RIPHAH INTERNATIONAL UNIVERSITY

DATA STRUCTURES
Project Proposal

Card Battle Game

Supervisor Name: Ma’am Saleha Nasim

Name of Students	**Sap ID**	**Program**
Eman Aleem	62395	BSCS-3
Arfa Zain	64255	BSCS-3
Sundas Ambreen	58822	BSCS-4

Date of Submission
03/11/2025

Contents

PROJECT PROPOSAL 2

1. Project Title

“Card Battle Game”, A Turn-Based Strategy Game Using Data Structures in C++.

2. Overview

The Card Battle Game is a console-based, turn-based strategy game developed in C++. It demonstrates the practical implementation of Data Structures and Algorithms (DSA) along with Object-Oriented Programming (OOP) principles. Each player has a deck of cards, where every card contains a name and a power value. Players take turns drawing cards from their decks, and the player with the higher power value wins that round.

3. Algorithm

1. **Initialize the game system and player profiles.**
2. **Create card objects** using the Card class, assigning each a name and power attribute.
3. **Store all card objects** in a central vector container.
4. **Shuffle the card collection** and **distribute** the cards equally between the two players.
5. **Populate each player’s deck** (represented as a **Queue**) with the distributed cards.
6. **Commence the gameplay loop**: each player **draws** the top card from their deck.
7. **Compare the power values** of the drawn cards.
8. **Determine the winner of the round** based on the higher power value.
9. **Transfer the winning card** (and optionally the losing card, depending on rules) to the winner’s **discard pile** (represented as a **Stack**).
10. **Repeat steps 6–9** until both decks are depleted.
11. **Calculate the final scores** by counting the number of cards in each player’s discard pile.
12. **Announce the overall winner** based on the final card count.
13. **Conclude the game** and present results.

4. List of Operations

- **enqueue():** Add a new card to the player’s deck (queue).
- **dequeue():** Draw a card from the top of the deck.
- **push():** Push the winning card to the discard pile (stack).
- **pop():** Remove a card from the discard pile if needed.
- **compareCards():** Compare two cards’ power values.
- **displayScore():** Show each player’s score and results.
- **generateDeck():** Randomly create a set of cards for each player.

5. Game Logic and Players

- The Card Battle Game is designed for two players.
- Both players have their own decks of cards.
- In each round, they draw the top card from their deck and compare their power values.
- The card with the higher power wins the round, and that card is added to the discard pile.
- This two-player setup is simple and easy to manage, which makes it perfect for beginner-level projects.
- It also clearly shows how queues, stacks, and vectors work together in a real game situation.

6. Data Structures Used

Data Structure	**Purpose**	**C++ Implementation**
Queue	To store each player’s deck (FIFO).	std::queue<Card>
Stack	To manage discarded cards after each round.	std::stack<Card>
Vector	To generate and hold the initial set of cards.	std::vector<Card>
Class (OOP)	To define the Card object with attributes (name, power).	class Card

7. Where to Use Each Data Structure

Data Structure	**Where Used in Game**	**Example**
Queue	Players’ decks cards are drawn in the same order they were dealt.	playerDeck.push(card);
Stack	Discard pile stores winning cards from each round.	discardPile.push(winningCard);
Vector	Used during deck generation for random card creation.	vector<Card> allCards;
Class (OOP)	Represents card properties and player structure.	class Card { string name; int power; };

8. Conclusion

The Card Battle Game effectively demonstrates the use of data structures such as queues, stacks, and vectors along with OOP concepts in C++. It provides a fun and educational way to understand the application of algorithms and DSA principles in a real-world inspired mini project.