**Users with Bank Accounts**

Objectives:

* Practice writing classes with associations

Update your existing User class to have an association with the BankAccount class. You should not have to change anything in the BankAccount class. The method signatures of the User class (the first line of the method with the def keyword) should also remain the same.

For example, our User class currently has a method like this:

*class User:*

*# other methods*

*def make\_deposit(self, amount):*

*self.account\_balance += amount # hmmm...the User class doesn't have an account\_balance attribute anymore*

But our User class no longer has a self.account\_balance attribute. Instead, we have replaced this with an *instance* of a BankAccount by the name of self.account. That means our make\_deposit (and other methods referencing self.account\_balance) need to be updated! That's the goal of this assignment.

Remember in our User methods, we can now access the BankAccount class through our self.account attribute, like so:

*class User:*

*def example\_method(self):*

*self.account.deposit(100) # we can call the BankAccount instance's methods*

*print(self.account.balance) # or access its attributes*

* Top of Form
* Update the User class \_\_init\_\_ method
* Update the User class make\_deposit method
* Update the User class make\_withdrawal method
* Update the User class display\_user\_balance method
* SENSEI BONUS: Allow a user to have multiple accounts; update methods so the user has to specify which account they are withdrawing or depositing to

Bottom of Form