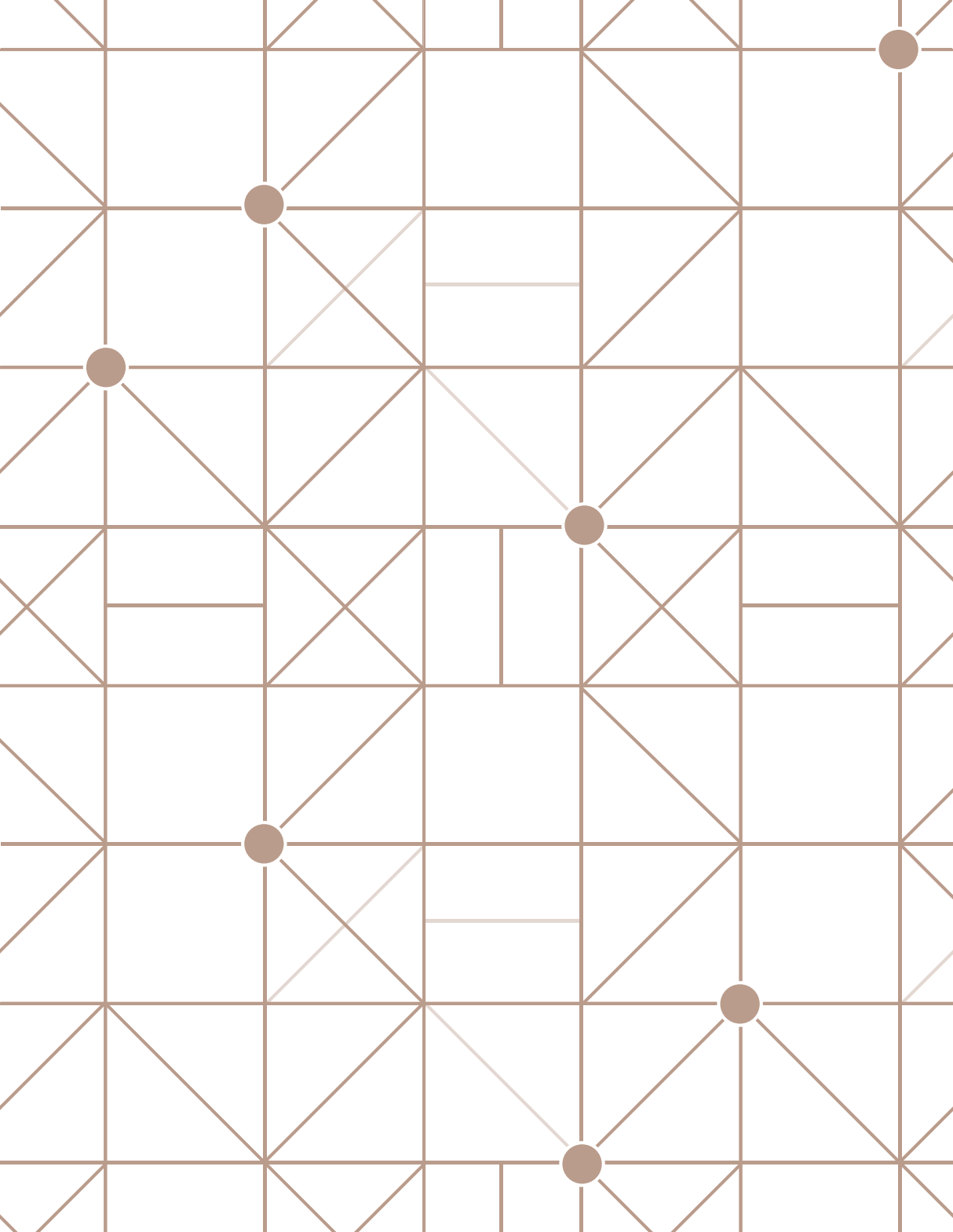


Branch: master ▾ Find file Copy path

project-1 / requirements.md

misteroh Update requirements.md

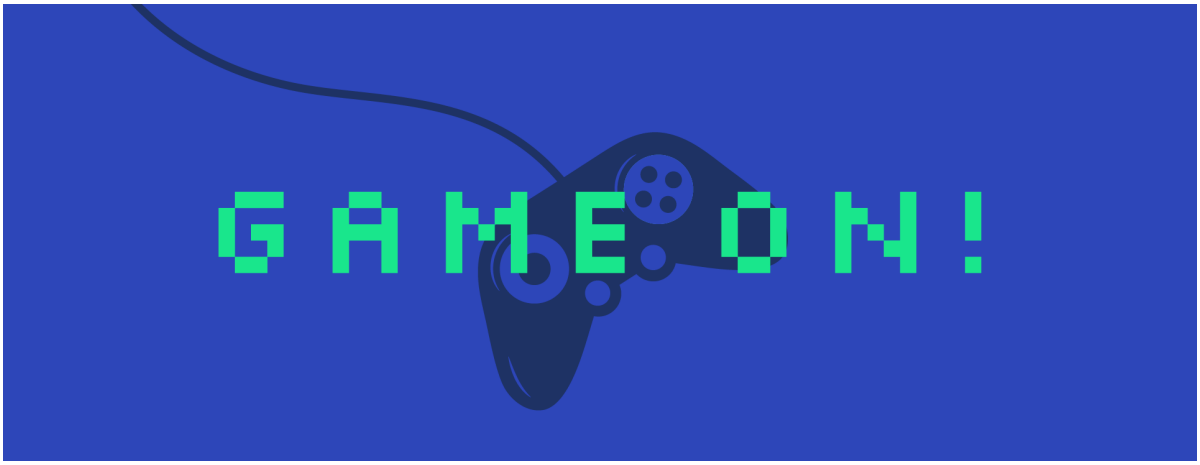
d5be506 3 days ago

1 contributor

Raw Blame History

151 lines (108 sloc) 8.29 KB

Project #1: The Game



📍 Attendance

There are no full day classes during project weeks, but we do expect you to be working on your project daily and be available during the usual class times (10AM - 6PM Eastern) for TA hours or any meetings with instructors as needed.

There are **no attendance checks during this project week, except:**

- **Monday, Nov 23**: Project presentation day! You're required to be in the class zoom starting at 10:00 AM Eastern

Failure to make an attendance check will result in an unexcused absence for both morning and afternoon for that day.

📍 Project Details

Mandatory To Pass:

✅ MVP - Minimum Viable Product

Your game must meet these requirements:

1. Built with HTML, CSS and JavaScript (this game will be played using the DOM, not the console)
2. Hosted on Github pages
3. Commits to Github every day
4. A `README.md` file with explanations of the technologies used, the approach taken, a link to your live site, installation instructions, unsolved problems, etc. Here is a great guide on how to write a readme

✅ Game must have:

1. **Must be a two player game** (either against the computer or against another player)
    - Example: Blackjack: A player plays against the dealer. The dealer is the computer
    - Example: Connect Four: Two players pass the game between themselves to take turns

2. **A win state** - a way for the player to win the game
    - High score can be considered a win state

3. **A lose state** - a way for the player to lose the game
    - Example: Blackjack - a player must be able to lose all of their money with losing hands and cannot play if their bankroll is at 0
    - Example: Connect Four - the other player has won or there are no possible plays left

4. **A way to keep playing if the game is not over**

5. **Multiple rounds to play** - a round must begin, end, and there must be a way to check if the game should continue or the overall game is won or lost
    - Example: Blackjack: a player takes turns playing a hand versus a computer - the player's hand can either win, lose or tie the dealer. If the player has enough money in their bankroll they can keep playing. A player must be able to win several rounds and increase their bankroll
    - Example: Connect Four: two (non-computer) players take turns adding chips to the board. The game will check if a player won or if the board is full and there are no more plays possible. A player gets four chips in a row (vertically or horizontally)- one person wins, one loses, there are no further plays in this case

📍 Stretch Goals (Not Mandatory):

Recommended Features

- A way to reset the board and play again
- CSS to give your game a personal and fun style
- Responsive mobile design
- Work with your instructor to determine additional stretch goals

📍 Make A New Repo

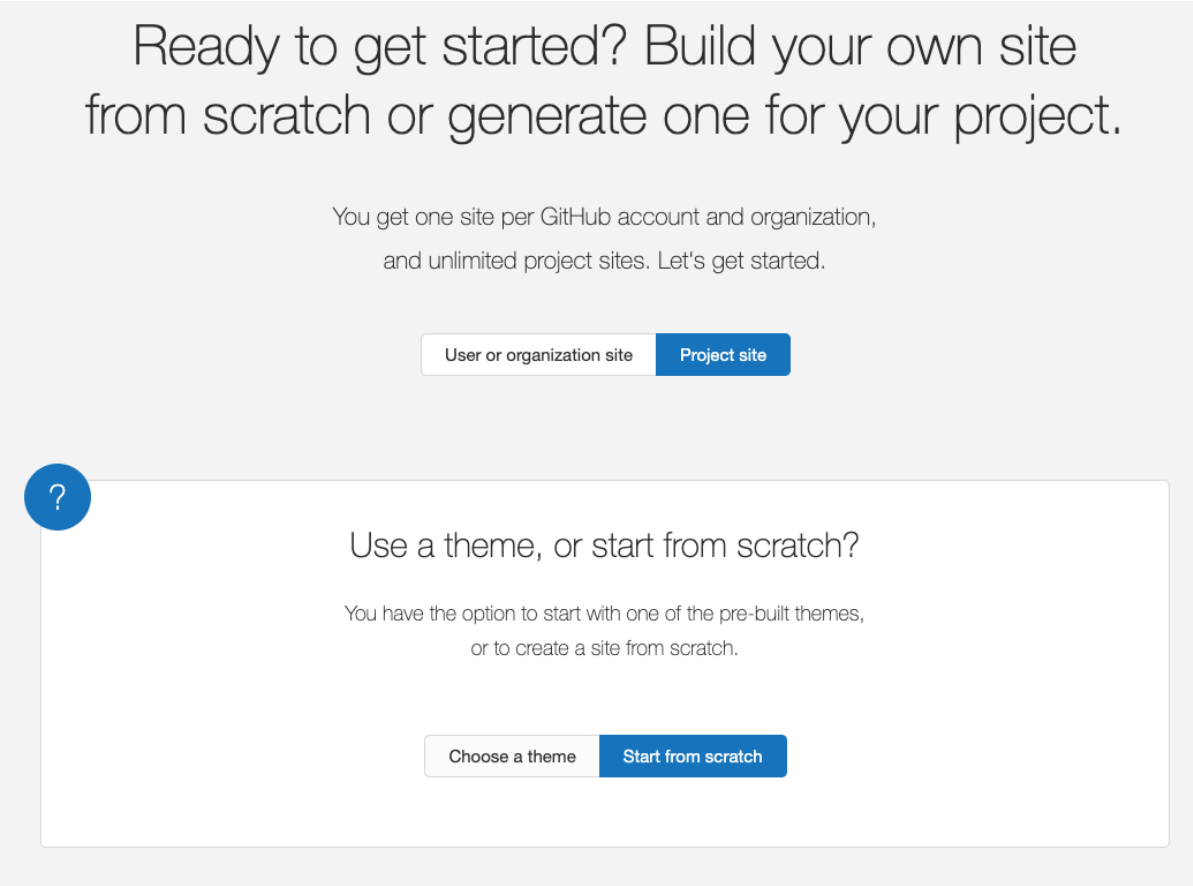
❗ You will be using GitHub, **not** GitHub Enterprise!

❗ **Do not** begin your project within a class repo.

❗ **Do not** clone your project into a class repo.

1. After your project has been approved, make a new Github repo for your project. Remember to keep your repo set to public so you can deploy it.

2. From there, follow the instructions for a Project Site outlined by Github themselves on github.io To get to the correct instructions, select 'Project site' and then 'Start from scratch'



Note: You can create your index.html on Github or you can create it from the terminal (like we've done in class) and push it up, but you will need at least an index.html to deploy your site.

▶ Extra: want your own domain name?

📌 Technical Demonstration

All projects will be presented to the class. Your presentation should:

- Be approximately 5 minutes in length
- Show off all features of the app
- Explain the technical details
- Explain the technical challenges
- Explain which improvements you might make You will be sharing your game and your code. Be prepared to answer questions from the instructors and other students.

📍 Meetings with instructors

An instructor will contact you to setup a meeting time to approve your project.

Monday, Nov 16 - Mandatory

You will meet with an instructor for 15 minutes to get your game idea approved. Be sure to write out what features you will need to build in order to meet MVP and some stretch goal ideas.

📍 How to Submit Your Project

Your project is due on Monday, Nov 23rd at 10:00 AM ET. You will present your project and show your code to classmates and instructors.

✔️ Add your project to this google sheet. Note that this will be the order you present your projects!

❗ Where to go for help during project week

1. Seek out help online
2. Seek out help with your classmates
3. Seek out help with our class TA

❗ TA Hours

1. Kay - Tuesdays: 8 - 10 pm ET || 5 - 7 pm PT `seir-margaret-evan-office-hours`
2. Evan - Saturdays: 2 - 4 pm ET || 11 am - 1 pm PT `seir-margaret-kay-office-hours`
3. Aegean - Sundays: 4 - 9 pm ET || 1 - 6 pm PT `seir-officehours`

📍 Suggested Ways to Get Started

- **Wireframe** Make a drawing of what your app will look like in all of the stages of the game (what does it look like as soon as you log on to the site? What does it look like while the player is playing? What does it look like when the player wins / loses?).

- **Break the project down into different components** (data, presentation, views, style, DOM manipulation) and brainstorm each component individually.
- **Commit early, commit often.** Don't be afraid to break something because you can always go back in time to a previous version.
- **Consult documentation resources** (MDN, jQuery, etc.) at home to better understand what you'll be getting into.

Think about...

- **Creativity**
  Did you add a personal spin or creative element into your project submission? Did you deliver something of value to the end user?
- **Code Quality**
  Did you follow code style guidance and best practices covered in class, such as spacing, indentation, modularity, and semantic naming? Did you comment your code as your instructors have in class?
- **Problem Solving**
  Are you able to defend why you implemented your solution in a certain way? Can you demonstrate that you thought through alternative implementations?

Useful Resources

- MDN Javascript Docs
- jQuery Docs
- GitHub Pages
- Trello

Inspiration - Projects by Previous SEI Students

- CeleZum
- Baseball Triva
- Blackjack
- Connect Four
- Pan Dulce Connect Four
- Simon
- Savage Sailors
- Ancient Wizards

- Trouble in Tribble Town
- Harvest Game

Branch: master ▾ Find file Copy path

project-1 / project-worksheet.md

misteroh Update project-worksheet.md

deda14a 2 days ago

2 contributors

Raw Blame History

117 lines (78 sloc) 5.27 KB

Project Overview

Key Project 1 Dates

- Nov 16 - Project Approvals
- Nov 17 - Nov 22 - Project Week! We will not meet during normal class times. You will be meeting with your squad at the times instructed by your Squad lead.
- Nov 23 - Project Presentations

Project Schedule

This schedule will be used to keep track of your progress throughout the week and align with our expectations.

You are **responsible** for scheduling time with your squad to seek approval for each deliverable by the end of the corresponding day, excluding `Saturday` and `Sunday`.

Day	Deliverable	Status
Day 1	Project Description	Incomplete
Day 1	Wireframes / Priority Matrix / Timeline	Incomplete
Day 3	Core Application Structure (HTML, CSS, etc.)	Incomplete
Day 4	MVP & Bug Fixes	Incomplete

Day	Deliverable	Status
Day 5	Final Touches	Incomplete
Day 6	Present	Incomplete

Project Description

Use this section to describe your final project and perhaps any links to relevant sites that help convey the concept and\or functionality.

