path 1 : 1,2,3,4,5, 6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25

Path 2 : 1,2,3,4,5, 6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,26

Path 3 : 1,2,3,4,5, 6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,27,28,29

Selanjutnya dilakukan pengujian value test yaitu sebagai pengujian pada setiap independent path.

# Tabel 4.7 *Value Test*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Path | Input | Output | Keterangan |
| 1. | 1,2,3,4,5, 6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25 | Mengambil jawaban benar | Berhasil mengambil jawaban benar dengan menambah skor | Berhasil |
| 2. | 1,2,3,4,5, 6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,26 | Mengambil jawaban salah | Berhasil mengambil jawaban salah | Berhasil |
| 3. | 1,2,3,4,5, 6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,27,28,29 | Mengambil jawaban salah dan waktu habis | Berhasil mengambil jawaban salah dan waktu habis | Berhasil |

Berdasarkan hasil pengujian diperoleh hasil

Tercapai = 3/3 x 100% = 100%

Gagal = 0 x 100% = 0%

Maka kesimpulan dari pengujian diatas adalah pengujian yang dilakukan menggunakan diagram alir tercapai.

1. Kebutuhan

Analisis kebutuhan sistem yang mencakup seperti kebutuhan perangkat keras ( hardware ) dan kebutuhan perangkat lunak ( Software ) yang digunakan dalam menjalankan aplikasi tersebut. Kebutuhan hardware dalam pembuatan aplikasi tersebut adalah sebagai berikut :

1. CPU dengan spesifikasi intel core i5
2. Memori RAM 8GB DDR3
3. Processor 512GB M.2 NVMe™ PCIe® 3.0 SSD

Sedangkan kebutuhan software dalam pembuatan apliaksi tersebut adalah sebagai berikut:

1. Unity 2020
2. Windows 11 enterprise OS
3. Corel draw x7
4. Figma