**Brief Analysis**

**Battle Royale**

We are going to make a battle royale game(eg. Fortnite, Player unknown battlegrounds, etc.). This genre of games recently saw a huge surge in popularity, especially because of streaming. The premise of these games is pretty simple. A hundred players are spawned into one big battleground and they battle each other in a big free for all to ultimately have one winner.

**Pokémon**

Another game we want to adapt is Pokémon. Pokémon is the biggest media franchise in the world, raking in a total revenue of $95 billion. The slogan often associated with Pokémon is “Gotta Catch ‘Em All!”. In the games you can catch and battle with certain creatures called “Pokémon(pocket monsters)”. In our game you will be the Pokémon yourself and you can perform certain attacks against other players, who are also playing as Pokémon. In the games every Pokémon has a limit of four attacks learned at once, this will also be the case in our game. The Pokémon you catch can level up by earning experience points, if the Pokémon reaches a certain level, it will evolve into a stronger Pokémon with a different form. This mechanic will also be implemented in our game. Every Pokémon has a certain type(e.g. Electric, Water, Fire etc.). Our Pokémon will also have types and also type advantages in battle will be implemented in our game.

**Design**

**Game design**

Rules:

* Certain amount of players are spawned into a game world.
* They have to battle against each other, to ultimately have one winner.

Mechanics:

* The player can attack by clicking the left mouse button.
* The player can move with the W, A, S and D keys.
* The player can pick up moves in the form of Discs(TM’s in Pokémon).
* The player can evolve by getting a certain amount of experience.
* The player can jump by pressing the Space key.
* The player can pick up items by pressing the E key.

Art and Sound:

* Took a lot of inspiration from Pokémon.
* Simple terrains.

UI:

1. Gameplay:
   1. The player can see a crosshair in the center of his screen, so he knows where he is aiming.
   2. A Health bar in the bottom right/left corner of the screen.
   3. A move indicator in the bottom right/left corner of the screen.
   4. A kill count in the upper right/left corner of the screen.
   5. A number telling the player how many opponents are currently alive in the upper right/left corner of the screen.
   6. …
2. Pre-game Menu:
   1. A start screen with a start button, so the player can start the game and join a lobby.
   2. A lobby screen with a countdown timer, a ready button and a counter for how many players have joined.
   3. A Character Select screen, where the player chooses his character.
      1. On the left side of the screen the player can see his chosen character and his opponents chosen characters.
      2. On the right side of the screen the player can see all the different characters and their icons. The player can click on them to choose his character.
      3. After every player has chosen his character or the timer becomes zero, the game will start.

**Technical design**

* Programming language: C#.
* Game engine: Unity.
* Events:
  + Player input.
  + Collision.
  + Dying.
  + Picking items up.
  + Evolving.
  + Using a move.
* Data:
  + Player variables are stored locally.
  + Assets are stored locally.
  + Information that needs to be sent to other players, will be sent to over the network.
* Class diagram...

**User Requirements**

* **Must Have:**
  + Player Movement,
  + Shooting mechanics,
  + Multiplayer,
  + Evolutions,
  + Character selection + Characters,
  + Inventory system / items,
  + Working Network,
  + Map,
  + Death system, win system
  + UI, --
  + PlayerSystem,
  + Moves.-
  + Audio,--
  + 3 different Pokemon + evolutions.--
* **Should Have:**
  + Being immortal while evolving,
  + A set of different attacks,
  + Spectating System,
  + Killfeed and remaining people,
  + Shrinking circle ”storm”.
  + Options Keybinds
* **Could Have:**
  + Multiplayer more than 6 players,
  + Match stats
  + terrain effects,
  + Multiple maps,
  + Mobile port,
  + Poison + antidote,
  + Vehicles,
  + Weather effects,
  + Pokecenter,
  + Pokemon trainers,
  + NPC’s,
  + Teams,
  + 6 different Pokemon,
  + Shinies,
  + Ice pokemon turning water into ice,
  + Flying pokemon
* **Won’t Have:**
  + A level system,
  + Split Screen,
  + Pokemonkart gamemode.