Wireframes

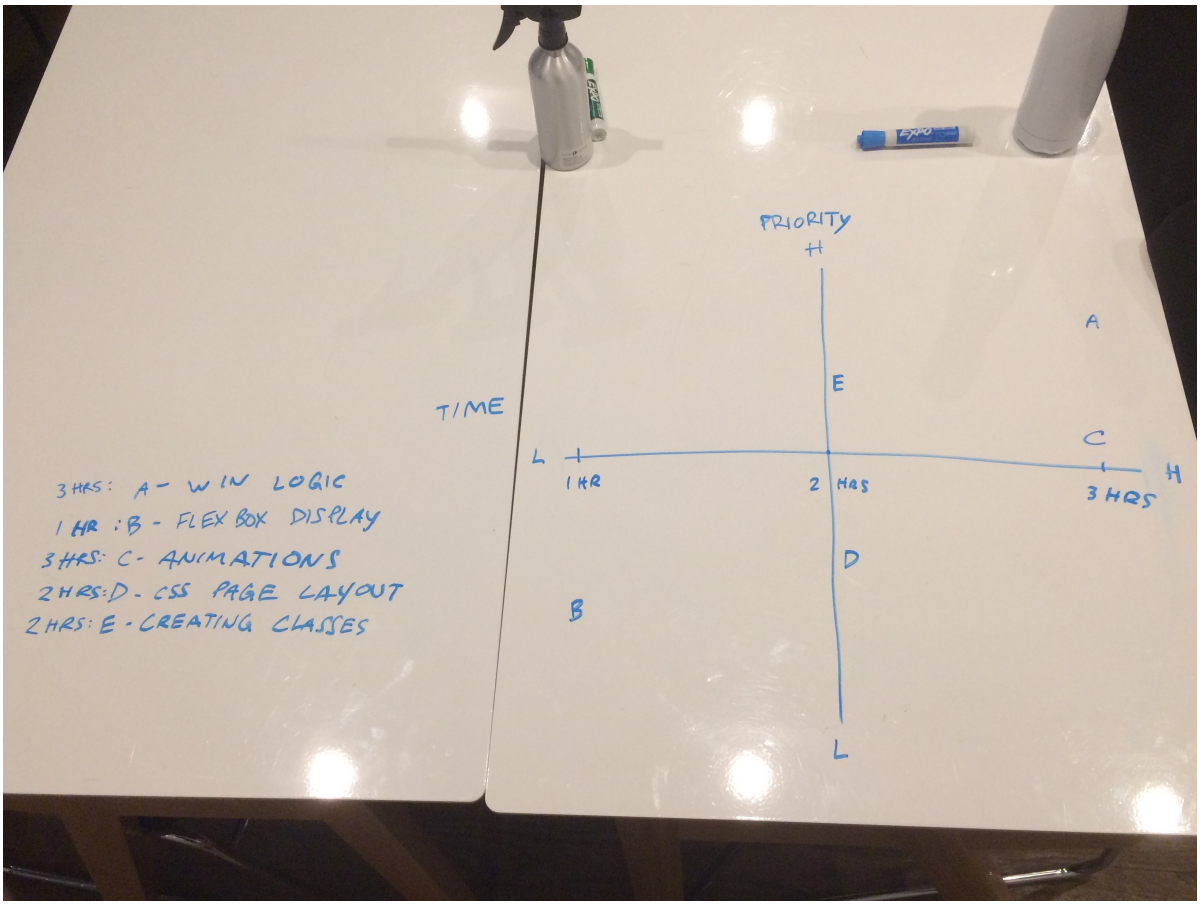
Upload images of wireframe to cloudinary and add the link here with a description of the specific wireframe. Do not include the actual image and have it render on the page.

- Mobile
- Desktop

Wireframing Resources:

- Mockflow
- Figma

Time/Priority Matrix



Include a full list of features that have been prioritized based on the `Time and Priority` Matix. This involves drawing a a square. In the middle of the square, on the x axis draw a line. The most left part of the line should start with 0hrs and the end of the line should include 2hrs. This line will be used to estimate how much time any one feature will take to complete.

Now draw a vertical line on the y axis. The top of this line should have `High` and the bottom `Low` . This line will be used to assign a priority to to each feature you wish to include in the project.

Now create a separate list starting with A and assign it one of the features. Continue to assign each feature a letter. Once complete add each letter to the matrix assigning based on what your feel it's prioirty is an how long it will take to implement. If any one feature takes longer than 2hrs to complete than break it down into smaller tasks and reassign them a new letter.

Once complete tally up the time and determine how long the project will take to complete. Now break those features into MVP and PostMVP so you can guarantee you will have a fully functioning project to demo.

MVP/PostMVP - 5min

The functionality will then be divided into two separate lists: MVP and PostMVP. Carefully decided what is placed into your MVP as the client will expect this functionality to be implemented upon project completion.

MVP (examples)

- Add game player stats to the dom
- Render html on the page
- Allow user to choose player
- Restart the game when the user loses

You should have at least 10 items here

PostMVP

- Anything else that is not MVP

Functional Components

Based on the initial logic defined in the previous sections try and breakdown the logic further into functional components, and by that we mean functions. Try and capture what logic would need to be defined if the game was broken down into the following categories.

Time frames are also key in the development cycle. You have limited time to code all phases of the game. Your estimates can then be used to evalute game possibilities based on time needed and the actual time you have before game must be submitted. It's always best to pad the time by a few hours so that you account for the unknown so add and additional hour or two to each component to play it safe.

