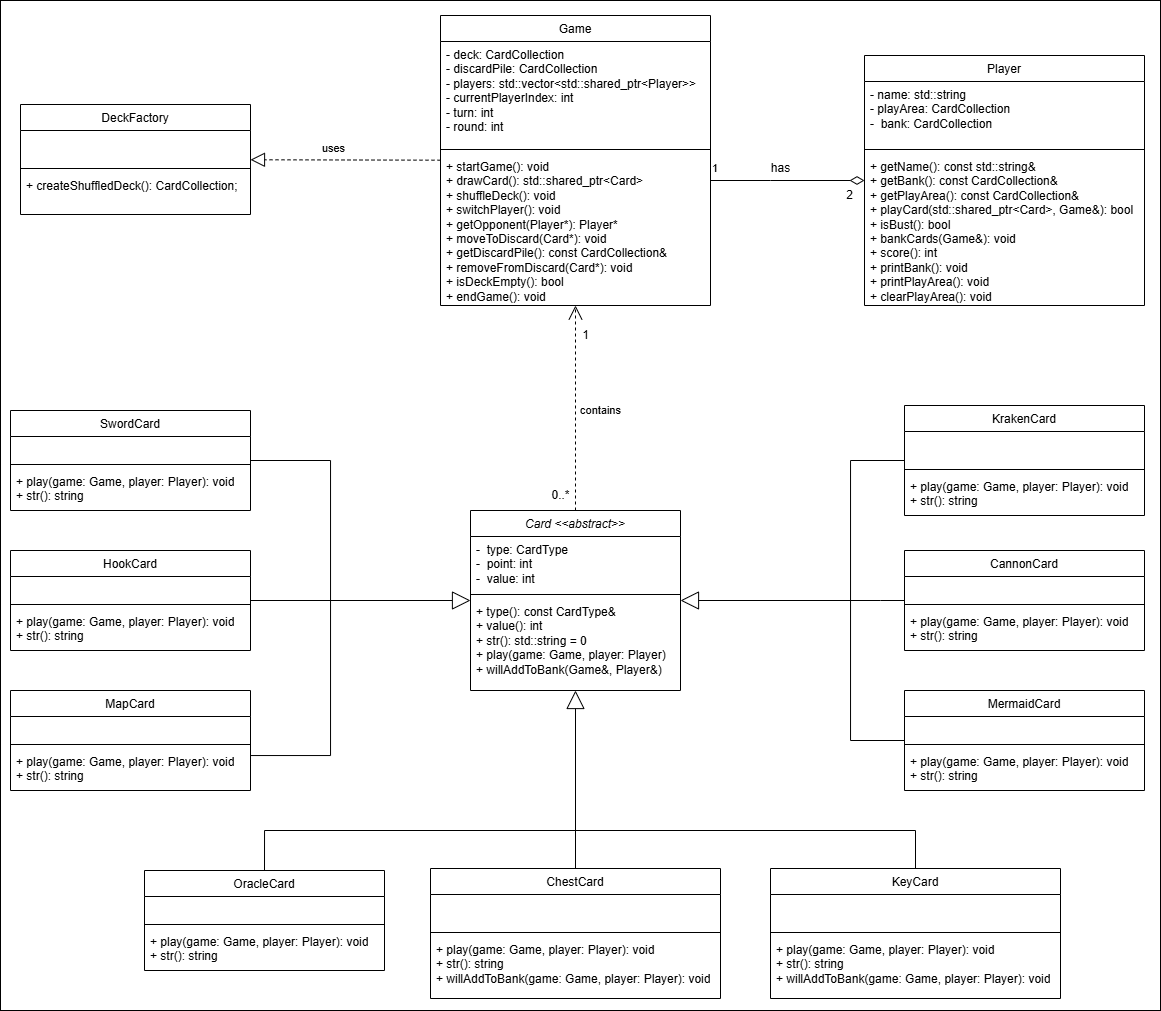
**Dead Man’s Draw++ – System Design**

**Name: Dev Patel**

**Student ID:**

**Email: patdy074@mymail.unisa.edu.au   
  
  
Here is the UML diagram: -**



**Design Decision made while building the system: -**

* All cards inherit from an abstract Card base class to follow the Open/Closed Principle.
* The Card class includes a pure virtual function play() to ensure each subclass provides its specific effect.
* Common attributes (type, point, value) and behaviours are centralized in the base class.
* Created a DeckFactory class to follow Single Responsibility Principle by handling only deck creation and shuffling.
* Game has composition relationships with Player, deck, and discardPile for clearer ownership.
* Each Player object manages their bank and playArea using CardCollection.
* Used std::shared\_ptr<Card> consistently for safe memory handling of dynamically created cards.
* Defined CardCollection as a type of alias for std::vector<std::shared\_ptr<Card>> to improve readability and maintainability.
* Applied Factory Pattern in DeckFactory
* Applied Polymorphism through card behaviour overrides
* Applied Encapsulation in all class designs
* Complied with SOLID principles throughout (documented via SOLID checklist)