Full Stack Challenge

Nordeus - Job Fair 2026

Welcome to the team

Imagine you've just joined a gaming studio as a Full Stack Engineer- congrats! You're the only engineer on a small team, which means you own both the client and the backend. Your Game Designer has already fleshed out the concept and has a working vision. Your job is to build it.

The game

Your Game Designer pitched a turn-based RPG where the player controls a hero fighting through a gauntlet of 5 monsters. The monsters are the levels — face them one by one, in order. Beat them all, win the run.

Combat is turn-based: the hero picks a move, then the monster responds, back and forth until someone's HP hits zero. The hero starts with a default moveset. After beating a monster, they learn one of its moves at random — and can equip it before the next fight.

The player can replay the fight as much as they want to earn experience in order to level up or to try to learn another move from the monster.

What you need to build

Client

Main Menu

- Start a new run
- Exit the game

Map / Run Overview Once a run starts, the player sees all upcoming encounters. They can:

- Click an encounter to enter the battle
- View their current equipped moves
- Open a move management screen to swap moves in and out from everything they've learned so far

Battle Screen

- Visual representation of both the hero and the monster
- Both characters display their current HP
- The player selects a move each turn; the monster responds

Post-Battle

- If the player loses nothing happens
- If the player wins: They're shown which monster move they've learned
    - That move is now available in the move management screen for future battles

Progression

- The hero starts at Level 1 with a default moveset
- Each battle awards XP; enough XP triggers a level up
- Leveling up increases stats: **Attack**, **Defense**, **Health, Magic**

Server

Your Game Designer wants to be able to tweak battle configurations and bot behavior without waiting for a new client build. That's where the server comes in. Keep the game logic server-side so changes can be made fast. Two endpoints are required:

A GET request response to fetch the configuration of the run. Called once at the start of a run. Returns the full battle configuration: which 5 monsters the player will face, their stats, movesets, and anything else the client needs to set up the run.

A GET request response to fetch the next move by the monster. Called after the player makes their move each turn. The server receives the current battle state and responds with the move the monster will play. The client then applies it.

Game Systems

Stats

Every character - hero and monster alike - has four stats:

- **Health:** Hit points. Reach zero and you lose the fight.
- **Attack:** Scales physical damage moves.
- **Defense:** Reduces incoming physical damage.
- **Magic:** Scales magic damage and healing moves.

The hero's stats increase on level up. Monsters have fixed stats defined in the server config. This means you can create your monsters without a concept of a level if you want and just generate a monster of appropriate difficulty by assigning sufficient stats to it. What appropriate difficulty here means is that *generally* it should be easier to beat the monster on level 1 than the monster on level 5 for example.

Moves

A move has a type (Physical or Magic), an effect (damage, heal, buff...), and a base value. When a move resolves, its output is calculated using the relevant stat:

- Physical moves scale off **Attack**, and are reduced by the target's **Defense**
- Magic moves scale off **Magic**, and bypass Defense entirely

The full moveset, all monster moves and the knight's default moves, is defined below. You are free to experiment with the exact numbers on these moves. Tweak the moves until you feel like the experience is fun.

Note: you can tweak the moveset of the characters a bit in your implementation. So if you find another character other than the knight interesting and would like your hero to be that character you can think of a couple of moves and make it happen. If you're

going to do this though, pay attention that you still end up with moves that are using the systems you have built. For example, don't disregard the Magic stat in your moves just by making every move physical.

Knight (default moveset) 

- **Slash:** Physical, deals moderate damage. Scales off Attack, reduced by target's Defense.
- **Shield Up:** No damage, raises the knight's Defense for two turns.
- **Battle Cry:** No damage, raises the knight's Attack for two turns.
- **Second Wind:** Heals the knight for a moderate amount. Scales off Magic.

Witch 

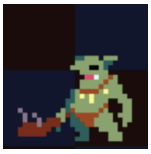
- **Shadow Bolt:** Magic, deals heavy damage. Scales off Magic.
- **Drain Life:** Magic, deals light damage and heals the witch for the same amount. Scales off Magic.
- **Curse:** Lowers the hero's Attack for two turns.
- **Dark Pact:** No damage, raises the witch's Magic for two turns at the cost of some of her own HP.

Giant Spider 

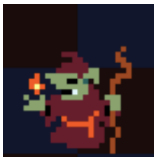
- **Bite:** Physical, deals moderate damage. Scales off Attack, reduced by Defense.
- **Web Throw:** Physical, deals light damage and lowers the hero's Defense for two turns. Scales off Attack, reduced by Defense.
- **Pounce:** Physical, deals heavy damage. Scales off Attack, reduced by Defense.
- **Skitter:** No damage, raises the spider's Defense for two turns.

Dragon 

- **Flame Breath:** Magic, deals heavy damage. Scales off Magic.
- **Claw Swipe:** Physical, deals moderate damage. Scales off Attack, reduced by Defense.
- **Intimidate:** No damage, lowers the target's Attack for two turns.
- **Dragon Scales:** No damage, raises the user's Defense for two turns.

Goblin Warrior 

- **Rusty Blade:** Physical, deals moderate damage. Scales off Attack, reduced by Defense.
- **Dirty Kick:** Physical, deals light damage and lowers the target's Defense for two turns. Scales off Attack, reduced by Defense.
- **Frenzy:** No damage, raises the user's Attack for two turns.
- **Headbutt:** Physical, deals heavy damage. Scales off Attack, reduced by Defense.

Goblin Mage 

- **Firebolt:** Magic, deals moderate damage. Scales off Magic.
- **Arcane Surge:** No damage, raises the user's Magic for two turns.
- **Mana Drain:** Magic, deals light damage and lowers the target's Magic for two turns. Scales off Magic.
- **Hex Shield:** No damage, raises the user's Defense for two turns.

Game Designer's notes - bonus content

Your GD has a backlog of ideas they want to explore after the prototype is solid. If you get through the core, here's what's next on their list:

1. **Move descriptions** - It would be nice if hovering over the move brings up the move description so you don't have to know what it does in advance.
2. **Attribute choices on level up** - Instead of fixed stat increases, let the player choose which stat to boost each time they level up, giving them more control over how their hero develops.
3. **Status effects** - Bleed, Poison, Damage Reduction, Damage Increase baked into moves.
4. **Resource costs** - Some moves cost HP; others cost mana. Figure out mana regen for the hero, and whether monsters use it too.
5. **Save & Exit** - Let the player save mid-run and return later.
6. **Battle log** - Show a running list of all moves played by both sides during a fight.
7. **Battle animations** - Make the hits feel good.
8. **Smarter bot** - The GD wants to test bots that play situationally (e.g. prioritizing heals when low HP) rather than randomly.
9. **Items** - Drop from monsters just like moves; equippable via the same system.
10. **A shop** - Spend in-run currency on direct upgrades instead of grinding fights.
11. **More enemies & moves** - Expand beyond the base set.
12. **Non-linear map** - Break the 5-encounter limit; generate branching paths like Slay the Spire's map
13. **Environmental effects** - Battles affected by where they take place.
14. **Endless mode** - Infinite encounters, see how far you can go.
15. **Hero classes** - A character selection screen that lets the player choose the Hero they take into battle. Heroes can differ by base moveset and attribute increases per level for example.

Assets

You can download some helpful assets for the characters from this link and for the moves you can use this asset or this site, for example. Of course, you are free to find or create your own assets.
You will not receive negative points during grading based on the looks of your game but we do value usability and readability of the things on screen 🙂

Submissions

- Should be sent via email to jobfair@nordeus.com. The email subject should be: **FullStack Challenge**. Please add your **full name** to the email, as well as the link to your **LinkedIn profile**, if you have it.
- Can be in any format (e.g. a link to your Github repo, zipped project, a link to your web page if you're making a web application, or anything else) but please, make it easy for us to access and view your work.
- It would help us a lot if you could provide a screen recording of you playing the game to make sure we see all the features you've implemented. Feel free to work in any environment you're comfortable with.

Interested in the Full Stack internship? We have great news! If you successfully complete the challenge, then you will skip one round of interviews for the Full Stack internship if you apply for it.

The challenge is open until May 3rd, 2026, end of the day. Good luck!