MVP

Component	Priority	Estimated Time	Time Invetsted	Actual Time
Modal	H	1hr	1.5hr	-hr
Carousel	H	3hr	-hr	-hr
Adding Form	H	1.5hr	-hr	-hr
Other sections and flex	M	4hr	2hr	-hr

Component	Priority	Estimated Time	Time Invetsted	Actual Time
Game Player HTML & CSS	H	3hrs	2hr	-hr
Create Classes in JS for Enemies	H	3hr	-hr	-hr
Game play main function	L	1hr	-hr	-hr
Total	H	15.5hrs	-hrs	-hrs

PostMVP

Component	Priority	Estimated Time	Time Invetsted	Actual Time
Play game for 3 rounds	L	3hr	-hr	-hr
Interactive Banner	M	4hr	-hr	-hr
Materialize	H	4hr	-hr	-hr
Bootstrap	H	4hr	-hr	-hr
Make own icon	L	4hr	-hr	-hr
Total	H	20hrs	-hrs	-hrs

Additional Libraries

Use this section to list all supporting libraries and thier role in the project.

Code Snippet

Use this section to include a brief code snippet of functionality that you are proud of an a brief description

```
function reverse(string) {
        // here is the code to reverse a string of text
}
```

Issues and Resolutions

Use this section to list of all major issues encountered and their resolution.

SAMPLE.....

ERROR: app.js:34 Uncaught SyntaxError: Unexpected identifier
RESOLUTION: Missing comma after first object in sources {} object

Recollection

A visual memory game!

Player flips two cards that start face down (randomized) if they are a pair player gets pts. if not [next player goes] or [computer goes] or [player goes again]

two player: who ever gets a pair goes again until they miss.
round winner: most pairs
game winner: most points

---

computer: randomly picks w/ same rules

---

single player: against a clock, each round increases dificulty.
round winner: all pairs round lose: clock out
player only play round 3 times before GO
game winner: all levels complete

data

presentation

views

style

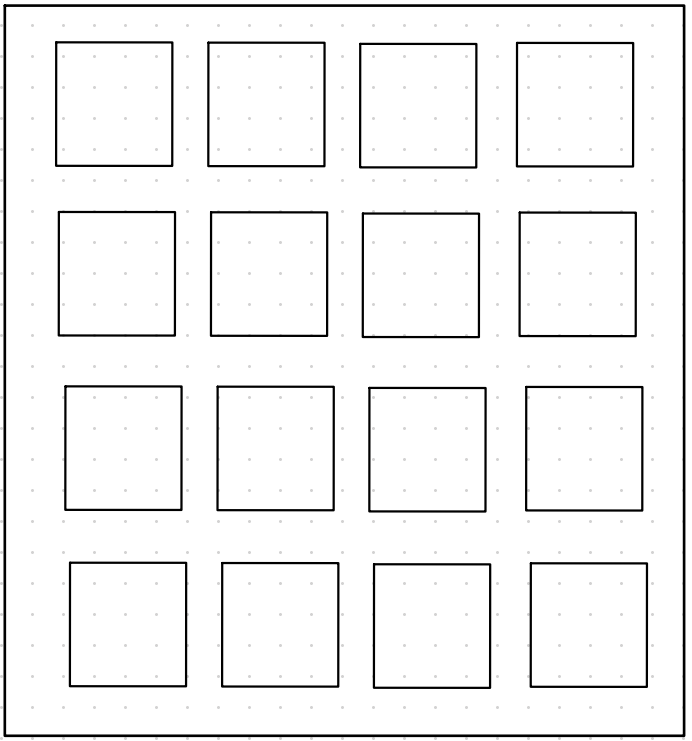
Dom Manipulation

- (create class for pairs (name, dificulty, color)
- if x = y → add pts
- When no more cards → win round
- when no more rounds → Win game
- (clock count down) if clock is out, Lose
- 2 player if player one = more pts && no more cards → player1 Win

Recollection

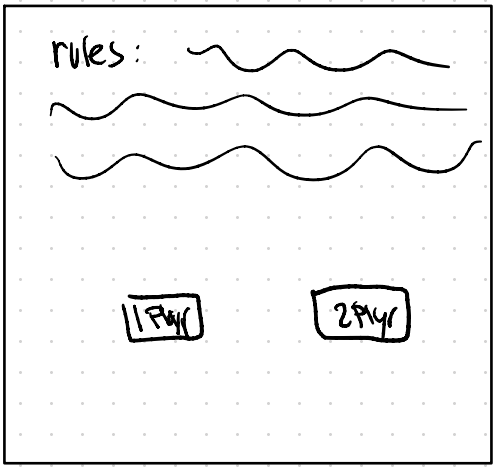
Player 1: /18 | reset | Player 2: /18 | Timer: 30

level 1 / 6



- add difficulty
  - more cards
  - b/w images

Modal



main color

font color

1 Player

2 Player

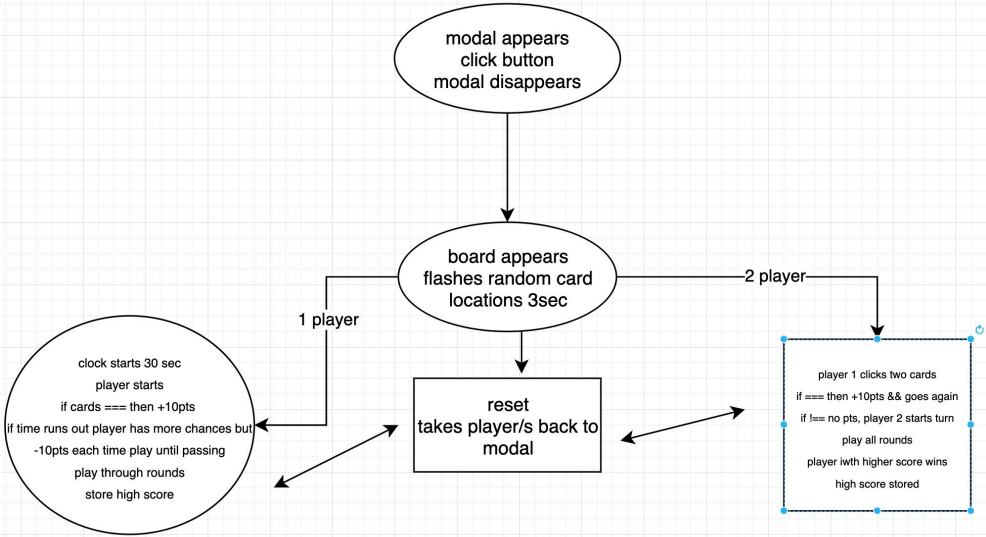
→ buttons for 1 or 2 player

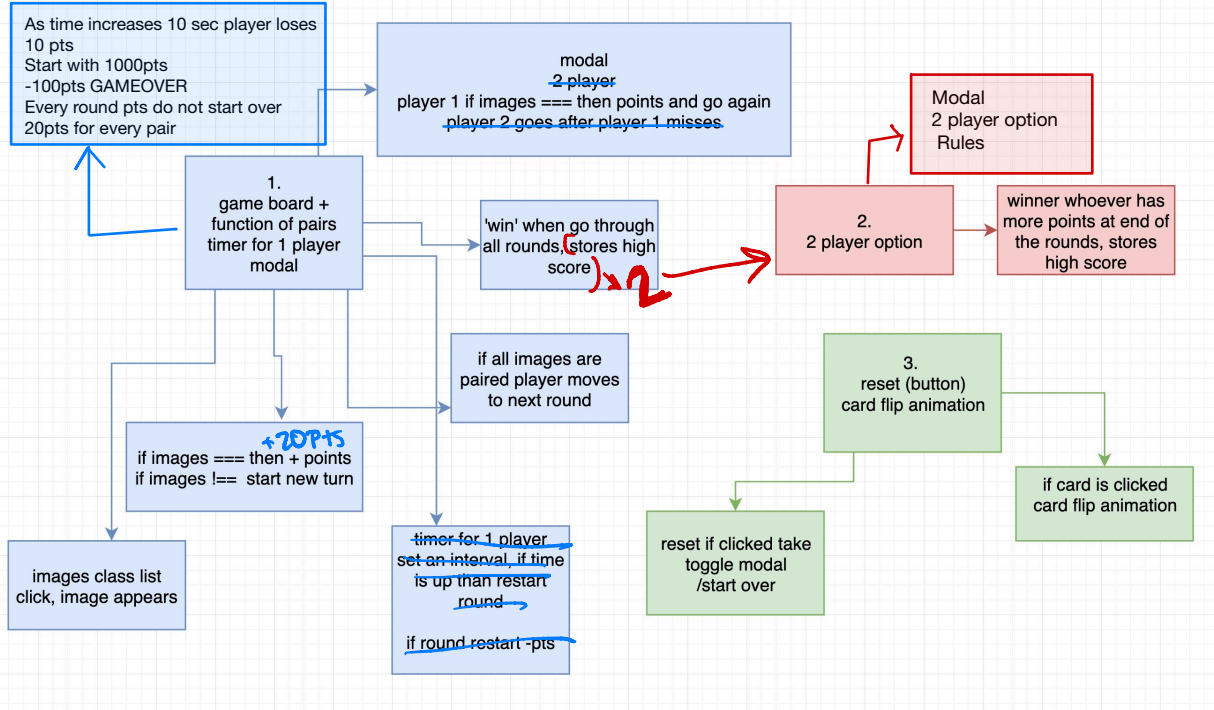
reset → reset game to modal page

Game over — one player ≥ -100

as time increases 5 sec decrease Score -10

Starts at diff per round


modal appears
click button
modal disappears
board appears
flashes random card
locations 3sec
1 player
2 player
clock starts 30 sec
player starts
if cards === then +10pts
if time runs out player has more chances but
-10pts each time play until passing
play through rounds
store high score
reset
takes player/s back to
modal
player 1 clicks two cards
if === then +10pts && goes again
if !== no pts, player 2 starts turn
play all rounds
player iwth higher score wins
high score stored


As time increases 10 sec player loses 10 pts
Start with 1000pts
-100pts GAMEOVER
Every round pts do not start over
20pts for every pair
modal
~~2 player~~
player 1 if images === then points and go again
~~player 2 goes after player 1 misses~~
Modal
2 player option
Rules
1.
game board + function of pairs timer for 1 player modal
'win' when go through all rounds, stores high score
2.
2 player option
winner whoever has more points at end of the rounds, stores high score
if all images are paired player moves to next round
3.
reset (button)
card flip animation
+20PTS
if images === then + points
if images !== start new turn
images class list
click, image appears
~~timer for 1 player~~
~~set an interval, if time is up than restart round~~
~~if round restart -pts~~
reset if clicked take toggle modal /start over
if card is clicked card flip animation

MVP

Component	Est. Time	Priority	Actual
modals	3hr	H	only did 1 – 10 min
Css HTML	5hr	H	~ 2 - 3
card creation	2hr	H	10 +
resets	2hr	L	less than 1
Card [redacted] + functions	15 hr	H	~ 20 +
randomize cards	3hr	H	less than 1
Dom	5hr	H	5+
card flip animation	2 hr	L	1 hr
function (rounds)	10 hr	H	—
round winner functions	5hr	L	—
HS (win) function	5hr	H	—
round timer	3hr	H	1 +
Game over	2hr	M	—
total	60hr		

10hr/day

Schedule

Day	Deliverable	Status
Day 1	Timeline / wireframe work flow	complete
Day 2	Card functions	incomplete took 2-3
Day 3	timer	Complete
Day 4	round functions	incomplete
Day 5	CSS HTML	complete
Day 6	extra	
Day 7	extra	

```
/* ======================
game plans
======================*/
//one player
//page opens => modal pops up to tell directions
//click continue => toggle to game board page level 1
//player starts with 200(?) points
//player clicks play
//timer begins to increase, every 5(?) seconds -10(?) points
//when card is clicked it shows card
//if first card clicked then second card is clicked and is === then they stay face up and add 10 points
//if first card is clicked then second card is clicked and is !=== then they both flip back over after 1.5 seconds
// if first card === true
//if points === -100 GAMEOVER, LOSE
//once all cards are face up GAME OVER
//shuffle cards every game



//=======extra parts for later==========
//once all cards are face up then clock stops, level is complete modal pops up, click continue to next level
//if there are no more levels then GAMEOVER, WIN
//button to make board flash for two seconds before player hits play but -30pts
// first modal → Put in name

☐need to create front and back of card
```

```
class Card {
    constructor (name, roundnum, front, back) {
        this.name = name;
        this.roundnum = roundnum
        this.front = front;
        this.back = back
    }
}
const cow = new Card (cow, 1, frontimg, backimg)
const cow2 = new Card(cow, 1, frontimg, backimg)
```

Questions

for cards use class? or div in HTML?

for rounds div in body?

(class) difficulty = which round

make full card or make in CSS?

How do I get class into board? still be able to randomize location?

(Back === Back — match)

Start/Stop button for timer instead of stopping at pairs? all?

call class / only one / array of objects

loop array