

Project: **Pazooock – System Requirements Specification**

Author: **Ivan Belov @ ivan.belov@gmx.com**

Date: **2019**

Table of contents

- ❖ Project Description
- ❖ Game Rules
- ❖ Domain Dictionary
- ❖ Functional Requirements
- ❖ Non – Functional Requirements
- ❖ Hardware Requirements
- ❖ Software Requirements
- ❖ Release Plan
- ❖ Training Plan
- ❖ Interface design
- ❖ Application's Classes
 - Candidate Classes and their Relationships
 - Class Interactions
 - Card Class
 - Card's Design
 - Deck Class
 - Side Deck Class
 - Hand Class
 - Player Class
 - Game Class
 - Match Class
 - Lane Class
 - Game Engine Class
- ❖ Software Architecture
- ❖ Design Constraints
- ❖ Main Activity
- ❖ Use Case and Sequence Diagrams
 - Create New Profile Use Case
 - Create New Profile Sequence Diagram
 - Select Profile Use Case
 - Select Profile Sequence Diagram
 - Delete Profile Use Case
 - Delete Profile Sequence Diagram
 - Play Single Player Game Use Case
 - Play Single Player Game Sequence Diagram
 - Play Cooperative Game Use Case
 - Play Cooperative Game Sequence Diagram
 - Record game Score Use Case
 - Record Game Score Sequence Diagram

Project Description

Pazack is the Star Wars Universe card game. Pazook is a personal representation of the intellectual entertainment product mentioned above. System Requirements Specification document describes version 1.0. Pazook is a single-player computer game that simulates a card game match between a computer and a human player. This document has Bell MT font of size 14 throughout. Title names and table of contents items are bolded for visibility. Wingdings 0x0076 type of a bullet used for listing details in the document. The intended audience of this paper is a future development team, marketing staff, and investors. The suggested sequence for reading this document is to go through the table of contents in sequential order. The purpose of this program is to provide entertainment to its users. Assumed factors such as the presence of the market or monetary support for the project could affect its requirements or development.

Game Rules

The rules of the game are as follows:

There shall be two opponents in this game.

Each of the opponents to have one single basic game deck of eighteen cards. There are three groups of cards: Positive – represented with the sign of plus - '+,' Negative – the symbol of minus - '-', Combined Group – positive and negative signs together - '+/-.' The cards in all three groups range from one to six.

Extra cards could be added to the deck by purchasing or winning them though out the gameplay — additional cards to extend decks with ranging values from seven to ten.

Each card shall have a sign, a value, a unique face representation and a custom deck suite. Any card could be present in the replicated state. Multiple copies of any card could occur in one deck, side deck, and a hand.

Each opponent has to select a side deck consisting of ten cards before each match. Side deck selection could be performed manually or randomly – automatic. After a side deck selection, a random four cards are selected as a hand of each opponent.

The game goes as many rounds as needed to determine the winner of three rounds. Each round, both opponents are trying to bring their own combined pot value as close to twenty-one as possible.

Each round consists of turns. Turns are taken by opponents one after another. Each turn, a single card is drawn from a third basic shuffled deck and placed in a player's pot. After an automatic card placement by a dealer, a player has a chance to perform an action. A player during his / her turn can select any number of cards from their hand to be added to own pot, can hold or remain as is, can withdraw from a game.

A player who wins the most rounds out of three is a winner; tie round or games are not taken into consideration, but still recorded.

Application's Classes

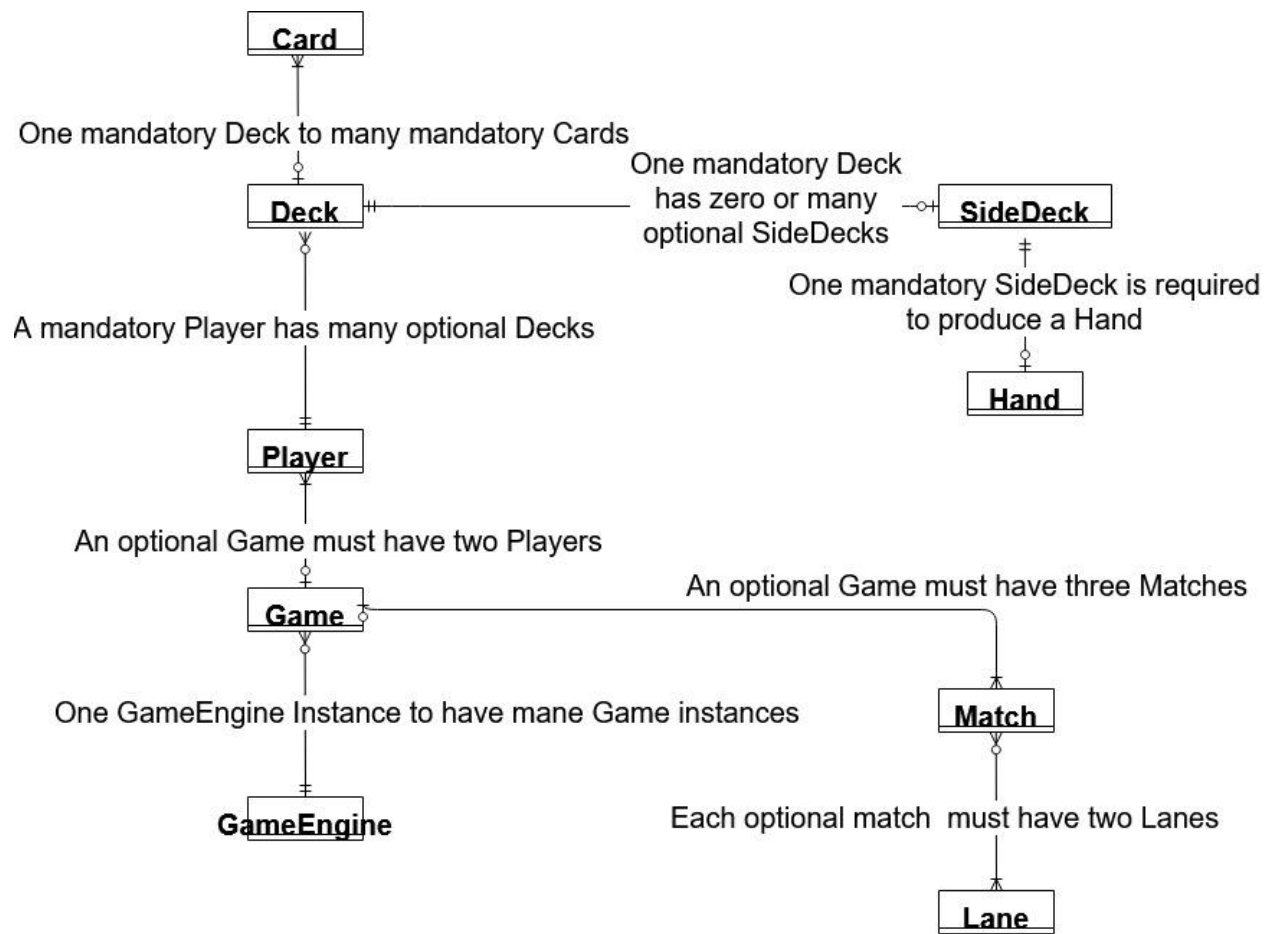
Candidate Classes:

- ❖ **Card**
- ❖ **Deck**
- ❖ **Side Deck**
- ❖ **Hand**
- ❖ **Player**
- ❖ **Game**
- ❖ **Match**
- ❖ **Lane**
- ❖ **Game Engine**

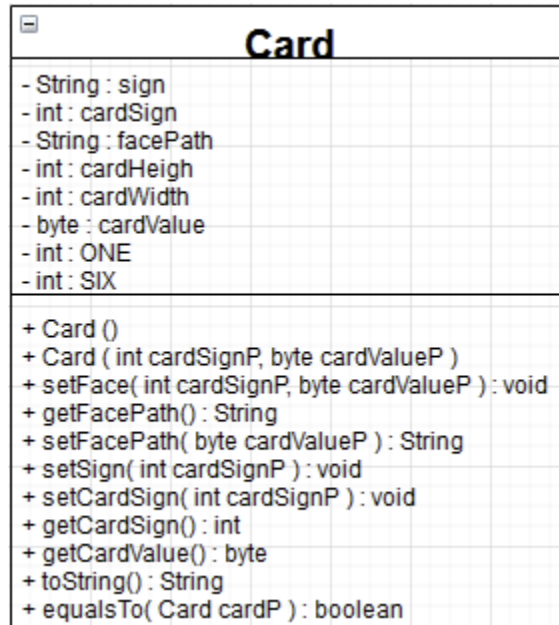
List of Relationships:

- ❖ **A Deck has eighteen or more Cards.**
- ❖ **A Player could have many Decks.**
- ❖ **A Side Deck consists of ten Cards selected randomly from a Deck.**
- ❖ **A Hand consists of four Cards selected randomly from a Side Deck.**
- ❖ **A Game consists of three Matches and must include two Players.**
- ❖ **A Match has two distinct Players.**
- ❖ **A Match much have two Lanes.**

Class Interactions

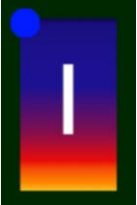

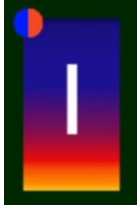

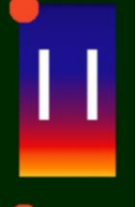
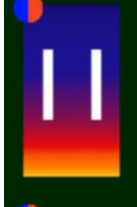








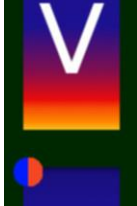








Card Class

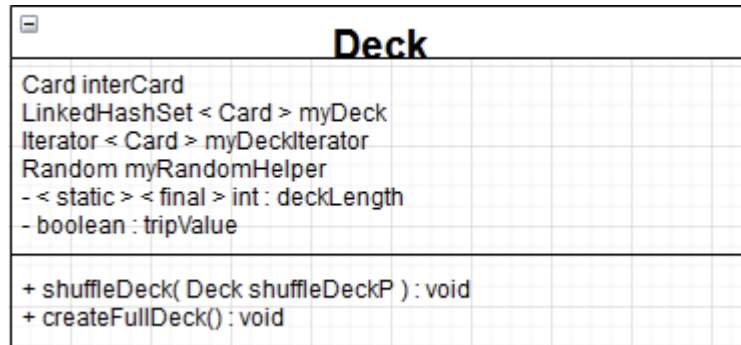


A Card is a class representing a card entity. A card is one of the **three** in-game **suits**: positive, negative, and combo. A card can be of one of **six ranks**. These ranks include the roman numerals from one through six. Wild cards might be implemented to be purchased with in-game currency. Each card has a face and back. A card's face describes a suit and a rank.

Cards' Design

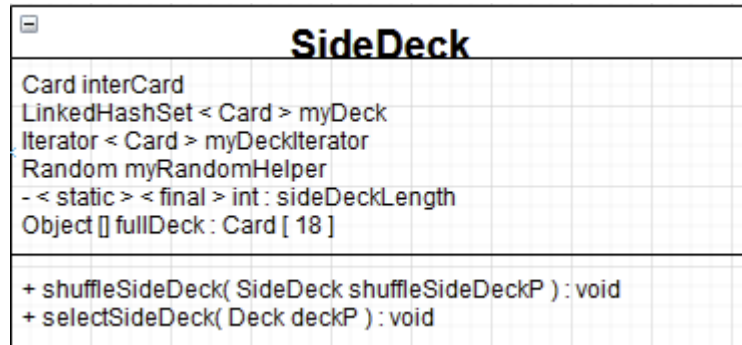
	Negative Suit	Positive Suit	Combo Suit
Rank One			
Rank Two			
Rank Three			
Rank Four			
Rank Five			
Rank Six			
	Player	Opponent	Card's Spot
Default Back Design			

Deck Class



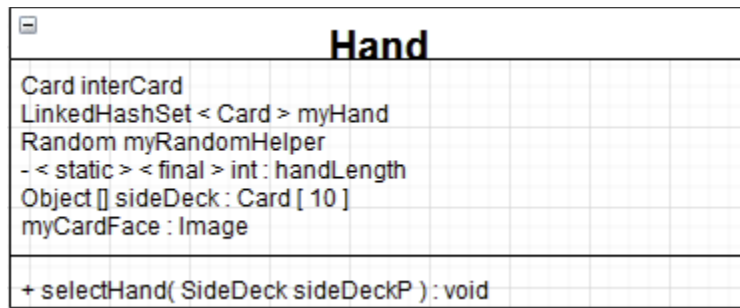
A Deck is a representation of a deck of playing cards. There are eighteen cards in each standard deck. A standard deck of cards is required for a player to play a game. The back of the deck of cards can have custom graphics set by a user. Ultimately a gaming deck could consist of a standard deck and a deck of random wild cards. A user could have multiple card decks.

Side Deck Class



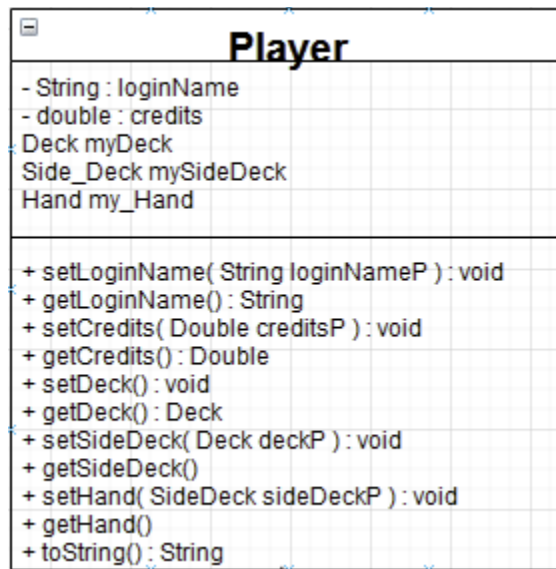
A Side Deck is a representation of a randomized, shuffled and shortened deck of cards. Ten cards shall be selected by a user or randomly by computer logic. Side Deck has ten cards.

Hand Class



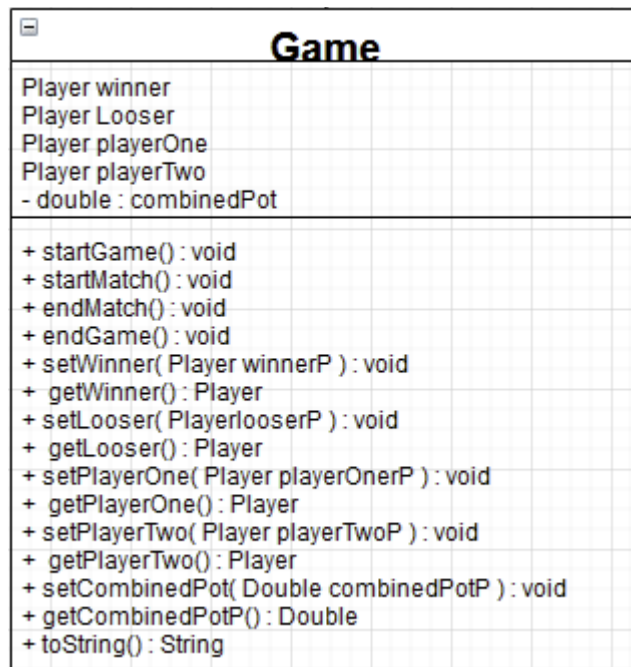
A Hand class represents a set of cards. There are four cards in hand. Those cards are selected randomly for a side deck before each match.

Player Class



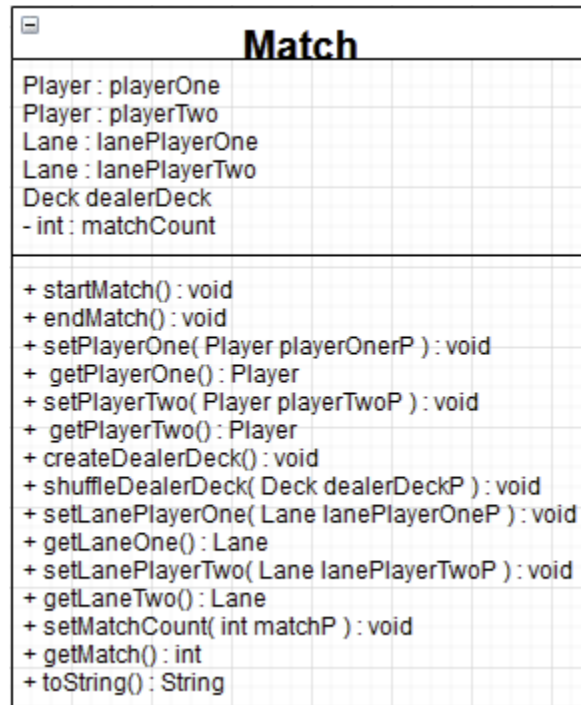
Player class hold variables and features that describe a game user in detail. The player is also an account holder for a network game. A player has login credentials, credits that he or she earned up to date, a deck of cards.

Game Class



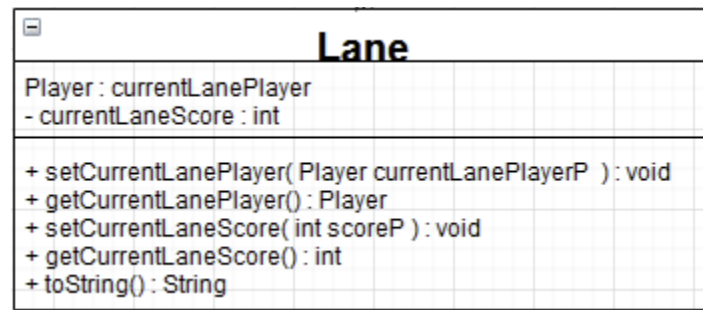
A Game is an instance of a Pazook game. It brings two players with two decks together for few matches to determine a winning party. A game shall have a pot of credits that players are betting their win.

Match Class



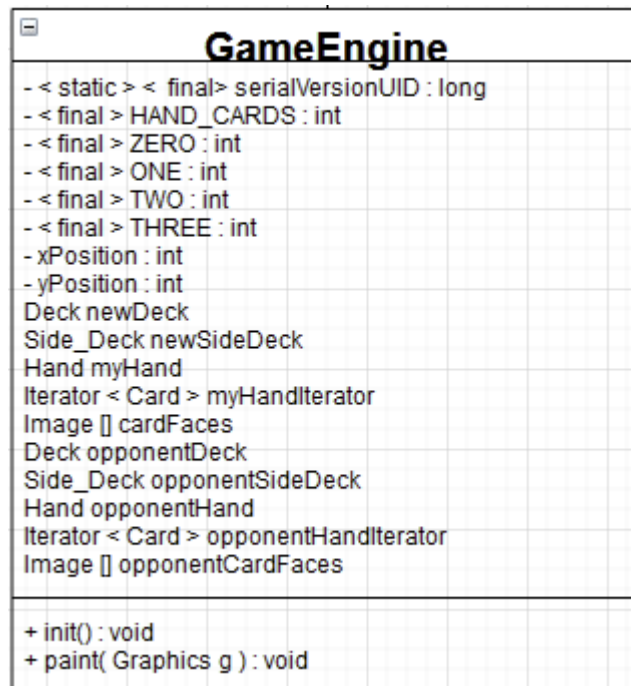
A Match is the inner work of a game. There are three matches in each game. The winner of the most matches is the winner of that game. Overall bet pot is awarded to a winning player. The statistics of each match and a game is recorded locally or remotely.

Lane Class



A Lane is a representation of a set of nine cards. This set consists of cards being dealt by a dealer, in addition to cards selected from a player's hand.

Game Engine Class



A Game Engine class provides Graphical User Interface and the logic behind the application flow.

Functional Requirements

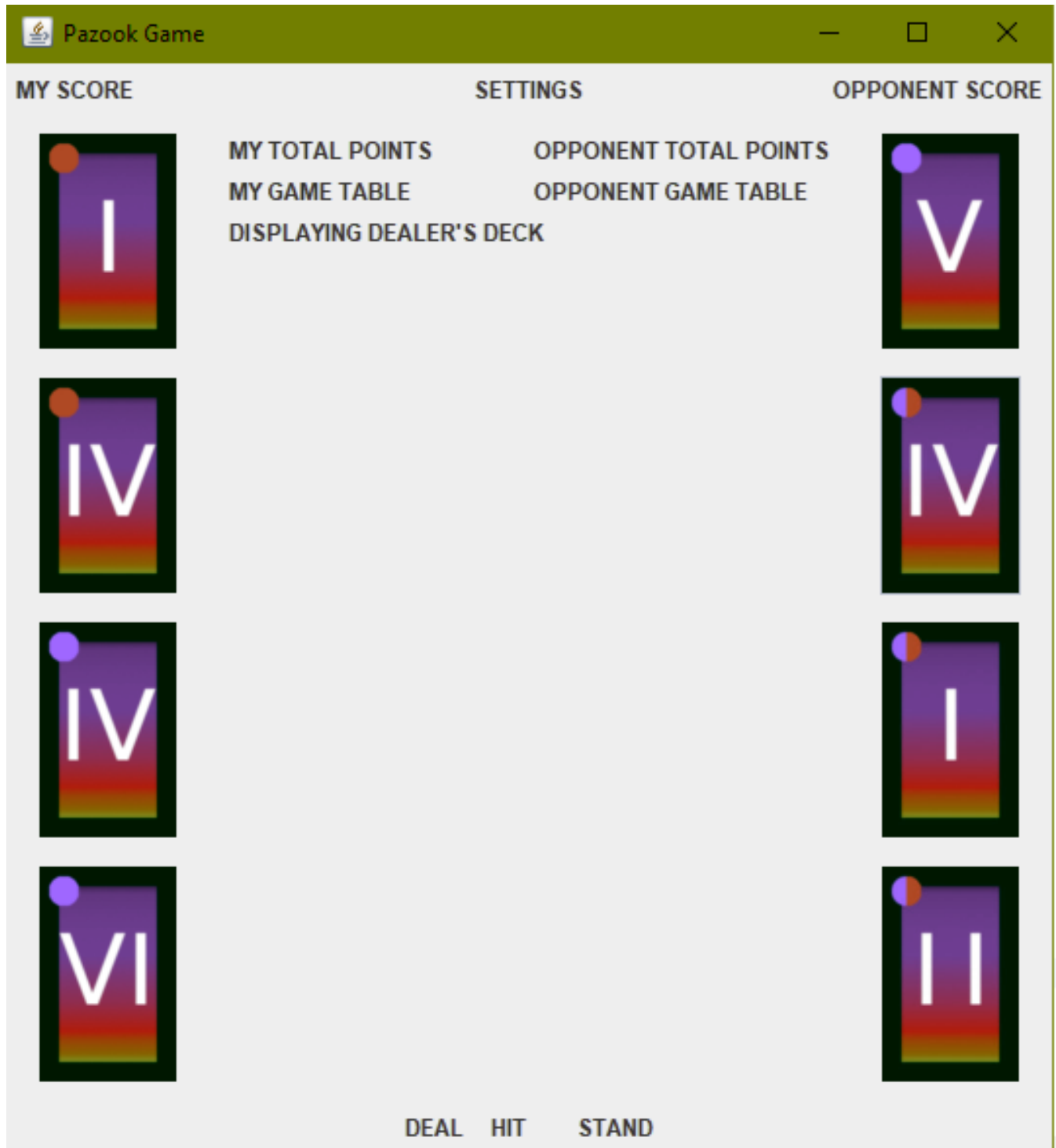
Code:	Description:
FR-1	Full-Screen ability shall be implemented.
FR-2	Display the user's hand (Four Cards).
FR-3	Display the opponent's hand suite up (Four Cards).
FR-4	Display the change of hand's cards after each turn.
FR-5	The application shall display the changing score of the current hand for both opponents and an overall game.
FR-6	The general deck shall be allowed to run out.
FR-7	A user shall be able to see her / his previous game scores.
FR-8	This application shall let users choose between a random side deck and a custom one (Ten Cards).
FR-9	The Start Page shall have Options, Rules, New Game, Profile, Scores History.
FR-10	The options tab shall have background sound on / off.
FR-11	There shall be a local database to keep player records and scores.
FR-12	There shall be a drop off folder for .mp3 files to play your music tracks during gameplay.
FR-13	The application shall suggest using a previously used side deck or let user save side deck.
FR-14	Each opponent shall be able to win, lose or have a tie at the end of each match or a game.
FR-15	The combined value of the player's pot shall have an amount of under twenty-one, twenty-one, and over twenty-one.
FR-16	A turn shall be able to be completed or held.
FR-17	Each opponent shall be able to withdraw from a game at any point in its progress, but only when it is that opponent's turn.
FR-18	A player shall be able to set a bet in in-game currency on his / her win at the beginning of each game.
FR-19	A Main Menu window shall be interactive.