**Project Analysis**

**Risks**

* Copyright issues with GameFreak™.
* Collisions won’t feel right over the network.
* Network not responsive enough.
* Not enough time for all characters and evolutions

**Planning**

*//A planning for the remainder of the period. Indicate how you are going to spend your time. Set deadlines for when certain user requirements have to be finished. Think about who is going to work on which component.*

**Netwerk overal toepasbaar met zijn allen.**

**Lucas: Moves (together with karst)**

**Sietze: fix shooting mechanics, Playersystem**

**Lars: item system (UI with luc)**

**Karst: Evolving System, Death system (together with Lucas)**

**Luc: Gui(HUD, UI, Menu’s), audio(later, together with Momo).**

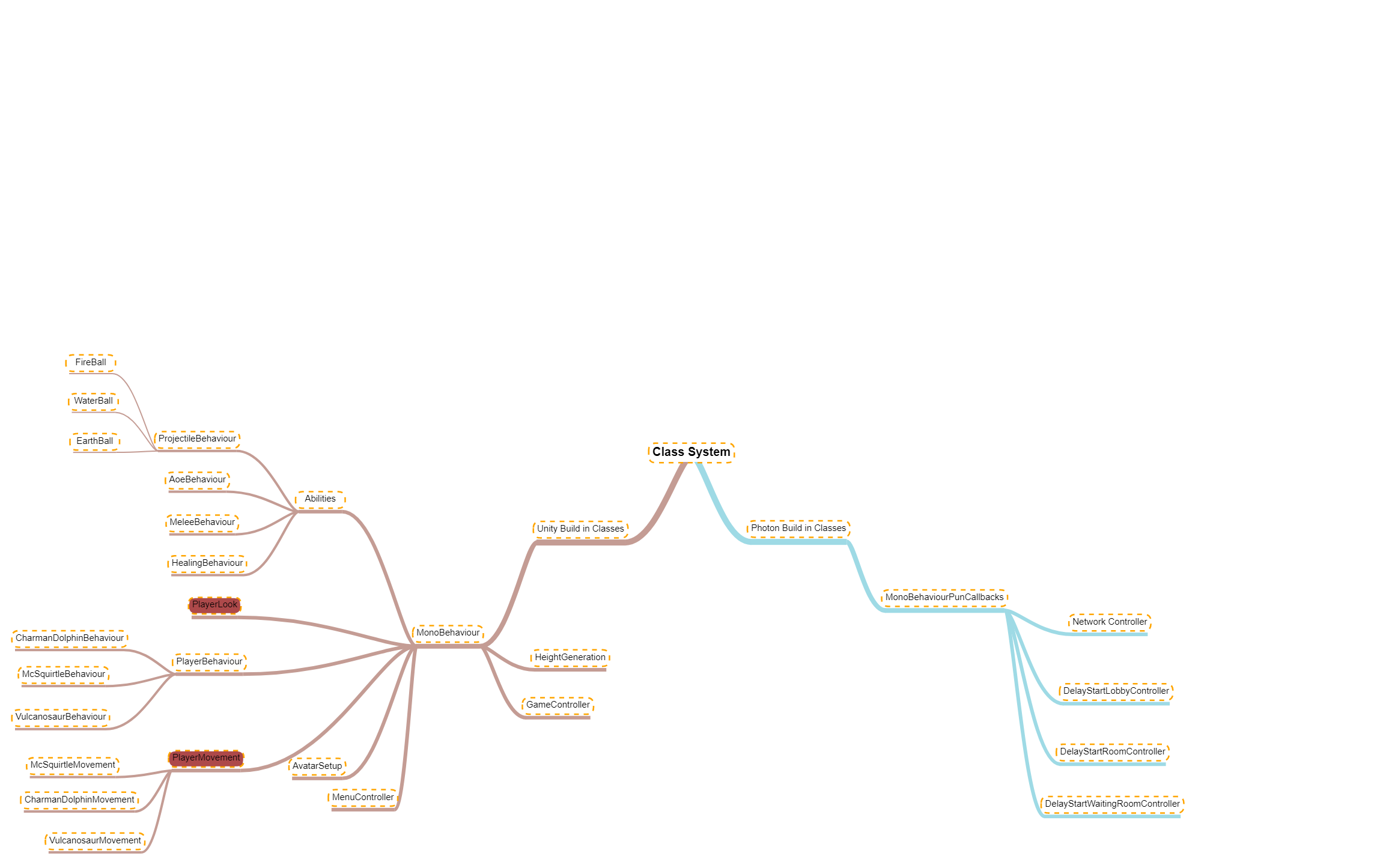
**Momo: Netwerk master for questions, audiosystem (not a main-priority)**

