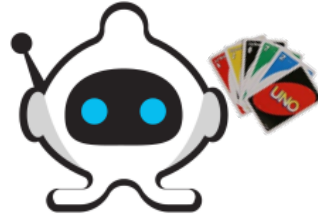


Presented by: Emiliano, Jordan, Ethan, Roberto, Andres,
Aly, Colin

Create-Gineers!



UNOverse



The Idea



- The app will be a desktop-based digital UNO game that allows players to enjoy a classic card game experience against AI bot opponents
- The goal is to recreate the fun and challenge of UNO into a simple, intuitive digital format that's accessible for everyone
- Users Needs & Motivation
 - Quick and accessible fun
 - Easy setup
 - Fair to play through automated rules

Scope & Consumer

- Designed for:
 - Children
 - Young adults
 - Grown adults
 - Families
 - Casual gamers
 - Students
- Scope (included)
 - Implement an UNO game with turns, card effects, and automatic rule enforcement
 - Support vs. Bot gameplay
 - Includes core components: Game manager, Rules Engine, Deck Manager, and UI.
 - Handle turn sequence, card drawing, validity checking, and game over detection.



Activity Diagram

Single Turn Process Overview

Purpose:
Defines the flow for one player's turn in the card game.

1. Player Phase

Check Playable Cards

If none → Request Draw

Add drawn card to hand

If new card playable → proceed

Choose and Play Card

2. GameManager Phase

If Action Card → HandleActionCard()