Non – Functional Requirements

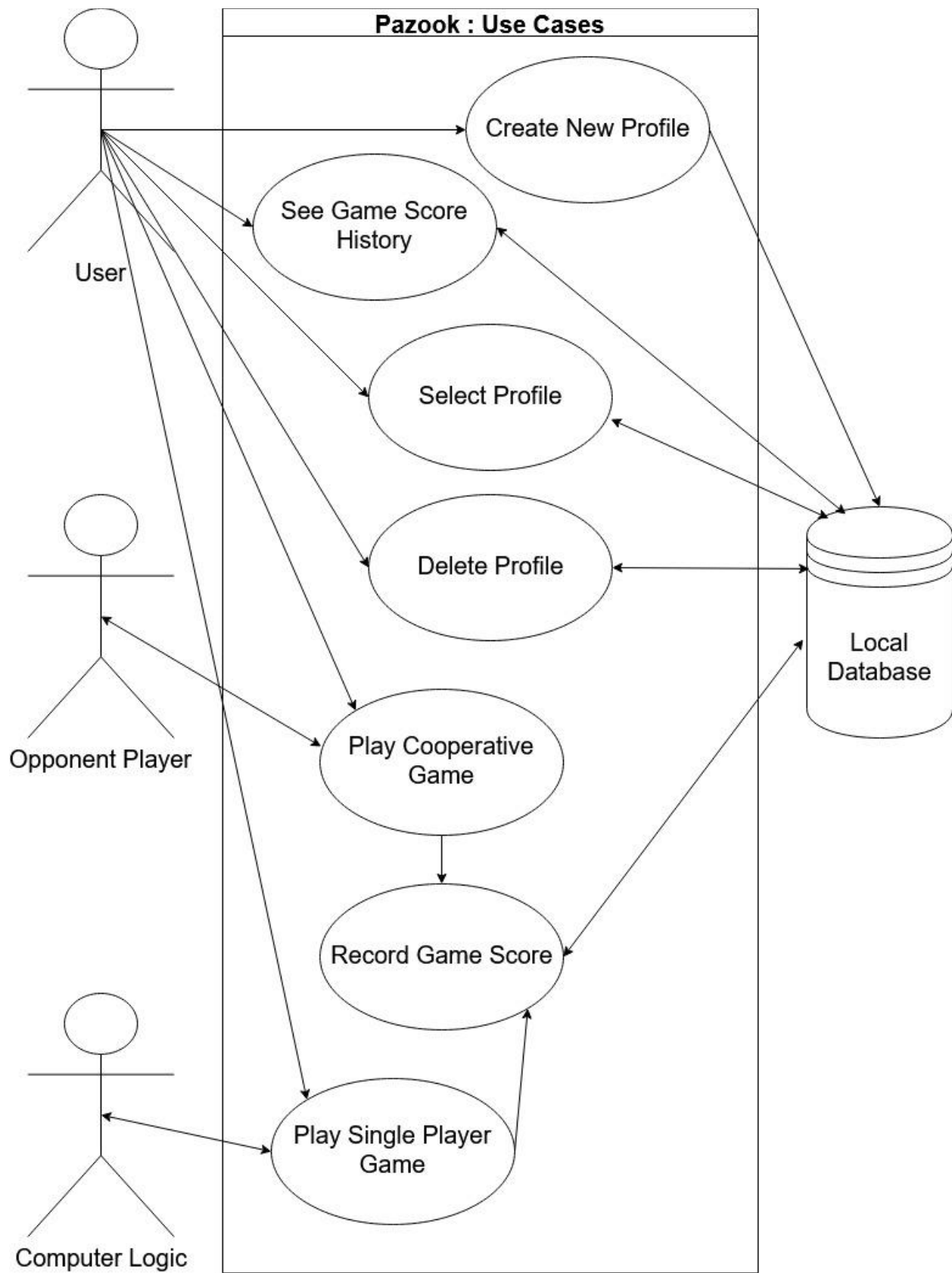
Code:	Description:
NFR-1	This application shall have a unified graphical interface.
NFR-2	Face of any card and a suite of each opponents' deck shall be displayed and available for view during the gameplay.
NFR-3	User account passwords shall be a minimum 8 characters in length and must contain a lowercase letter, an uppercase letter, a number, and a symbol.
NFR-4	Users should be able create their own deck suits.
NFR-5	The payment system shall be available 24 hours a day, 7 days a week, and shall have a 99.5% uptime.
NFR-6	The startup time for the application shall not exceed 5 seconds.
NFR-7	Monetary amounts shall be accurate to 2 decimal places, rounding where necessary.
NFR-8	user shall be able to top up in game currency with own money at any time with any amount.
NFR-9	The application shall be simple and intuitive such that a new user with no training shall be able to use the service immediately.
NFR-10	The system shall be scalable to support unlimited growth in the number of concurrent users.
NFR-11	A user shall be able to select a custom design of the back of player's deck.
NFR-12	A user shall accept terms of services during the installation of the Pazook.

Interface Design

❖ Sample Graphical User Interface



Use Case Diagram



Create New Profile Use Case

Name:	Create New Profile
Description:	
Primary Actors:	
Secondary Actors:	
Preconditions:	
Main Flow:	
Postconditions:	

Create New Profile Sequence Diagram

Select Profile Use Case

Name:	Select Profile
Description:	
Primary Actors:	
Secondary Actors:	
Preconditions:	
Main Flow:	
Postconditions:	

Select Profile Sequence Diagram

Delete Profile Use Case

Name:	Delete Profile
Description:	
Primary Actors:	
Secondary Actors:	
Preconditions:	
Main Flow:	
Postconditions:	

Delete Profile Sequence Diagram

Play Single Player Game Use Case

Name:	Play Single Player Game
Description:	
Primary Actors:	
Secondary Actors:	
Preconditions:	
Main Flow:	
Postconditions:	

Play Single Player Game Sequence Diagram

Play Cooperative Game L.A.N. Use Case

Name:	Play Cooperative L.A.N. Game
Description:	
Primary Actors:	
Secondary Actors:	
Preconditions:	
Main Flow:	
Postconditions:	

Play Cooperative Game L.A.N. Sequence Diagram

Play Cooperative Game W.A.N. Use Case

Name:	Play Cooperative W.A.N. Game
Description:	
Primary Actors:	
Secondary Actors:	
Preconditions:	
Main Flow:	
Postconditions:	

Play Cooperative Game W.A.N. Sequence Diagram

Record Game Score Use Case

Name:	Record Game Score
Description:	
Primary Actors:	
Secondary Actors:	
Preconditions:	
Main Flow:	
Postconditions:	

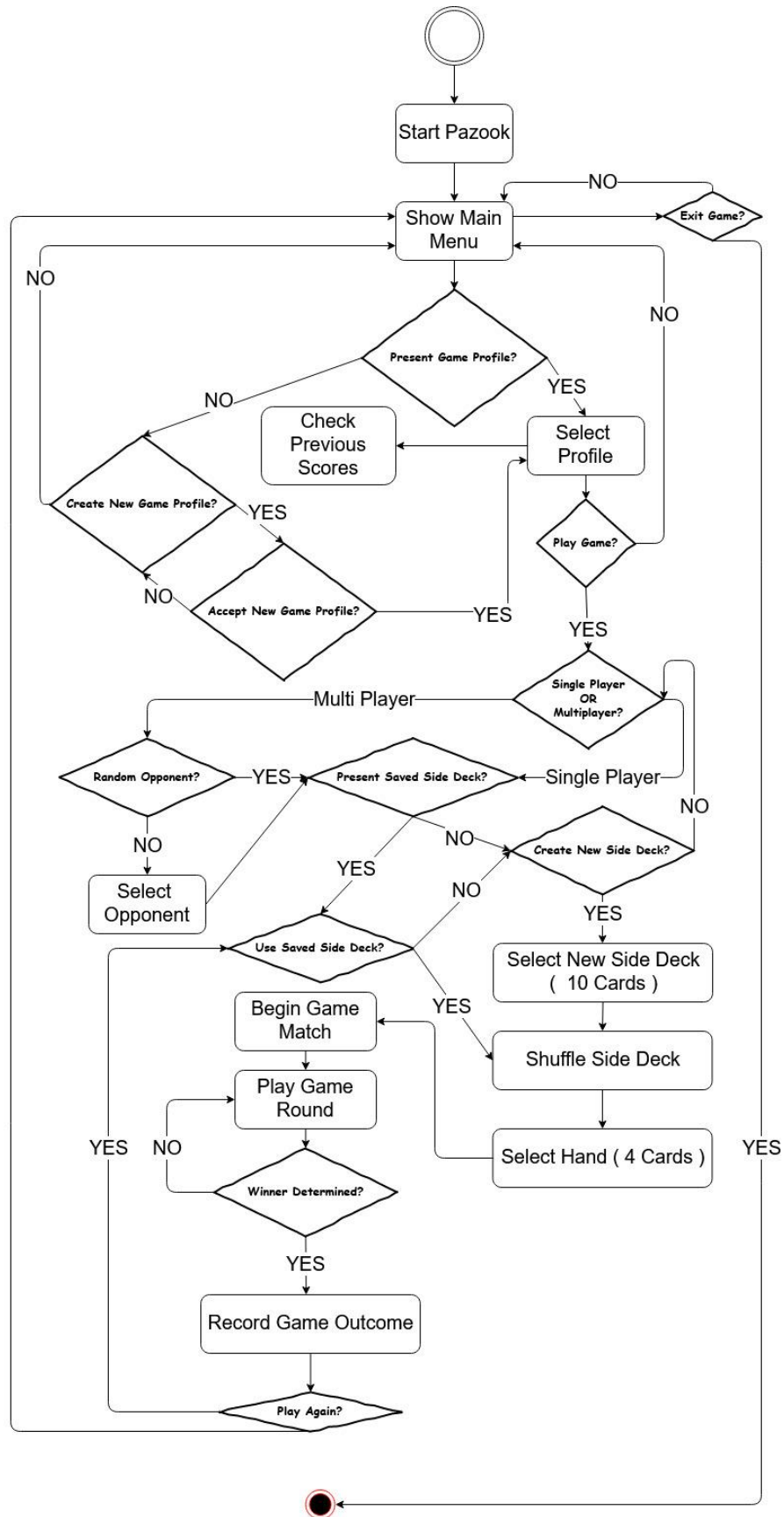
Record Game Score Sequence Diagram

See Game Score Use Case

Name:	See Game Score Player Game
Description:	
Primary Actors:	
Secondary Actors:	
Preconditions:	
Main Flow:	
Postconditions:	

