Backend engineer candidate technical problem

For this exercise, you will be coding a simulation of the game of casino style Blackjack. If you are currently unfamiliar with that game then you can familiarize yourself with the rules of play via the following Wikipedia link: https://en.wikipedia.org/wiki/Blackjack#Rules_of_play_at_casinos. T he rules are also outlined below. Only focus on the basic rules and ignore things like double down, split, surrender and insurance. What we are looking for here is your ability to model something from the real world with code as your ability to code the logic for a basic set of
requirements. When done, we are expecting to see coherent object oriented class structure representation of the major components of the game (i.e. Player, Dealer, Card, Deck, etc...) as well as encapsulation within each of those components of the operations and logic that each will contribute to the game simulation. The only additional requirements we are going to give are:

A simulation run will always have one dealer and at least two players.
Each player in the game has a finite amount of money. When that money is exhausted, they can no longer participate in that run of the game simulation.
A player will also stop playing in that simulation run when they reach a certain amount of winnings.
If there are no more players left (either they all ran out of money or they all hit their money total or a mix of both) then the game simulation run is over
We expect there to be some main entry point to kick off a game simulation. When running a simulation, what is happening in the game should be conveyed by printing to the console. It should be obvious, by reading the console output, what is happening in the game simulation.
The dealer needs to abide by standard House Rules; hitting on anything under 17 and standing on anything 17 or above
The non-dealer players should be setup to follow a strategy that is configured at time of player creation. Two possible strategies are:

Conservative: Standing on anything where the total is greater than 10; hitting on anything where the total is ten or below

Aggressive: Hitting on anything under 18
You do not need to support a multi deck shoe. Just a single deck will suffice. If the deck runs out it needs to be reshuffled. Mid round re- shuffle is OK
You need to demonstrate some level of unit testing within this exercise. At a minimum, the logic from one of the main components of the simulation needs to have a unit test associated with it.

The rest is up to you. You don't need to nail all the nuances and rules of the game perfectly. What's more important to us is how you go about structuring and developing the code. If you have questions about anything, it is expected that you ask them to your point of contact

Basic Blackjack Rules:

The goal of blackjack is to beat the dealer's hand without going over 21.
Face cards are worth 10. Aces are worth 1 or 11, whichever makes a better hand.
Each player starts with two cards, one of the dealer's cards is hidden until the end.
To 'Hit' is to ask for another card. To 'Stand' is to hold your total and end your turn.
If you go over 21 you bust, and the dealer wins regardless of the dealer's hand.
If you are dealt 21 from the start (Ace & 10), you got a blackjack.
Blackjack means you win 1.5 the amount of your bet.
Dealer will hit until his/her cards total 17 or higher.
If both the player and the dealer have a tie - including with a blackjack - the bet is a tie or “push” and money is neither lost, nor paid