

# WEB3 Assignment 4

## The Case

(Note: the case hasn't changed – we're doing this again)

UNO is a card game invented by Merle Robbins in 1971. The game can be played by 2 or more players. The game is played as a series of *hands*. After each hand, the winning player is awarded points (in standard rules). The first to 500 points wins the game.

Each hand is played out as follows:

- Every player is dealt 7 cards
- A card is placed face up in the middle of the table – this forms the discard pile
- The remainder of the cards are placed next to the discard pile – this is the draw pile
- Every player in turn must play a card that matches the card on the top of the discard pile
- A card is a match if it has the same colour (i.e. blue) or the same type (i.e. same number or same type of special card) as the other card
- If a player can't match the card, they must draw a card instead.
- The first player to play their last card wins the hand
- The winner is awarded points based on the remaining cards in the other player's hands
- UNO: Before a player plays their penultimate card, they must say "UNO". Failure to say "UNO" is liable to a 4-card draw penalty.

On top of that, there are several special cards with different effects. Also, there are a lot of details not covered above. The rules are uploaded to itslearning.

## The Task

(This is essentially a task of converting the code from assignment 1 to functional)

The task of assignment 4 is to implement the standard rules of UNO as described in the uploaded rule set. (**NOTE:** The special case of 2-player UNO is considered standard for this purpose.)

## Requirements

*Must have*

The solution must be as functional as possible.

1. Define **immutable** types for UNO corresponding to the types you defined in assignment 1
2. Define functions that manipulate data according to the UNO rules
3. Use lodash or immutable.js (or both) for ease of implementation and performance
4. The functions must be pure

#### *Should have*

1. The functions should be written in functional style
2. Implement functions for saying “UNO”

#### *Could have*

1. Define an immutable type to represent a full game of UNO
2. Make as much of the test suite as relevant pass. The test suite is based on my own understanding of the UNO rules.

As before, I have made a test suite based on my understanding of the UNO rules.

#### **NOTES ON THE TEST SUITE:**

- I did not have the time to convert the test suite fully to functional style – do as I say not as I do.
- The test suite uses lodash, but not immutable.js. If you use immutable.js, you may need to change some of the tests.
- It is not a requirement that the tests pass. You may have a different understanding of the rules.
- Feel free to modify the tests as you please.

## Deadlines and Other Rules

- Deadline for the last 3 assignments is 30 November 2025.
- The assignments are meant for groups of 2 or 3 students, but I’ll allow groups of 4 students. If you want to try to do it by yourself that’s okay but be aware that it’s a high workload.
- I cannot give feedback on all assignments, but I’m always available for questions.