See Game Score Sequence Diagram

Main Activity



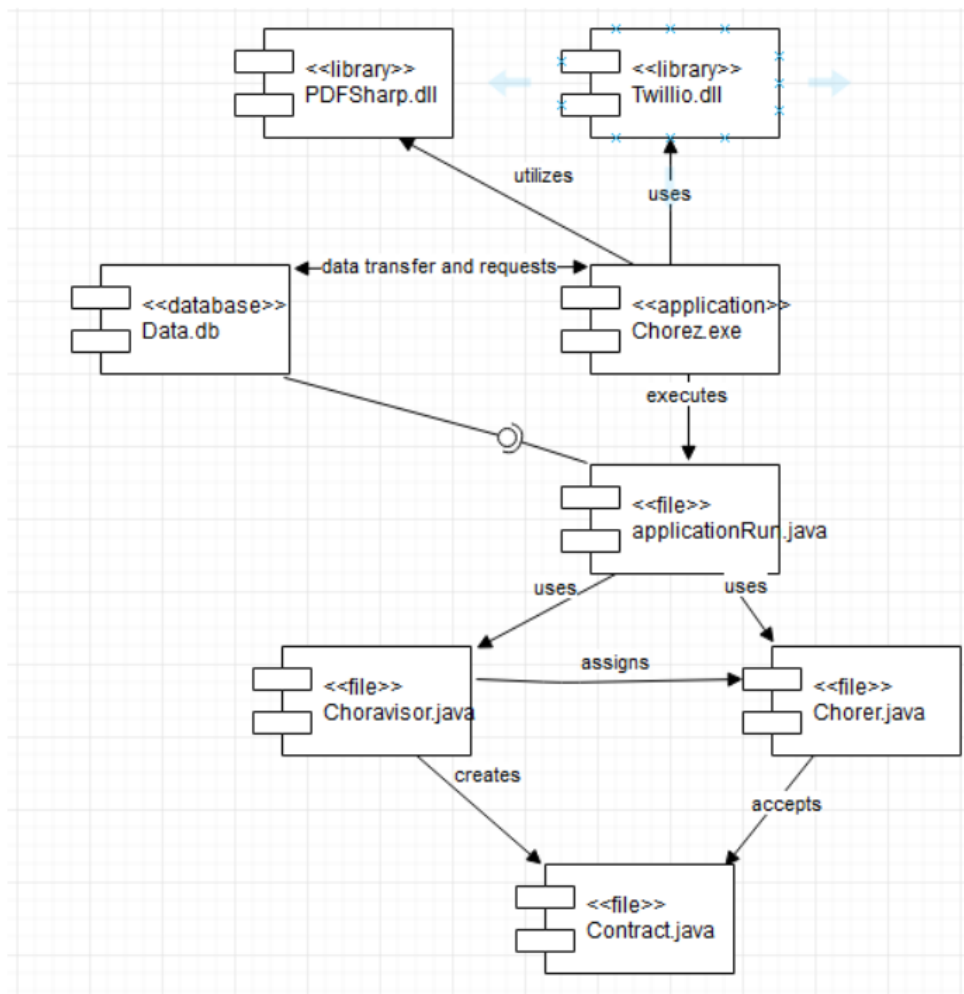
Software Architecture

The top-down approach as an architectural style and an iterative development model are used in the research and development phase.

The software is a three-tier business application to be accessed from Windows, Android or Mac operating systems. Locally and remotely stored databases are used for data storage. Microsoft Azure can host the application's logical layer. A printer is required to print transaction statements but not needed for the Chorez - Biz performance or functional operation.

Component diagram below describes modules of the software.

Component diagram:



Release Plan

Version:	Description:	Release Date:
Pazook_v10.exe	This version features an ability to play a game of Pazook with computer logic and save data locally and securely. Update of SRS Documentation.	December 2020
Pazook_v20.exe	This version features an ability to play a game of Pazook with another user over a secure connection on a local network. Update of SRS Documentation if required.	December 2021
Pazook_v30.exe	This version provides an ability to play a game of Pazook with another player over a secure connection on a remote game server.	December 2022

Training Plan

Needs and Skills Analysis:	<p>Anticipated users are private citizens - contract initiators, and private, small businesses owners- contractors, as well as those with necessary computer skills: data entry clerk, and other various clerical staff members.</p> <p>Tasks to be taught are:</p> <ul style="list-style-type: none">❖ originate contract❖ manage contract❖ select a contractor❖ track work progress❖ verify completed contract❖ pay contractor❖ leave contractor review <p>Skills to be learned:</p> <ul style="list-style-type: none">❖ Data input utilizing computer input devices❖ Basic troubleshooting and ability to find help or identify an issue
Development Approach:	<p>A prototype approach is to be used when creating a recorded set of correct instructions needed for the execution of basic application tasks. This development work is dependent on constant end-user feedback to provide a high retention training product.</p>
Training Curriculum:	<p>A document describing a how-to of a feature is to be developed as soon as a given software part is implemented and published. The questionnaires and feedback pop-up windows are provided based on user needs and requests to facilitate the user-developer communication.</p>

Design Constraint

Constraint:	Type:	Description:
Commercial	Self - Imposed	Time and budget
Legal Compliance	Imposed	Applicable laws, regulations, standard
Design	Self - Imposed	Identical Graphical User Interface features, predetermined syntax format and design pattern.
Localization	Self – Imposed	Providing an ability to switch or change User Interface or modules based on cultural or geographical differences
Integration	Self - Imposed	Usage of various modules, systems, and products with Chorez – Biz conjunction.
Security	Self - Imposed	Production of streamlined and effective database queries to reduce resource usage and prevent data leaks. Database duplication for redundant local and hosted data storage. Encryption to mask any remote or internal communication.
Ease of Access	Self - Imposed	Screen optimization, clean, consistent and simple User Interface.

Domain Dictionary

Term:	Type:	Description:
Card	Software entity	
Deck	Software entity	
Side Deck	Software entity	
Hand		

Hardware Requirements

- ❖ Dedicated desktop Windows machine with support of .Net - 4.0
- ❖ CPU 1.4 GHz 32-bit (x86)
- ❖ RAM 4 GB
- ❖ 20 GB local hard disc space
- ❖ Broadband Internet connection if remote data storage and messaging modules are utilized
- ❖ Input and pointer devices such as keyboard and mouse
- ❖ Output device such as VGA or HDMI supported monitor
- ❖ Portable device with Android 6.0 operating system
- ❖ An ink or laser printer with P2P infrastructure for invoice printing

Software Requirements

Software needed to operate:	Description:
Windows OS – 10 Android – 6.0	Chorez – Biz is supported by Windows for products running on stationary machines, and Android for those who prefer the portable version of our line of products. The ability for cross-platform software communication is one of the leading development activities currently pursued by the development team.
Microsoft SQL Server 2019 My SQL – 5.0	Those who wish to store their data redundantly on off-site servers may decide to utilize Microsoft SQL Server implementation of their database. Those who look for a cheap and reliable solution for data storage might prefer hosting information on a local machine.