Name: Krishna Varshith Pabbisetty
SJSU ID: 017416477

CMPE-202: Individual Project Documentation

Github Repository: https://github.com/gopinathsjsu/individual-project-VarshithPabb1setty

Primary Problem:

- The primary problem your project solves is parsing and processing credit card records from different file formats (CSV, JSON, XML). Each record includes a credit card number, expiration date, and cardholder's name. The main challenge is to read these records, validate the credit card number, determine the card issuer, and create an instance of the appropriate credit card class.

Secondary Problems:

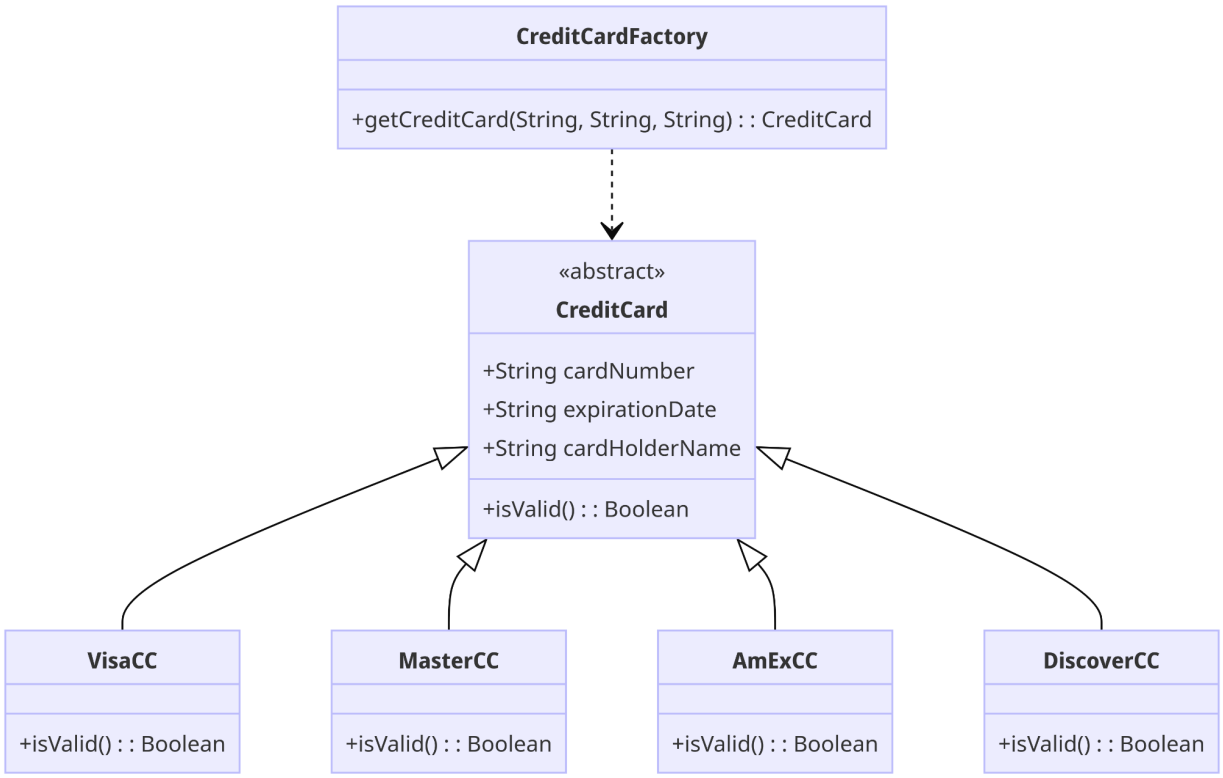
- **File Format Flexibility:** The system needs to handle various input file formats (CSV, JSON, XML) and potentially accommodate new formats in the future.
- **Credit Card Validation:** Each credit card number must be validated to check if it's a legitimate number and identify the card issuer (Visa, MasterCard, AmericanExpress, Discover).

Design Patterns Used:

- **Factory Pattern:**
  - Used in `CreditCardFactory` to create instances of different credit card types based on the card number.
  - This pattern allows easy extension to support new credit card types in the future.
- **Strategy Pattern:**
  - Implemented through the `FileParser` interface with different strategies for parsing files (`CSVParser`, `JSONParser`, etc.).
  - This pattern provides flexibility to add new file parsing strategies without modifying the existing code.
- **Utility Class:**
  - `CardValidationUtil` for shared validation logic, enhancing code reuse and separation of concerns.