Possible effects: Skip Turn, Reverse, Draw Two, Choose Color

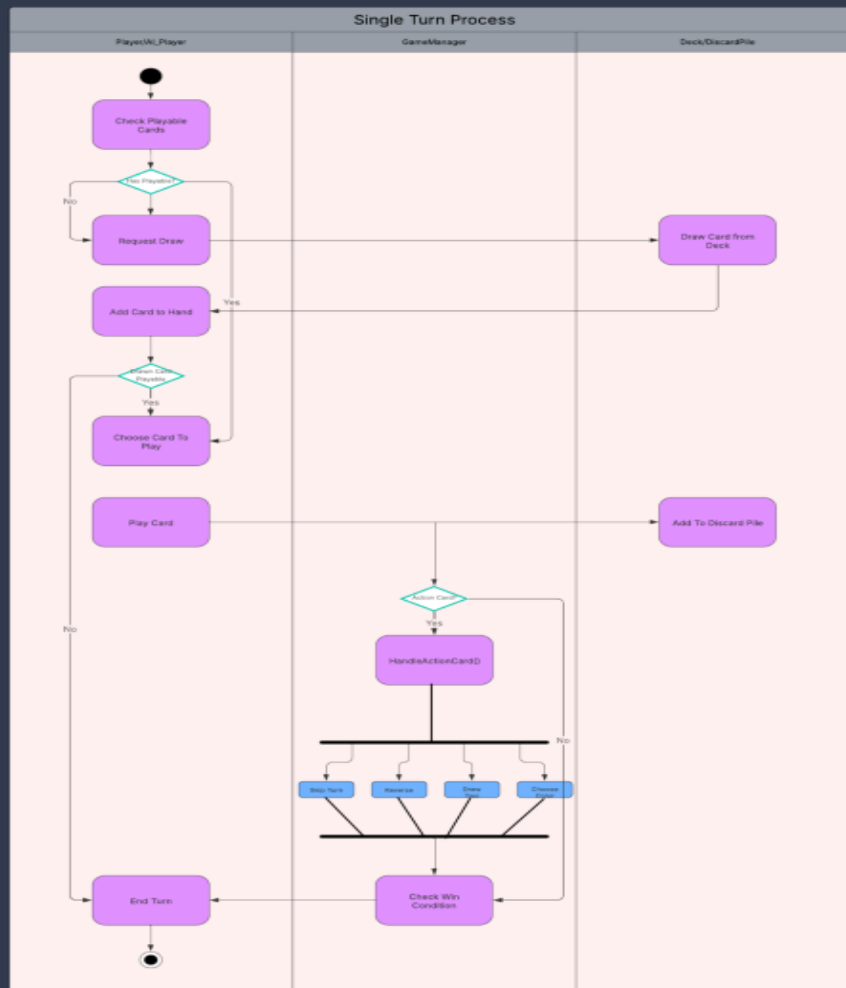
Check Win Condition

3. Deck/DiscardPile Phase

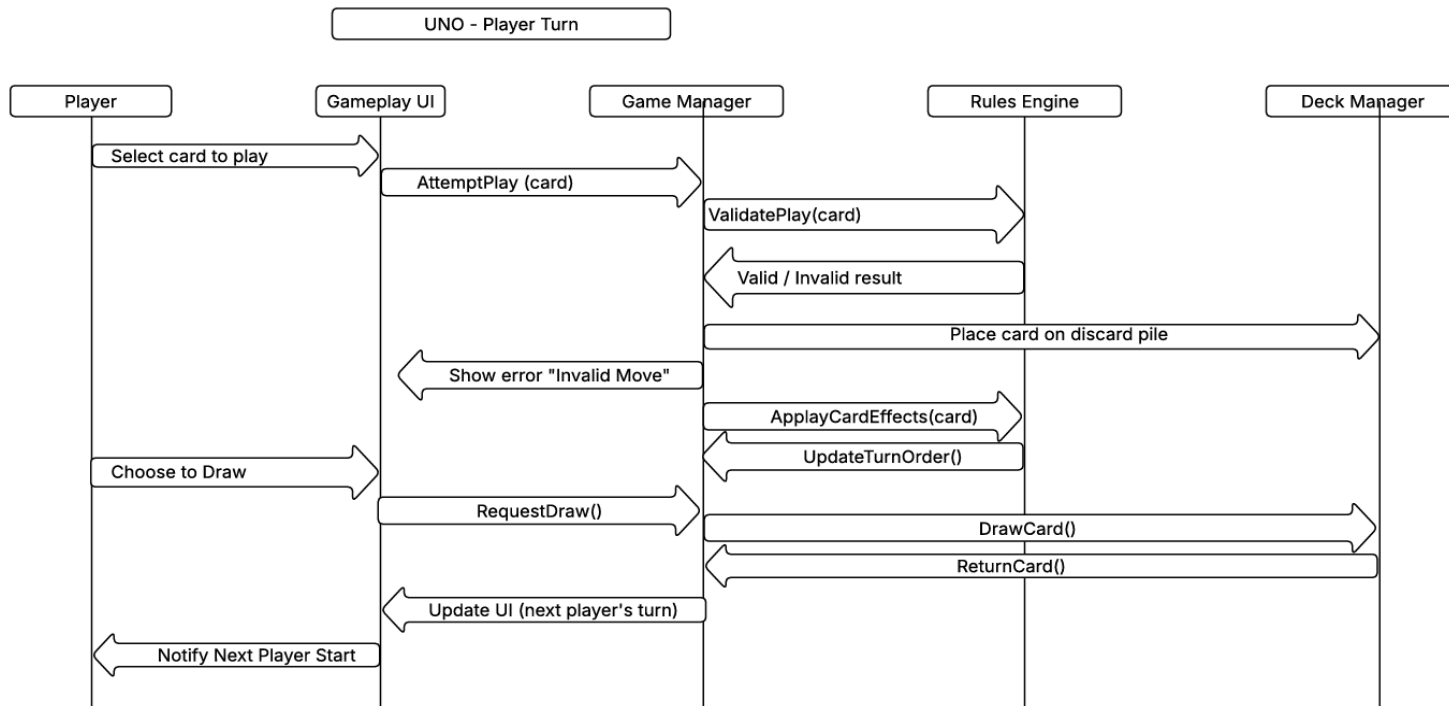
Draw Card from Deck

Add Played Card to Discard Pile

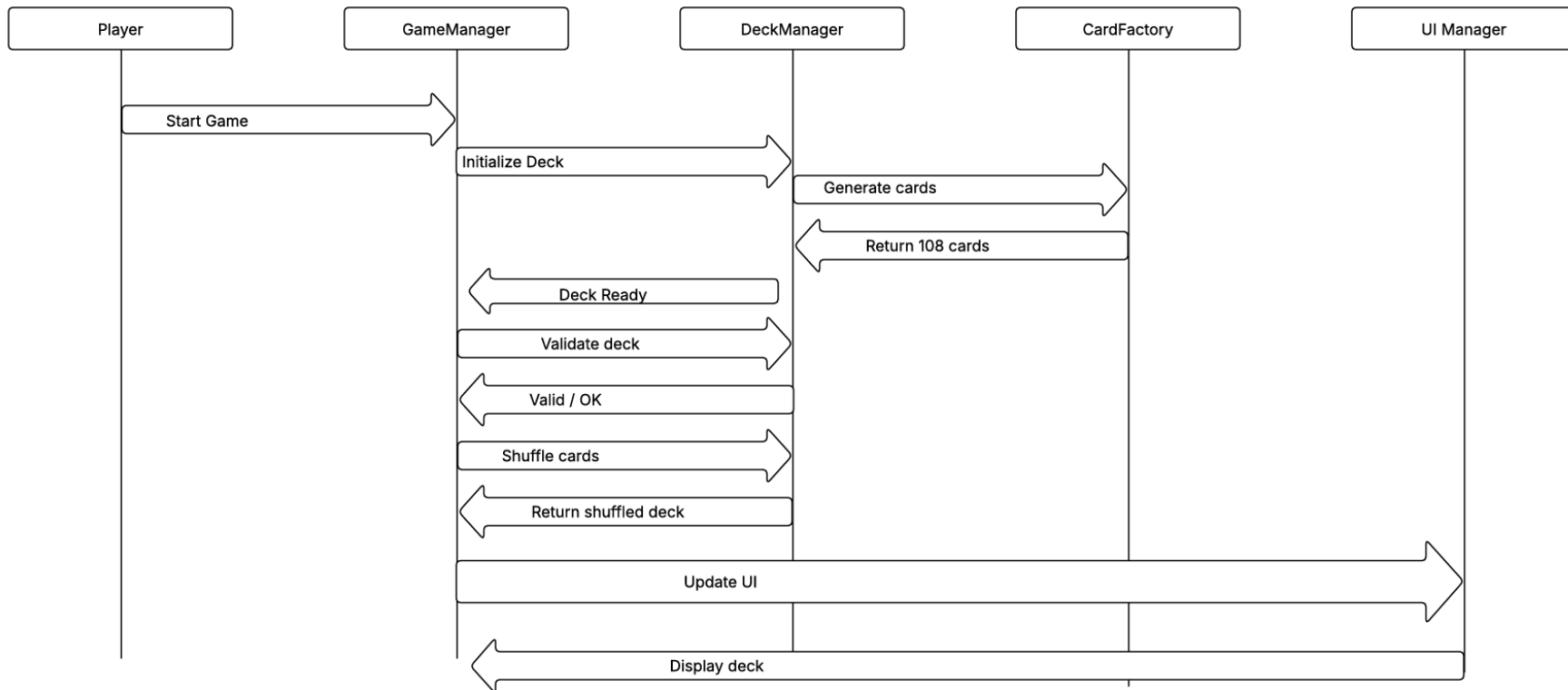
End Turn → Next Player



Sequence Diagram 1: Player Turn



Sequence Diagram 2: Deck Initialization



Use-Case

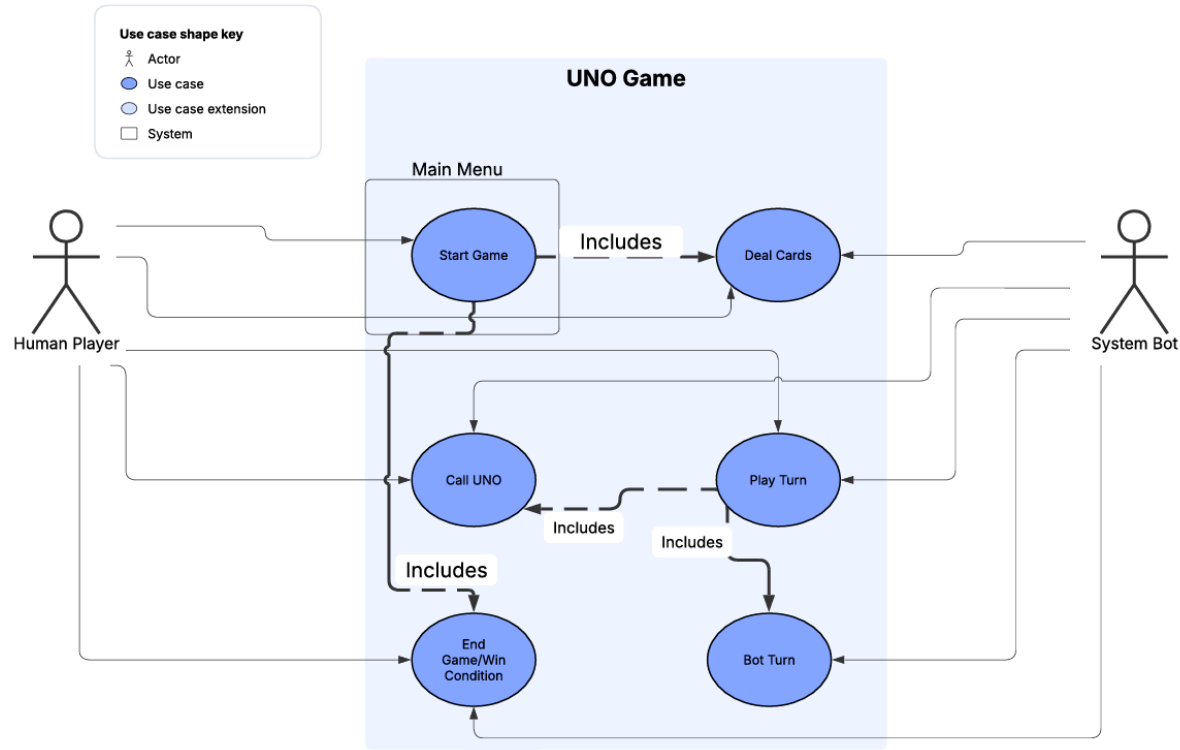
Here we show how different actors interact with the program

Human Player directly interacts with

- Starting the game
- Calling Uno
- Dealing Cards
- Ending/Winning the game
- Taking their turn

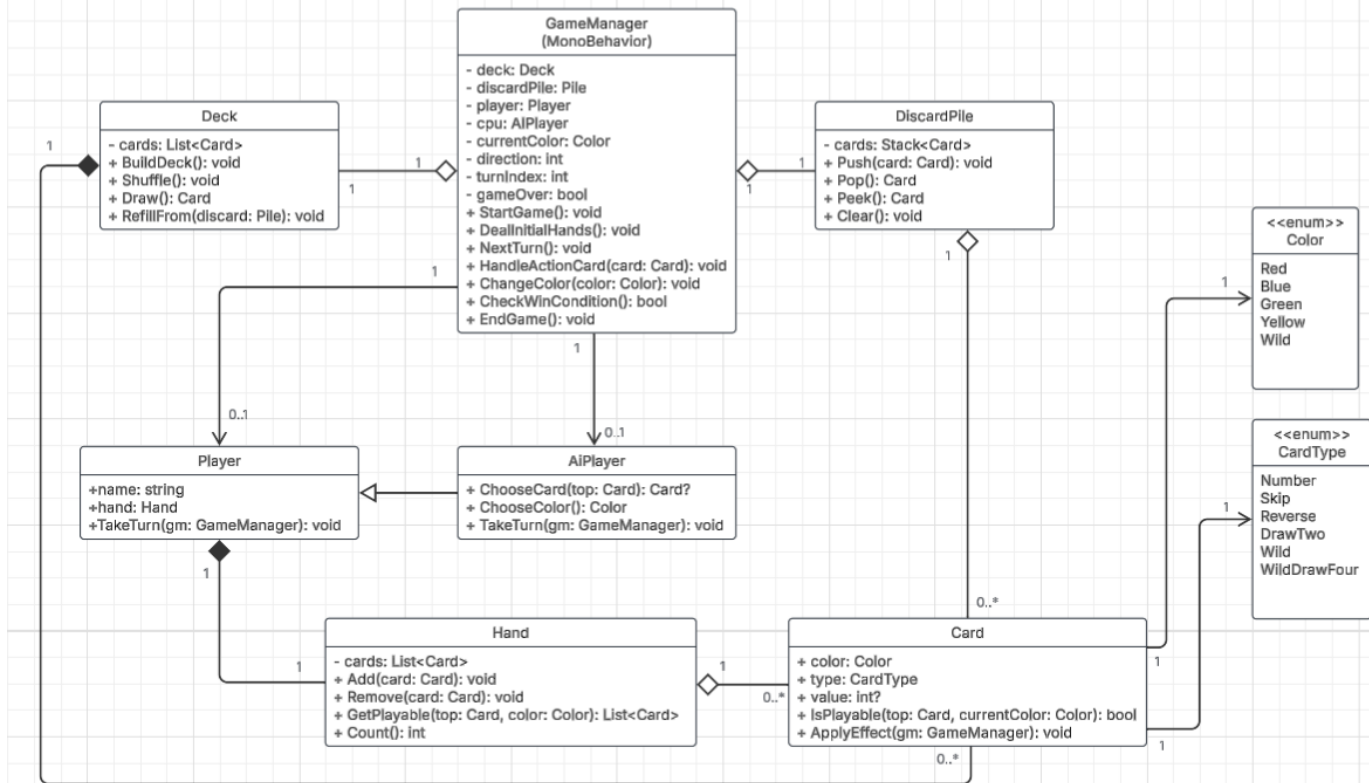
System Bot directly interacts with

- Calling Uno
- Dealing Cards
- Ending/Winning the game
- Taking their turn
- Automatically determines card to play



Class

- Shows the structure of our UNO game and how classes connect
- GameManager controls game flow, turns, and win conditions
- Player (human) and Bot handle card actions and decision.
- Card defines color, number, and type
- Deck manages drawing and shuffling cards
- DiscardPile collects played cards
- Clear relationships = easier coding, testing, and debugging



SCRUM Artifacts

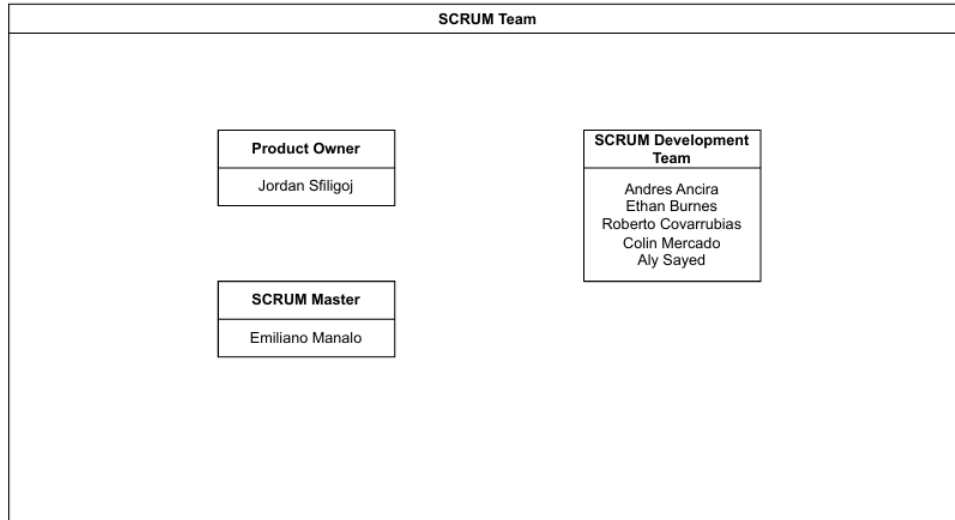


Figure 1: SCRUM Team Formation



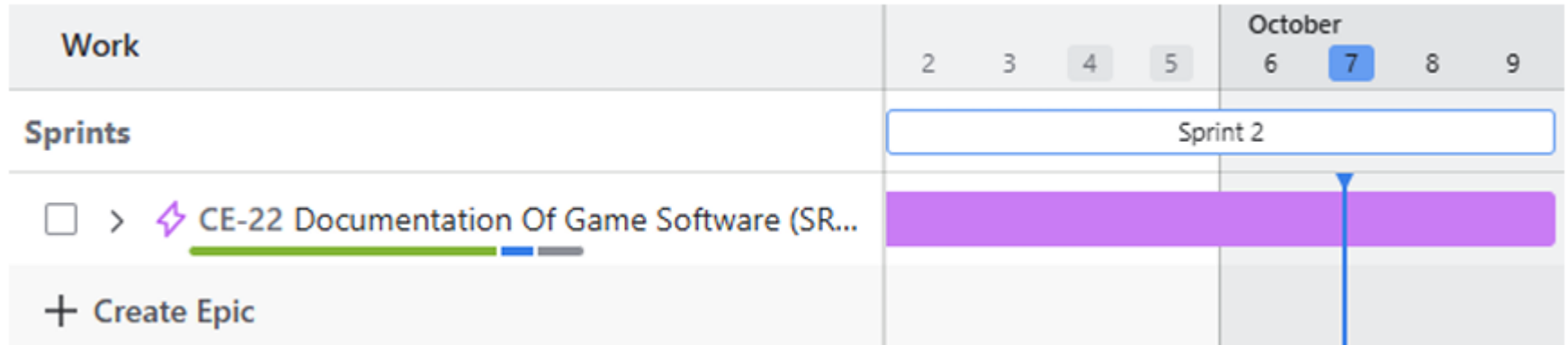
SCRUM Artifacts

The screenshot displays a Jira web interface for a project named 'Create-Engineers'. The main view is the 'Backlog' tab, showing a list of 21 work items. The first item, 'CE-52 Activity Diagram', is highlighted. The right sidebar shows the 'Activity Diagram' details for this issue, including fields for Assignee, Labels, Parent, Due date, Team, Start date, and Sprint.

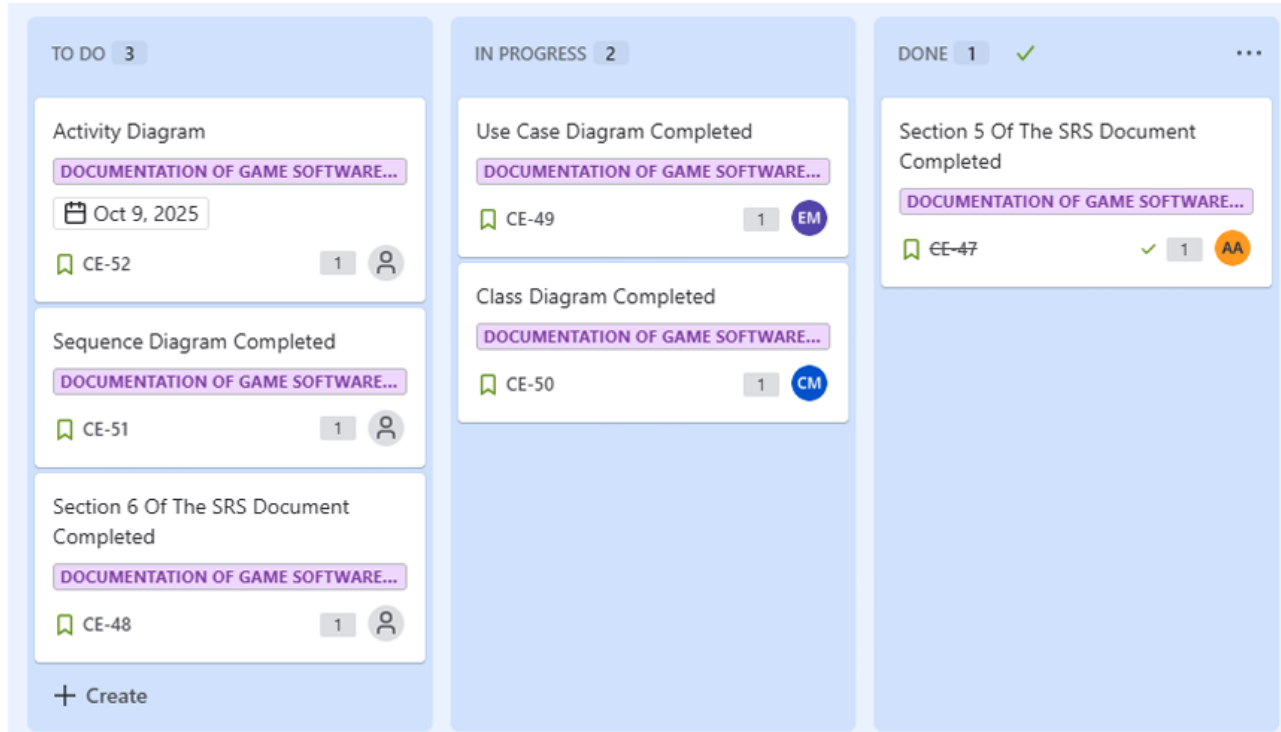
Issue Key	Issue Name	Status	Assignee
CE-52	Activity Diagram	TO DO	Unassigned
CE-51	Sequence Diagram Completed	TO DO	Unassigned
CE-50	Class Diagram Completed	TO DO	Unassigned
CE-49	Use Case Diagram Completed	TO DO	Unassigned
CE-48	Section 6 Of The SRS Document Completed	TO DO	Unassigned
CE-47	Section 5 Of The SRS Document Completed	TO DO	Unassigned
CE-2	Menu	TO DO	Unassigned
CE-4	Standard UNO Deck	TO DO	Unassigned
CE-5	UNO Game Logic	TO DO	Unassigned
CE-6	Sequential Bot	TO DO	Unassigned
CE-7	Player Name Saving	TO DO	Unassigned
CE-8	House-Rules With Menu	TO DO	Unassigned
CE-9	Difficulty Modes	TO DO	Unassigned
CE-10	Adding More Than One Bot	TO DO	Unassigned
CE-11	Bot Names	TO DO	Unassigned
CE-12	Local Multiplayer PVP Function	TO DO	Unassigned
CE-13	Card Customization	TO DO	Unassigned

Figure 3: Sprint 2 - Product Backlog

SCRUM Artifacts



SCRUM Artifacts



Questions?