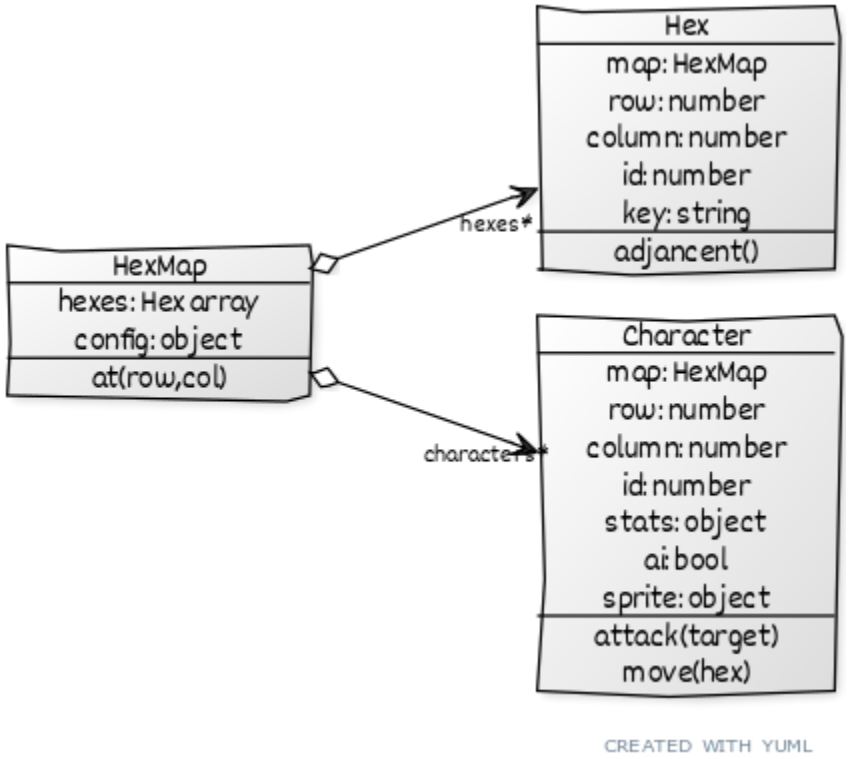
Deliverable 2
Group: Wizard Monkeys

Project Description:

1. "Hex Army" is a digital strategy board game built on a hexagonal grid. The game will be built primarily as a level-based player vs ai game.
2. The game consists of a player controlling units and attempting to destroy enemy units or achieve level-specific objectives.
3. The game is being written in JavaScript using the Phaser3 engine and a general NodeJS production environment. (Capacitor JS will be used for mobile porting)
4. The final project will be available to play on mobile devices, either in browser or as an app.

Section I. System Structure

The game will primarily be structured as a map containing various hexagonal tiles and characters which are able to interact with the grid and attack each other. A simplified overview UML diagram is provided below.



More classes will inevitably be necessary as development continues.

(Perhaps menus or interfaces for specific controls) These 3 core subsystems, however, will drive the game logic.

Section II. Requirements

1. The Game Board
    a. Hexagon-based digital “board” on which the game is played
    b. 20x20 tile grid
    c. Board is represented as a 2-dimensional table, with odd rows visually staggered to display as hexagons
2. Avatars/Characters
    a. Avatars representing the player’s armies will be moveable via some on screen button or other input
    b. Enemy characters will follow scripted patterns and reactions to player character positions.
    c. All characters will display animations when moving and attacking.
    d. Health will be monitored for each character and displayed with their graphic.
3. Inputs
    a. Inputs will be in the form of clicks or taps
    b. Any button inputs should be virtualized (displayed on screen) to maintain mobile support
    c. Enemy moves by calculating shortest path to its target
    d. Characters can move to any of their six neighbors
    e. Inputs outside of the board’s grid should be ignored
    f. Adjacent characters can attack each other, and a prompt is provided to the player to do so
4. Level Structure
    a. Levels should contain a unique map of hexagonal tiles
    b. Levels should contain 1 or more enemies, with a positive correlation between level number and number of enemies
    c. The number of tiles an enemy can move each turn should have a positive correlation with level number
5. Non-Functional Requirements
    a. Performance
        i. Our baseline device for performance is the iPhone 7; Devices with slower processors or less memory are not supported.

ii. The game should run at 30 frames per second on the baseline device
iii. Load time to the first screen should be less than 3 seconds on a 50Mb/s connection

b. Safety
i. Local user data is not transferred online
ii. Online data consists only of game states and does not include any personal data

c. Security
i. Users only have access to store data (game states) under their own unique id
ii. Attempts to modify any cloud data not assigned to the user’s id will be rejected

6. Software Quality Attributes

a. Reliability
i. The system should not crash on common use cases
ii. Connection issues should resolve cleanly and display errors on screen

b. Maintainability
i. Project source code should follow OOP paradigm
ii. Project source code should utilize JSDoc comment structure
iii. Developers should utilize the GitHub repository for version control

c. Portability
i. The system will be portable to any device with webview, such as Android, iOS, and most desktop devices

7. Interfaces

a. User Interfaces
i. Player can select level to play
ii. Player can load their most recent game
iii. Player click the pause button, the game pauses. When he click the restart button, the game restarts.
iv. Player clicks the exit button; the application is closed.
v. Game setting
vi. Player clicks the setting button, he could set up options.

vii. Players can determine the level of sound.
viii. Player can define his/her username
ix. Game control
x. Player touch the mobile screen to control game

b. Hardware Interfaces
i. Players will need a browser or device with a webview

c. Communications Interfaces
i. Players will need an internet connection

Section III. 3 Phase Development Plan

Phase 1

We can be able to see following by the end of phase one

1. Hexagonal grid(20*20)
2. Front page with main menu
3. Player Characters and enemies
4. Health bar
5. Possible Sound effects
6. buttons and touch(or either of these)

Phase 2:

1. In Phase 2 following will be tasks
2. Placement of enemies
3. Shortest path towards the player
4. Weapons
5. Testing the developed code

Phase 3:

1. Level editors
2. CapacitorJS for different platform
3. Full and final testing

Section IV. Member Contribution Table

Member	Contribution description	Overall Contribution (%)
Sravanthi Tummala	UML Diagram and Description	12.5%
Vinay Kumar Bathula	UML Diagram and	12.5%

	Description	
Jeremy Glebe	Requirements specifications Editing	12.5%
Haiyi Wang	Requirements specifications	12.5%
Prudhvi Krishna Jarabani	project Development phases	12.5%
Vishnu Sai Konka	Project Development phases	12.5%
Jaswanth Korapati	Update the meeting and discussion about the project	12.5%
Anand Paul Kamadana	Update the meeting and discussion about the project	12.5%

Section V: Meetings

Meeting minutes are found in the group repository.