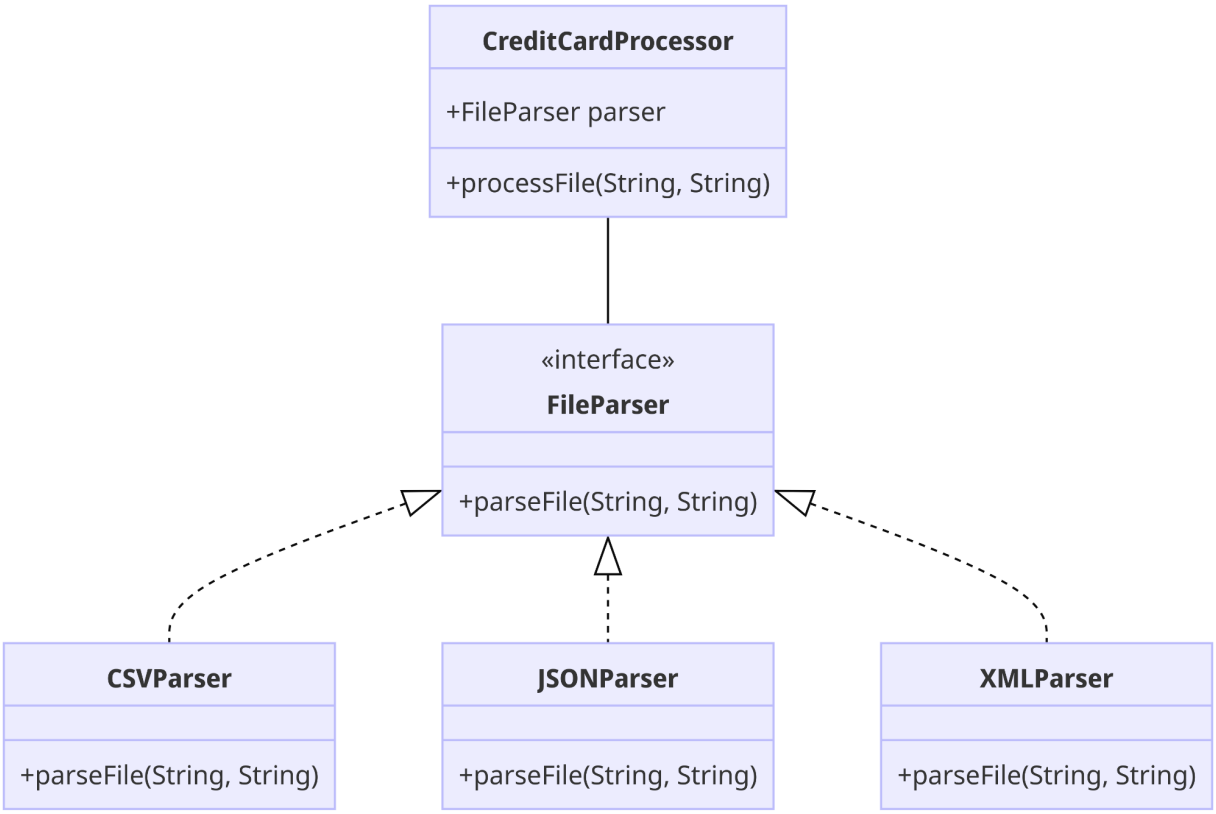
Name: Krishna Varshith Pabbisetty
SJSU ID: 017416477

UML Class Diagram for Factory Pattern Implementation:



Name: Krishna Varshith Pabbisetty
SJSU ID: 017416477

UML Class Diagram for Strategy Pattern Implementation:

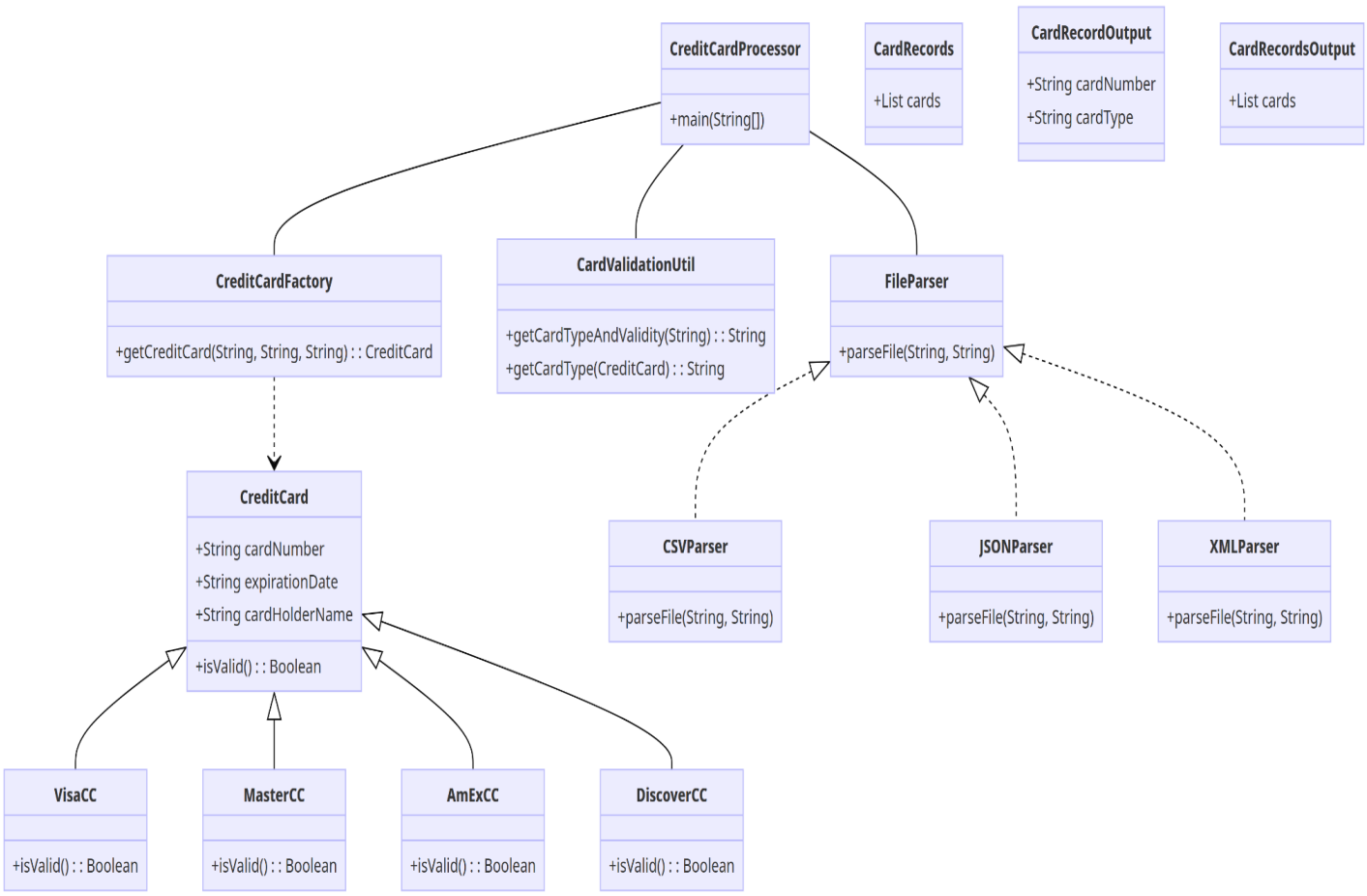


Consequences of Using These Patterns:

- **Factory Pattern:**
  - Pros: Simplifies the creation of credit card objects, centralizes the creation logic, and makes the system more maintainable and extensible.
  - Cons: The factory class can become a single point of failure and may need modification when adding new subclasses.
- **Strategy Pattern:**
  - Pros: Enhances the flexibility and scalability of the file parsing process. New parsing strategies can be added without altering existing code, adhering to the Open/Closed Principle.
  - Cons: Can introduce additional complexity with multiple strategy classes. Requires proper handling of the context where strategies are switched.

Name: Krishna Varshith Pabbisetty
SJSU ID: 017416477

UML Class Diagram:



Name: Krishna Varshith Pabbisetty
SJSU ID: 017416477

UML Sequence Diagram: