Dylan Gomez Mark _____/50

1. Brief introduction __/3

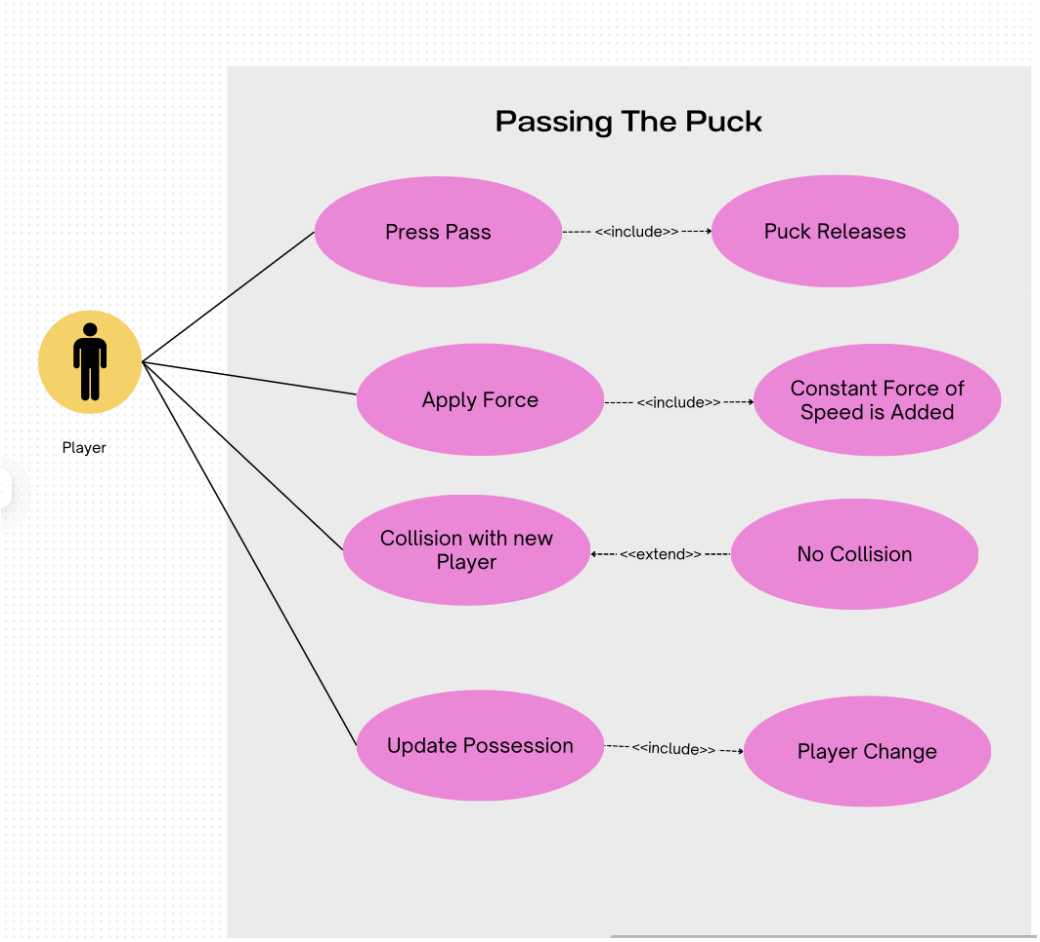
My feature for the Retro Hockey game is going to be focused on two different areas:

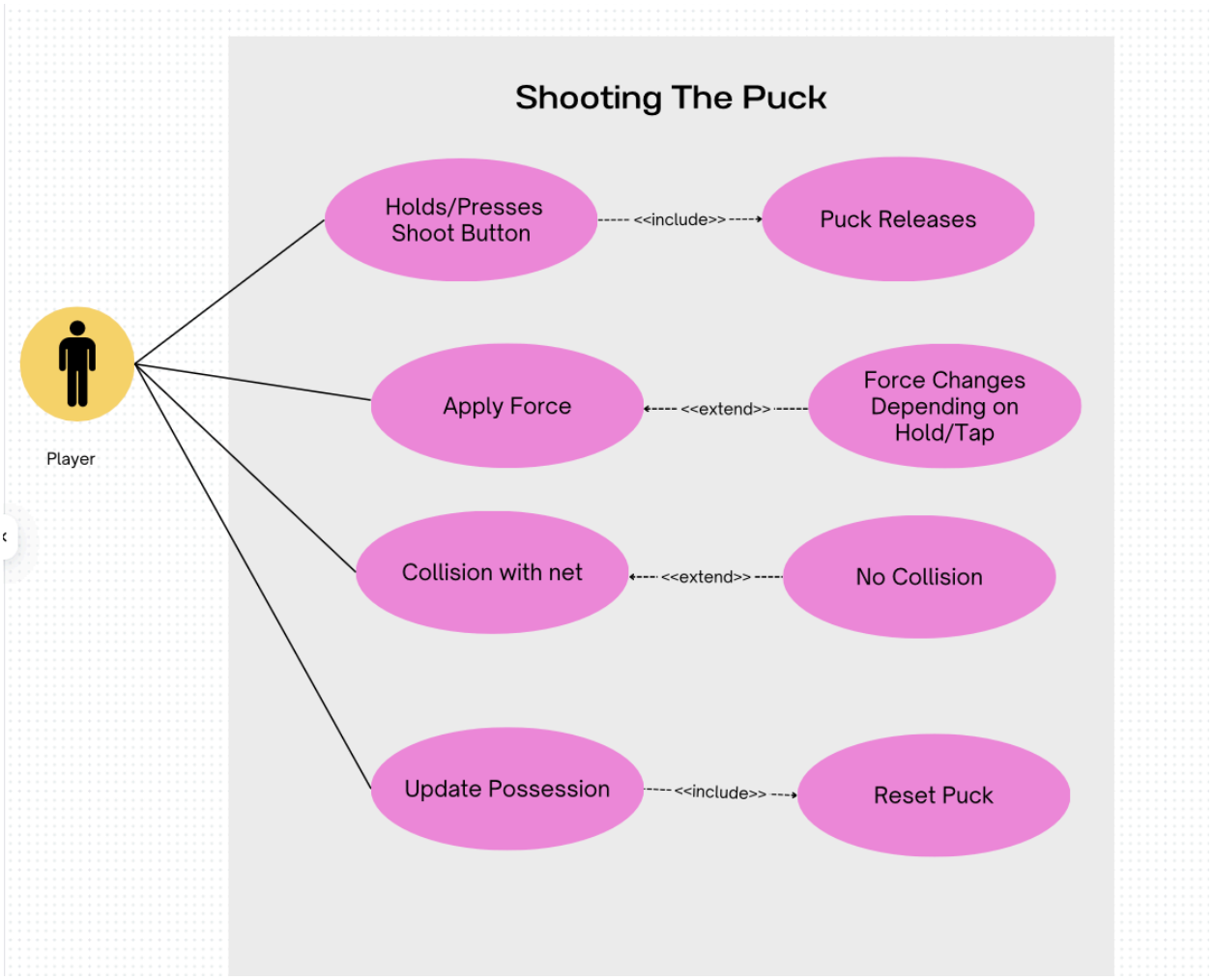
1. First off, I am TL-6, which is the Version Control person. I am going to have to become a bit more advanced in Git so I can ensure smooth collaboration with all of Github's actions. For this I am going to want to maintain a clean history on Git for our team. I will work alongside Matt to read any changes to files and to make sure that no bugs or edge cases will effect our repo. From that point on we will pull branches and test the code.

2. Secondly, I have been tasked with the core mechanics(passing and shooting). I am going to have to collaborate a lot with Robbie since he is developing the AI/opponents for this project. My job is ensuring that the mechanics look clean and work well to make our game complete. I will develop the shooting and passing for this game keeping in mind collisions, opponents, different types of shots, and all the animations that make it seem like a retro bowl-inspired hockey game!

