My name: Jaider Santiago Avila Robles - 20231020200

User Stories

- **As a** student passionate about video games
  I want to be able to choose between an Arcade machine or a Console
  So what I can buy the machine that best suits my preferences.

- **As a** player interested in the aesthetics of my machine
  I want to be able to select the material of the machine (wood, aluminum, or carbon fiber)
  So what I can have a personalized machine that looks good in my gaming space.

- **As a** user purchasing a video game machine
  I want to see a list of the available games for my machine
  So what I can choose the games I like the most before making the purchase.

- **As a** player who enjoys a wide variety of games
  I want to be able to add multiple games to the machine without having to restart the purchase process
  So what I can enhance the shopping experience and save time.

- **As a** user buying an arcade machine
  I want to see a summary with the details of the machine and selected games at the end of the purchase
  So what I can ensure the machine I'm buying includes everything I want before completing the payment.

- **As a** fan of retro video games
  I want to add retro games to my arcade video game machine
  So what I can customize the selection of games according to my preferences.

Object-Oriented Principles Analysis

- **Encapsulation:**
  The classes Machine, ArcadeMachine, and ConsoleMachine use encapsulation by keeping their attributes (such as material and added games) private. This prevents other classes from directly modifying the machine's internal state and forces the use of public methods (setMaterial, addGame, etc.) to interact with these attributes.

- **Inheritance:**
  The classes ArcadeMachine and ConsoleMachine inherit from the abstract class Machine. This allows sharing common machine logic (such as material selection and game management) while differentiating in specific behaviors (such as the machine's name).
- **Polymorphism:**
  The method getMachineName is an example of polymorphism. Being declared as abstract in Machine, each subclass (ArcadeMachine and ConsoleMachine) must provide its own implementation, ensuring that the machine's name adapts to the correct type without relying on conditional structures.
- **Abstraction:**
  The abstract class Machine encapsulates the general logic of an entertainment machine, such as material selection and game management. Subclasses are responsible for implementing specific details like the machine's name.

CRC Cards

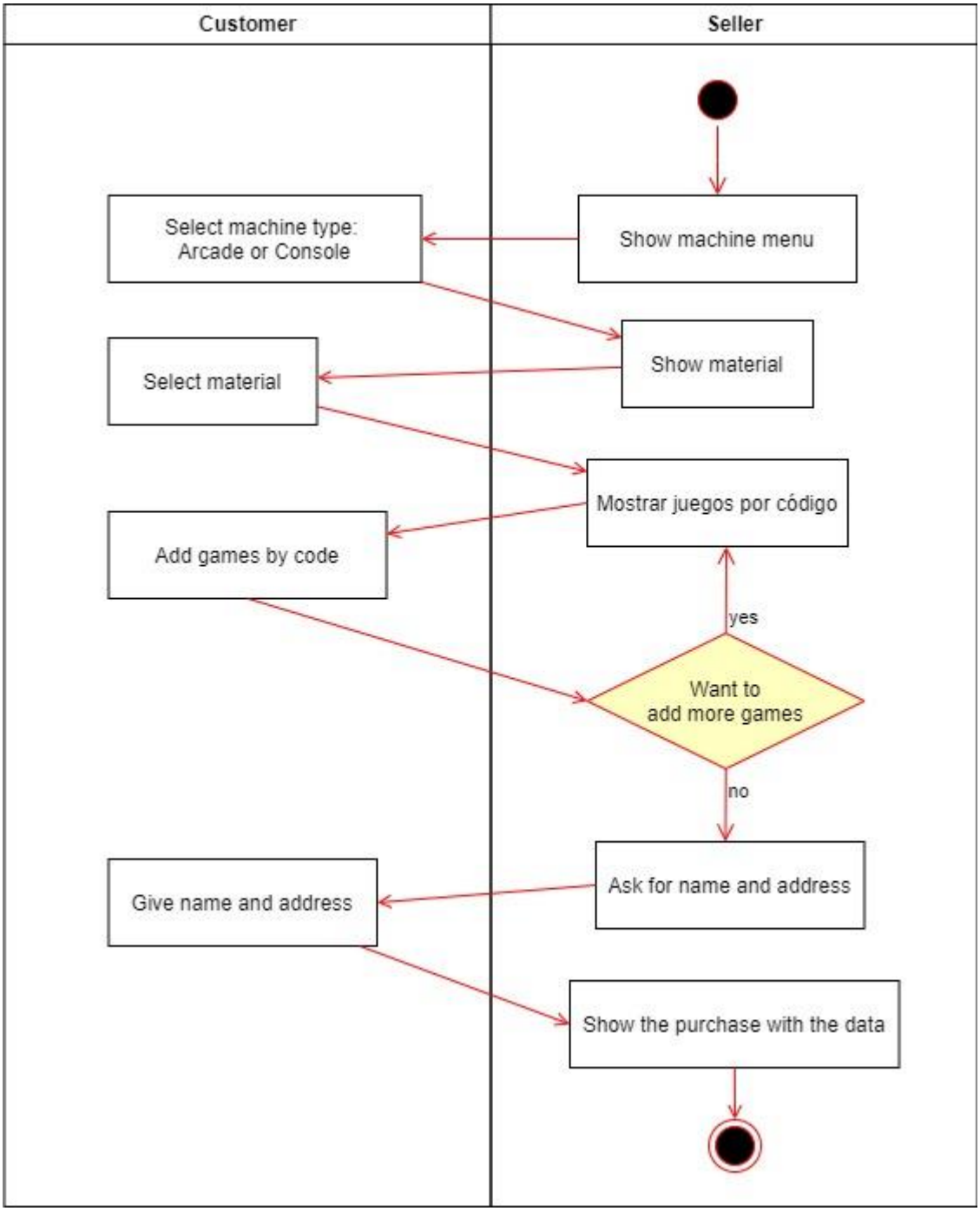
Machine	
Show available games Select material Add game by code	Game Arcade machine Console machine

Arcade Machine	
Say type of machine	Machine

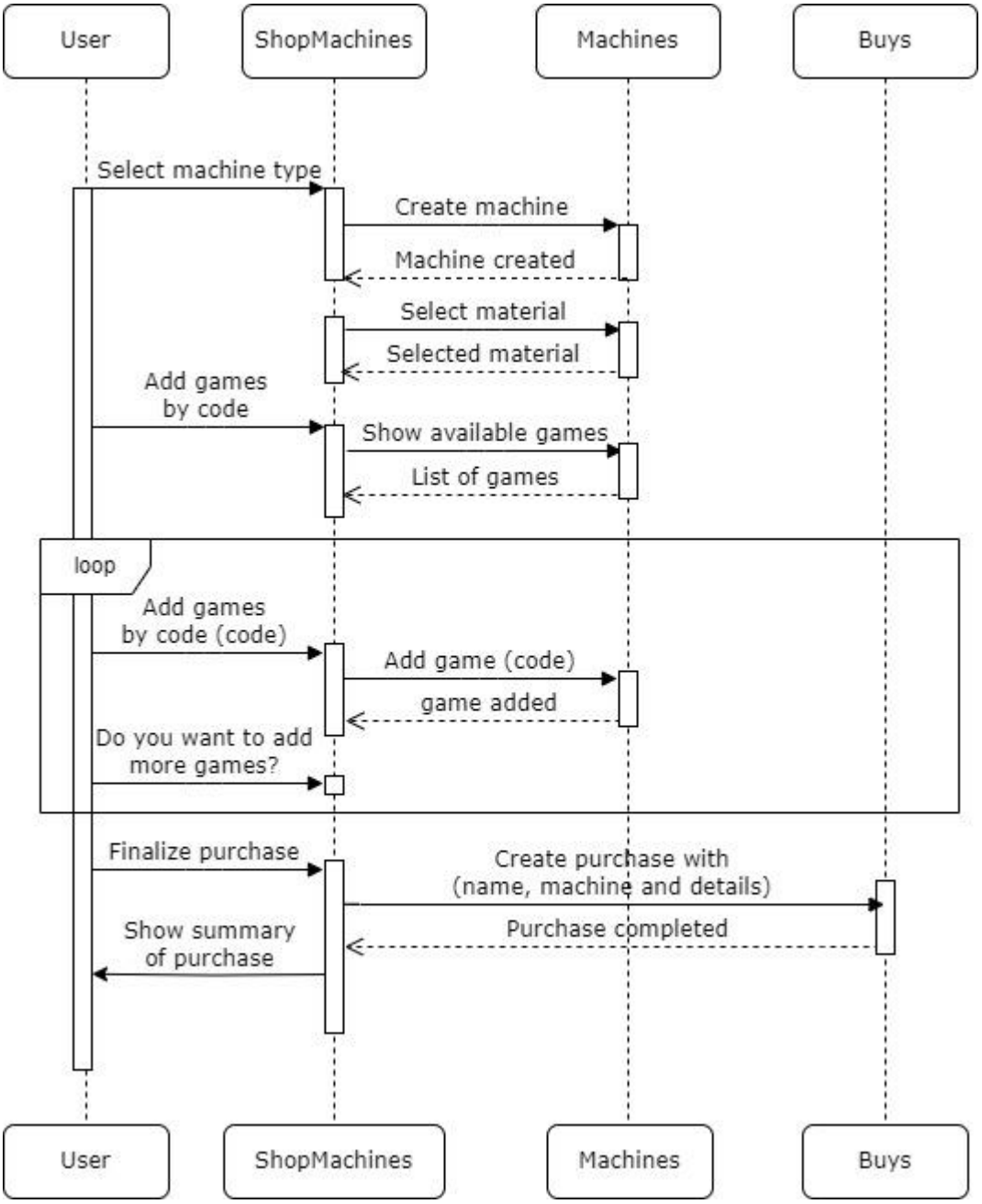
Console Machine	
Say type of machine	Machine

Buys	
Finalize the purchase Show purchase summary	Machine Game

Activity Diagrams



Sequence Diagrams:



Class Diagrams: