Backend Development

The main reason for this task is to measure proficiency in programming and ability to learn new technologies under time pressure. You don't have to complete all the requirements. You can move on to the next part, that requires a slightly different skill set, at any time. It's important that you present your strongest side in this limited time.

Using Python and either Flask or Tornado framework implement the backend service for registering players, providing leaderboards and accepting battle requests. Then, using language and framework of your choice, implement engine (worker / job / script etc) for processing submitted battles.

1. Model the following in the primary database
    a. Player - should consist of at least the following
        i. Identifier
        ii. Name (max length 20 characters, unique)
        iii. Description (max length is 1k characters)
        iv. Amount of gold (max value 1 bln)
        v. Amount of silver (max value 1 bln)
        vi. Attack value, Hit points and luck value
2. Implement the web application with the following endpoints / handlers
    a. Create player
        i. Validate and store details in database
    b. Submit battle - opponents
        i. Put the battle request into battle processor queue
    c. Retrieve leaderboard
        i. Retrieve list of all the players on the leaderboard, each entry consisting of rank/position, score and player identifier
3. Implement the battle processor
    a. Players will submit battles, specifying who they want to attack. Those submissions should be processed by a battle processor that needs to meet the following requirements
        i. Battles must happen in the order that they were submitted
        ii. Each battle should only be executed once
        iii. No battles should be missed
        iv. Battles should be processed as soon as possible, but there is no hard requirement
    b. Processor should implement at least the following steps for each battle in the queue
        i. Process the battle - see sections 2 below
        ii. Battle should be logged
        iii. Resource should be subtracted from losing player
        iv. Resource should be added to winning player
        v. The total amount of resources stolen should be submitted to the leaderboard

Backend extension for Game Server Development

1. Implement the battle engine
    a. The battle
        i. The battle system is turn based, with whoever submitted the battle going first
        ii. With each attack, the attacker must calculate the damage it can do. The attack value is proportional to the players hit points. The lower the hit points - the lower the attack, with a cap at 50% of the base attack value.
            1. For example: with the loss of 1% of health, reduce attack by 1% rounded up, so starting health 100, attack 70, every time 10 damage is taken, the attack value is reduced by 7. In this scenario, the attack can never be reduced below 35 which is 50% of the base attack value.
        iii. The attacker then becomes the defender, and the defender the attacker.
        iv. Use the luck value of the defender to decide if an attack misses.
        v. The battle continues until one of the players hit points value reaches zero.
    b. Resources stolen should be between 10% and 20% of their total amount of resources, evenly distributed across their resources
    c. A battle report should be generated - which contains a detailed log of everything that happened during the battle.
    d. Implement concurrent execution of non-conflicting battles
2. All endpoints should be protected against public/unauthorised access in some way.