2. Use case diagram with scenario __14

Use Case Diagrams





Scenarios

Scenario 1

Name: Passing Puck
Summary: The player attempts a pass, the system then will select a valid new player to receive the puck, and then the puck updates the new possession.
Actors: Player
Preconditions: Player has possession of the puck
Basic sequence:

Step 1: Player presses the pass input
Step 2: System finds a teammate within the aim cone

Step 3: System applies pass force

Step 4: Collision into opposing team or same team

Step 5: Possession is updated

Exceptions:

4: No Collision

Post conditions: Puck has transferred possession

Priority: 1*

ID: HP1

Scenario 2

Name: Shooting

Summary: The player shoots, system computes the power/direction/collisions with other players, and then updates the score and then the puck is reset

Actors: Player

Preconditions: Player has possession of the puck

Basic sequence:

Step 1: Player presses/holds the shoot button

Step 2: System applies force to the puck at direction player is facing

Step 3: Collisions with goalie/players or the net

Step 4: Possession is updated

Exceptions:

2: Force Changes depending on how long the player holds

3: No collisions

4: Puck Resets in the center

Post conditions: Goal is recorded or play continues

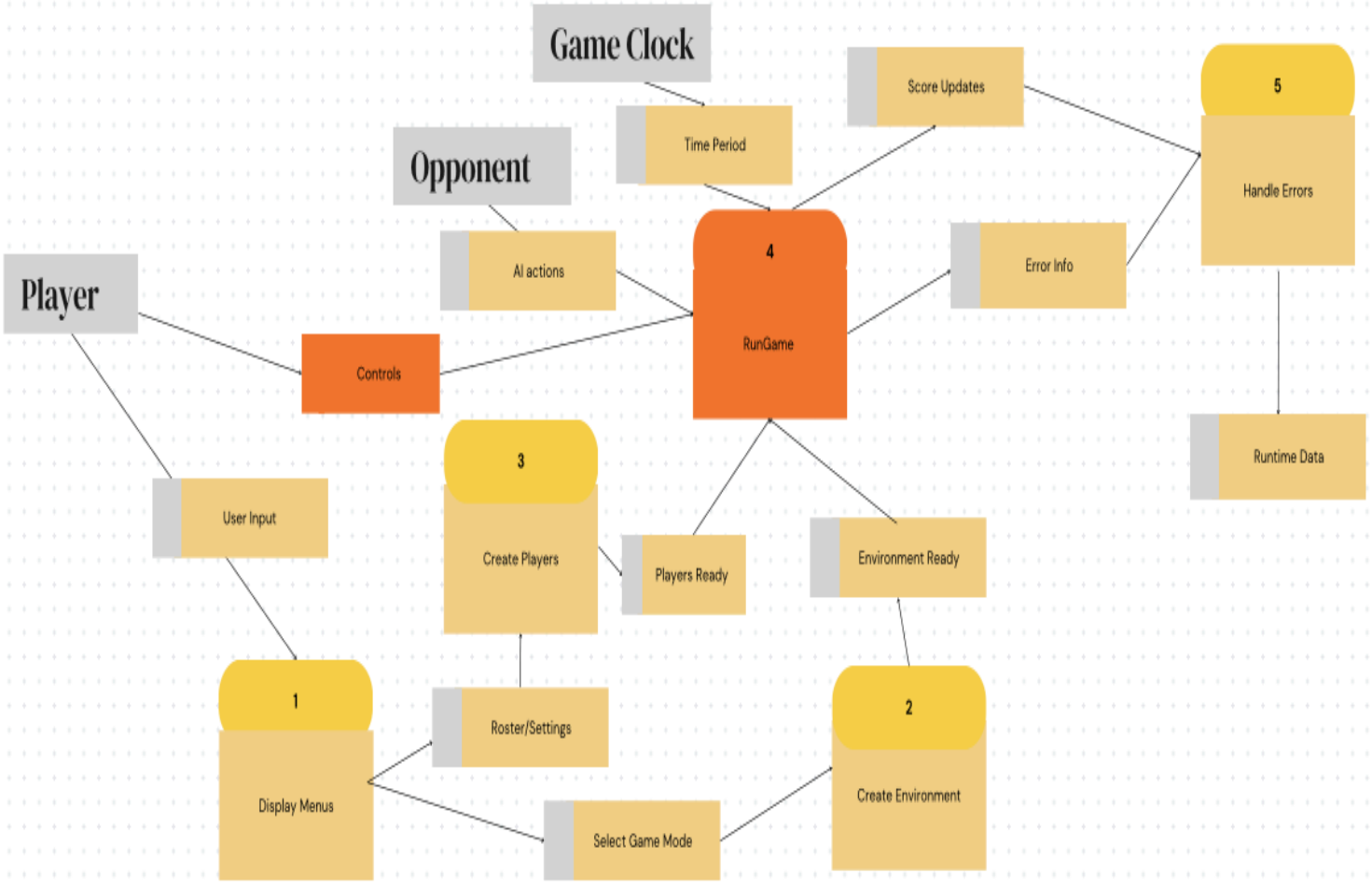
Priority: 1*

ID: HS1

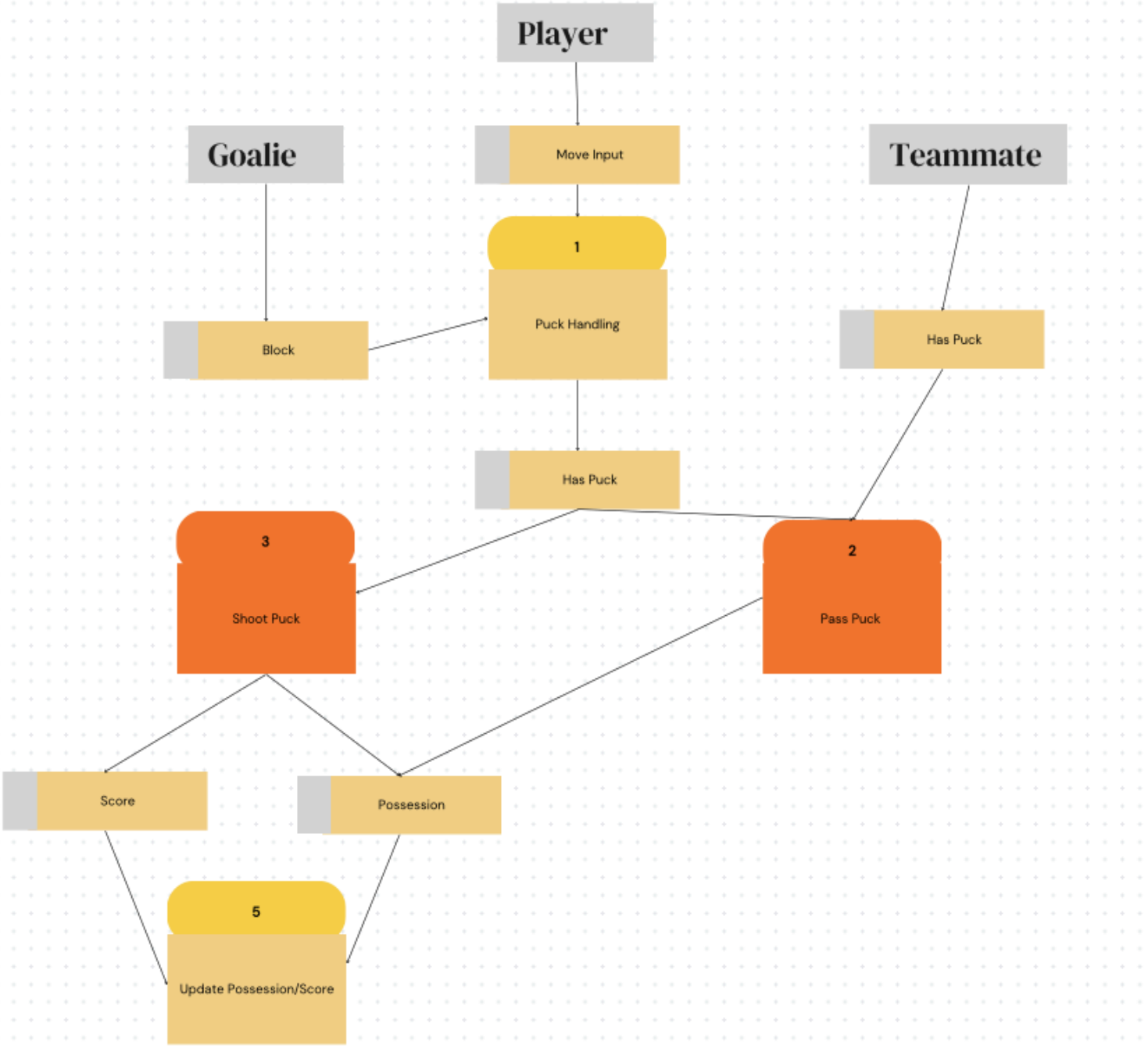
3. Data Flow diagram(s) from Level 0 to process description for your feature _____14

Data Flow Diagrams

Diagram 0


Game Clock
Time Period
Opponent
AI actions
Player
Controls
User Input
1
Display Menus
Roster/Settings
Select Game Mode
3
Create Players
Players Ready
4
RunGame
Score Updates
Error Info
2
Create Environment
Environment Ready
5
Handle Errors
Runtime Data

RunGame


Player
Goalie
Move Input
Teammate
1
Puck Handling
Block
Has Puck
Has Puck
3
Shoot Puck
2
Pass Puck
Score
Possession
5
Update Possession/Score

Process Descriptions

Display Menus:

SHOW Main menu options to player
WAIT for player input
IF player selects “Roster/settings”
LOAD roster screen
STORE roster choices
ELSE IF player selects “Game Mode”
LOAD game mode screen
STORE mode choice
ELSE IF player selects “Exit”
END program
END IF
END

Create Environment:

LOAD environment assets
SET default physics rules
IF game mode = “Stanley Cup”
Apply Stanley Cup settings
ELSE
Apply default environment settings
END IF
Environment is ready
END

Create Players:

FOR each player
IF user-controlled
ASSIGN user controls
ELSE
ASSIGN ai controls
END IF

```
        END FOR
        IF roster/settings provided
                APPLY attributes
        END IF
        Player is ready
END

Run Game:
        While clock > 0
                RECEIVE user input
                RECEIVE opponent actions
                UPDATE puck position and possession
                IF goal scored
                        INCREMENT score
                        RESET puck to center after a delay
                END IF
                IF error occurs
                        SEND error to HANDLE ERRORS
                END IF
        END WHILE
        Show Results
END


Handle Errors:
        IF runtime data is invalid
                SHOW error info
                ATTEMPT recovery
                IF can't recover
                        END game
                END IF
        END IF
END


Pass Puck:
        IF player has puck AND presses pass input
                IDENTIFY teammate in direction of pass
                APPLY pass force to puck
                CHANGE possession
        END IF
```

Shoot Puck:

IF player has puck AND presses shoot input
CALCULATE shot direction
APPLY shot force to the puck
IF goalie blocks
CHANGE possession to goalie
ELSE
CHECK if puck crosses goal line
IF true
INCREMENT score
END IF
END IF
END IF

Update Possession/Score:

IF puck possession changes
UPDATE possession status
END IF
IF goal scored
UPDATE score
RESET puck position to center
END IF

4. Acceptance Tests _____9

Passing: Passing will be accepted when the puck travels toward the nearest teammate within a 60 degree aim cone 8 out of 10 times. Also if a defender is within this 60 degree aim cone the defender must also steal this pass 8 out of 10 times.

Shooting: Shooting will be accepted when quick shots always travel at a lesser speed than power shots(achieved from holding the button). Release time for quick shots must always be under .2 seconds and release time for power shots must always be .5 seconds or more. Goals must also be registered 100% of the time when the puck crosses the line.

Inputs VS Outputs

Inputs	Intended Outputs		

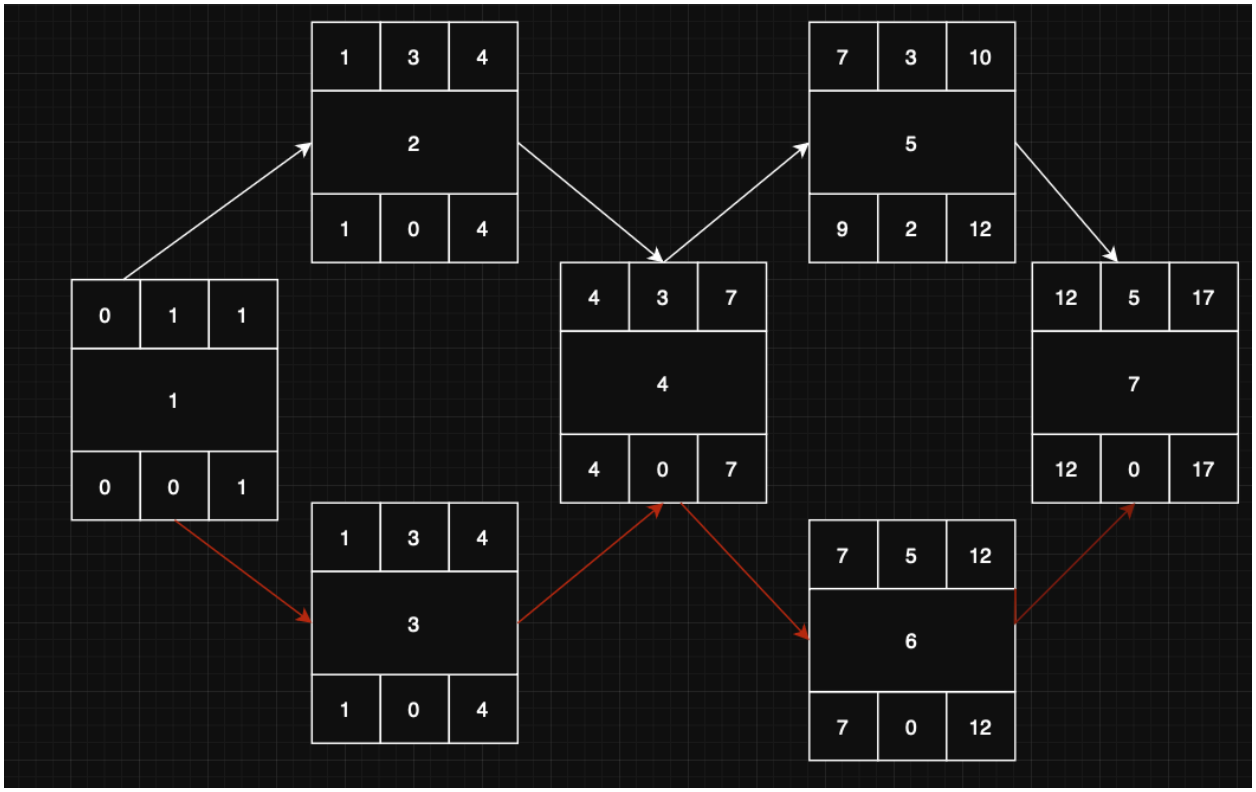
60 degree pass (Same Team)	80% >= pass success		
60 degree pass(Opposing)	80% >= pass failure		
Power vs Quick Shots	Must Differ		
Puck Crosses Line	100% Goal Success		

5. Timeline _____/10

Work items

Task	Duration (PWks)	Predecessor Task(s)
1. Define Input Actions/Triggers	1	-
2. Passing	3	1
3. Shooting	3	1
4. Shared Rules	3	2, 3
5. Documentation	3	4
6. Integration/Testing	5	4
7. Bug Fixing/Polish	5	5,6
8. Version Control Workflows	6	-

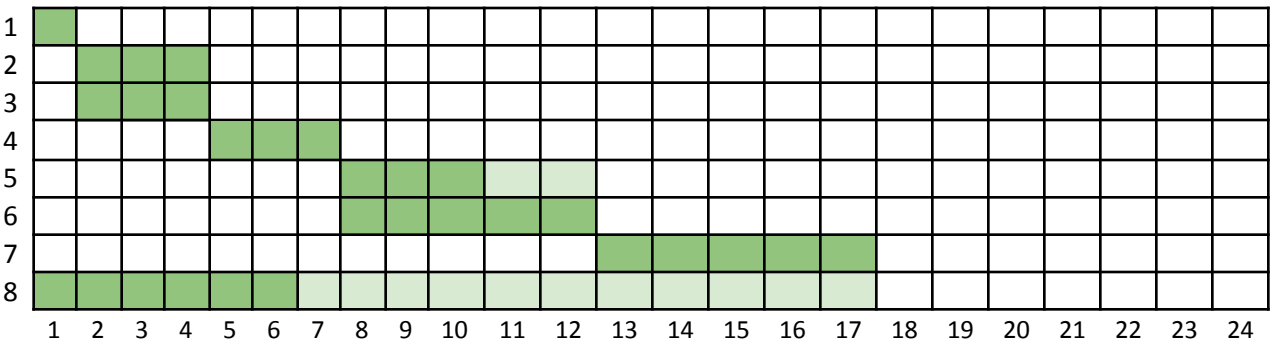
Pert diagram



Key:

	Work Hours
	Slack

Gantt timeline



p