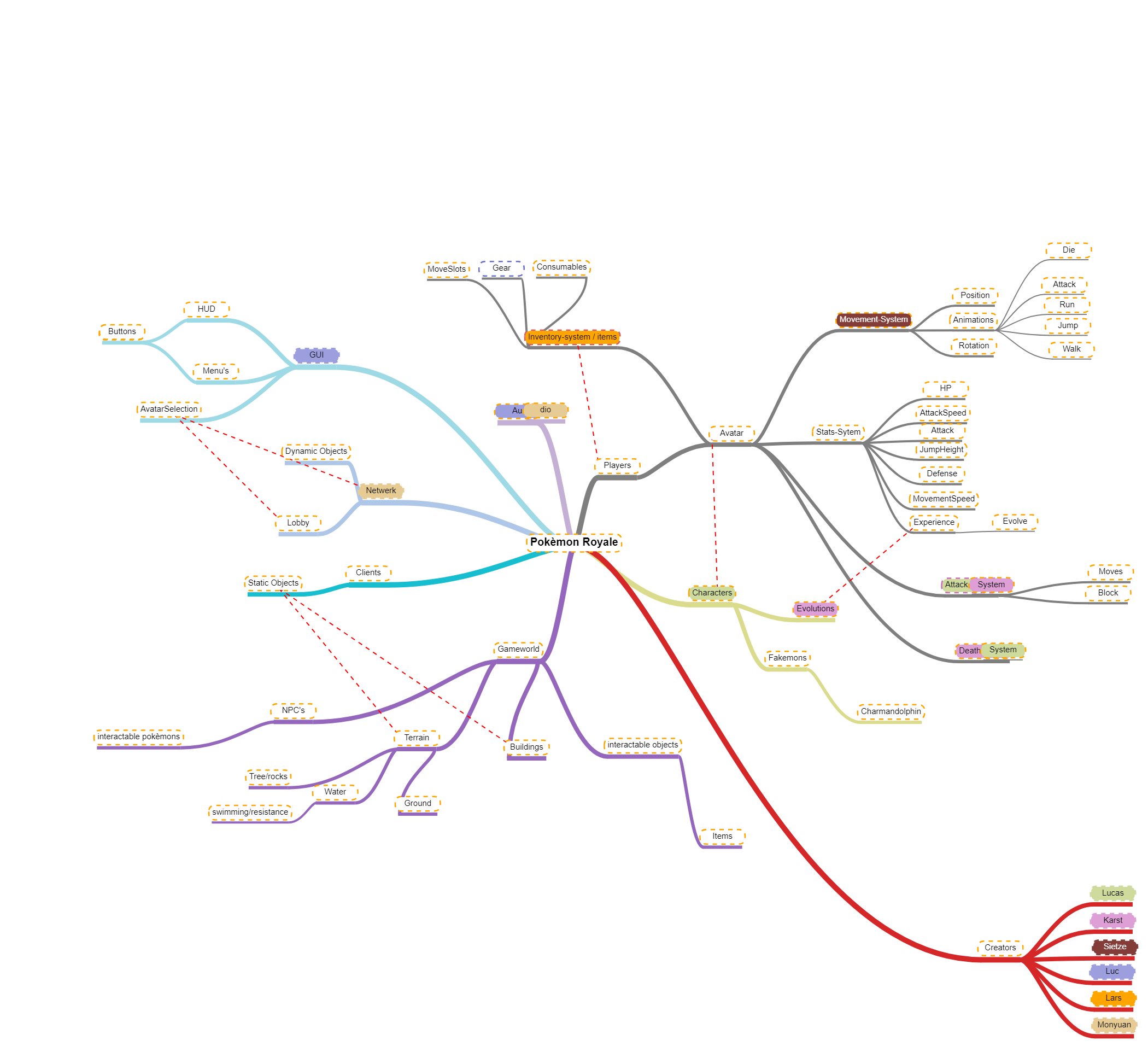
**Tools**

* Visual Studio: for coding.
* Unity: building the game
* Blender3D: making the models.
* GitHub: distributing the code.
* PhotoShop: Texture Editing.

**How we structure everything:**

**Class system:**

****

**Division of tasks**

**Teamstructure:**

* **Lucas = Project Leader**
* **Sietze = Git-master / Co-Leader, Specialist**
* **Karst = The Critic / Caretaker**
* **Lars = Teamplayer**
* **Luc =**
* **Momo =**