**Overview:**

Highcarder is a casual/semi-competitive card game where the main premise is to survive as many “antes” as possible while gathering the highest score possible by playing poker hands from an ever-dwindling deck of cards. The aim of this web application is to develop a simple, yet addicting gameplay loop that would keep the player engaged and aim to reach for the highest possible score and level. The game is inspired by other card games and deck builder roguelikes like Texas Hold ‘em Poker, *Uno (YEAR), Persona 5 Royal’s (20XX)* Tycoon Minigame, Slay the Spire (YEAR), and mainly the recent *Balatro (2024).*

**Gameplay:**

To start, the player is presented with a set of 7 random playing cards. They must make the best poker hand out of the cards given to them to progress a round, out of 5 in total. When a hand is played, the cards in the played hand are discarded permanently. Each poker hand has their own base score (e.g. High Card is 5 points, Pair is 10 points, etc) and they are multiplied by the ranks of the cards that make up the hand (e.g. if the played Flush consists of a 2, 5, 7, 3 and 10, the multiplier will be added up to 27. A Flush is 40 points, so it would be multiplied by 27 to make a total score of 1080).

Figure 1. Highcarder UI (Example gameplay).

To be able to progress onto the next ante, the player must reach the “Score Requirement”. If the player is unable to meet the threshold, they must continue to play hands until they do. If they meet the threshold, they can upgrade their deck by through Boosters, Refinements, Abilities, or Perks. Exceeding the score requirement by multiples allows the player to choose more than one upgrade (Score / Requirement = Number of Upgrades to Choose). The threshold will increase exponentially every ante. After every 5 antes, a debuff will be added to increase the difficulty, ranging from influencing scores negatively, removing cards, or disabling cards or perks from use.

Figure 2. Highcarder UI (Upgrade phase).

There is no win condition, only a lose condition. If the player runs of cards to play, they will lose and are given the option to submit their final score to a leaderboard or play again.

**Usability, User Experience, and Gameplay:**

I wanted to create a game that was simple but also had depth to it’s gameplay. To do this, I looked to games with simple gameplay loops. I settled on a deck-builder card game as one game, Balatro, inspired me as it was coded entirely in the Lua language using the Love2D framework, without using a game engine. To create a similar game, I decided to use JavaScript and the p5.js library as both were like Lua and the Love2D framework, but arguably more powerful as JavaScript featured Classes where Lua does not.

Highcarder only requires one mode of input, which is mouse or touch input to interact with any of the elements on the screen: to click on cards, upgrades, or to navigate the page. This makes it widely accessible to mobile devices.

Window limited to 1200x800 minimum due to scaling issues.

Users will be able to view their scores in the Leaderboard. Hypothetically, scores will be shared across other players, and they will be able to see

Due to the nature of the game being coded entirely on a canvas and in p5.js, the game itself is not accessible via keyboard (arrow keys, enter, etc). keyboard is only used to input name for score

**Technical Implementation**

The code of Highcarder is mostly cantered aroun

Upgrades -> How they are stored, applied, conditions, effects, etc

Burning and Freezing -> How that works

Card Evaluation -> How cards and hands are evaluated

Leaderboard ->

HTML display ->

**Critique**

Sound  
More focus on HTML and CSS – achievements page maybe/unlockable  
More features that had to be cut because going over scope/too ambitious.  
Maybe animations

500 introduction

* Aims, intention, and what the website is about
* How to play the game
* How it stores info and relays them
* Css and html

500 HTML and CSS

1000 words JS implementation

Maybe 250 – 500 words how to improve, what ive learnt etc

BUG -> “Cramped Hand” perk does not work as intended

OBSERVATION -> drawHand() only draws the difference between handSize and the current hand.length. If hand.length is already full from a previous round and handSize has been reduced, no new cards will be drawn and the hand size won’t shrink.

BUG -> choosing an upgrade with a “Gain X for every X card in deck” will not update the counter.  
OBSERVATION -> chooseUpgrade doesn’t have updatePassivePerkDisplay() for packs

New Implementation

Player is able to sort the cards in their hand -> before not a feature  
not having it would caused confusion/headaches.

A computer code with white text

AI-generated content may be incorrect.

Solution -> hand = []  
Result -> unplayed cards will be cleared and the player is required to draw more cards than necessary

Solution -> if hand.length > handSize, shuffle furthest cards back into the deck  
Result -> hand size is being reduced properly

A computer screen shot of code

AI-generated content may be incorrect.

Include a more implementations of the localStorage, better HTML and CSS instead of sole focus on the JavaScript element – incorporating the HTML (such as showing what perks the player had at the moment in a separate toolbar/section)