Software Requirements Specification

For

Pokémon Go Back

Prepared by:

Name	ID
Amitt Bhardwj	40037899
Gaganjot Kaur	40037933
Gunpreet Ahuja	40041311
Gurpreet Kaur Lotey	40038315
Gurpreet Raju	40013682
Harinder Kaur	40041313
Mandeep Singh	40040964
Navjot Singh	40014477
Navrang Pal Kaur Dhaliwal	40047555
Simranjit Singh Bachher	40040601

Instructor name: Stuart Thiel

Table of Content

1 Introduction

1.1 Purpose

The purpose of this document is to define and communicate the software requirements of "Pokémon Go Back". The requirements are documented as a means to provide a common understanding to stakeholders. The requirements will be verified through reviews. The structure of this software specification is inspired by IEEE standard 830-1998. Diagrams and detailed use cases it contains apply UML notations and description.

1.2 Project overview

Pokémon Go Back is a standalone application, which is based on Pokémon Go, which is a trading card game. It is a game developed in single player mode and the game is played between a user and the AI. The players act as Pokémon trainers and they battle their Pokémon with special abilities and powers to reach the final winning stage.

1.3 Project Scope

Pokémon Go Back is an interactive card game with a graphical user interface. The goal is to develop a card game using the software engineering processes. The game, which is developed, does not need any special expertise or skills. The developed game will be tested with various test cases and techniques. The basic purpose of the game is entertainment.

1.4Requirement Gathering Techniques

For requirements gathering, we have used following techniques:

1. Searching for similar products.
2. Reading Documentation of existing products.
3. Customer's interaction

1.5 Roles AND Contribution

For Roles and contribution, please refer to Section 7 of Software Design Document.

1.6 Definitions, Acronyms and Abbreviations

AI	-	Artificial intelligence
PGB	-	Pokémon Go Back
COCOMO	-	Constructive Cost Model
GUI	-	Graphical User Interface
KLOC	-	1,000 Lines of Code
SRS	-	Software Requirements Specification
Http	-	Hypertext transfer protocol

1.7 References and Acknowledgments

- http://www.gnu.org/copyleft/gpl.html
- http://www.pokemon.com/us/pokemon-tcg/play-online/
- https://en.wikipedia.org/wiki/Software_requirements_specification
- http://groups.engin.umd.umich.edu/CIS/course.des/cis525/js/f00/kutcher/kutcher.html

2 Overall Description

2.1 Project Perspective

PGB is a windows based java application, which is a card game. It is a single player game in which user battles with AI. Two decks will be provided to each opponent and each deck contains shuffled cards. Each card has its name, type, category and one or more ability. The game is played in alternative turns and in each turn player uses the card's ability to damage the opponent's Pokémon. Whichever player knocks down two Pokémon's of opponent is the winner.

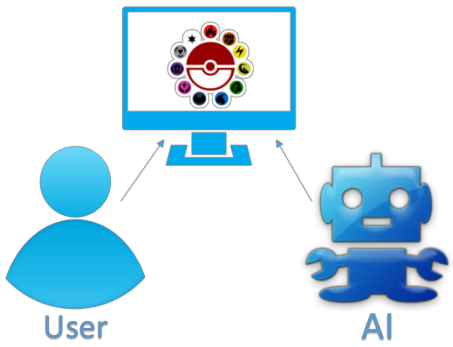

Figure 2.1- Pokémon Go Back perspective

2.2 Product Functionality

1. Deck is loaded and each deck contains 30 cards.
2. User can set active and bench cards.
3. Energy cards can be allotted to the user.
4. User can attack the opponent and put damage on his card by using Pokémon's abilities.
5. Trainer card can be used on Pokémon.

2.3 Users and Characteristics

Pokémon go back is a single player game which can be played between user and AI. User and AI will play their turns to win the game.

Present user characteristics using the selected technique.

A **Pokémon go back** has a massive user base with an eclectic set of user characteristics and needs. They can be students, teens, kids, divorced, single, may belong to any ethnicity or have any

sexual orientation. However, we have a few user profiles to which majority of people relate due to one or another attributes of such profiles. Thus, to enhance these user profiles and make them more efficient for use in the design purposes personas should be best for documenting user characteristics.

A few user profiles are as follows:

1. Students
2. Bloggers and Authors
3. Developers
4. Teens

The representation of such user profiles in personas would be as follows:

Persona	Major User
Photo	
Name	Stuart Thiel
Job	Professor
Demographics	39 years old, married, father of one
Goals	He is a passionate professor who devotes proper time to his professional and personal matters. He is dexterous in his work and knows technicality of

	Computers very well. • Spending spare time efficiently • Play game without internet connection
Context of Work	Works from home as well as from his office. He is quite comfortable with computers.

Persona	All other categories [Users]
Photo	
Name	Martha Miller
Job	Teen/Student/Developer/Author or etc.
Demographics	36 years old, married to Henry, mother of two

Goals	She provides online cloud services to the users such as hosting websites and storing backup data. She is very hard working and often works late due to complexity of her business. A frequent computer user. • Want to use their free time by playing simple games.
Context of Work	Works from home as well as office and is an expert of computers.

2.4 Operating Environment

Pokémon go back is a standalone application, which is developed in Java, programming language by using Java platform. Pokémon go back is a windows based application which does not require any special environment.

2.5 Assumptions and Dependencies Constraints

- Each card follows a specific format to figure out how it can be played. Each card has a name, type, possibly a category and possibly one or more abilities.
- Pokémon may be affected by the status effects, which affect how they may be used during a player's turn. Status effects are always removed when retreating or evolving. Each status effect may only be applied once per Pokémon.
- Abilities on cards act when appropriate. A Stadium card would create a permanent effect, Supporter and Item cards apply an effect when played. Pokémon abilities apply when they attack.
- Loading the game will first check that deck-building rules are in effect.
    1. Each deck must have exactly 60 cards
    2. You may have any number of cards that are of type: energy
    3. You may have at most 4 of any card that is not of type: energy
    4. Each deck must have at least one type: Pokémon card

5. All cards must be "playable"

- Decks are not shuffled at the start of the game. While certain abilities may be used to shuffle the deck. The initial decks should be played in the order of the files provided.

2.6 Budget

We estimate our budget for this part of the project will be $14718. We have used COCOMO technique to calculate the total cost. The explanation on how we derived estimations follows.

Our domain model contains 15 object classes. From our experience, we know we will need an average of 80 new lines of code per class, hence, we expect to have 1200 total lines of code. Our project is **SEMI-DETACHED** which means an intermediate, (in size and complexity), software project in which teams with mixed experience levels must meet a mix of rigid and less than rigid requirements.

Effort =a*KLOC^b, in person/months, with KLOC = lines of code, (in the thousands)
Our Project is Semi-detached so coefficients will be
a= 3
b= 1.12
Effort will be = 3.6796308 person/month
Total Estimated Cost: 3.6796308 * $4,000.00= $14718

(Based on a monthly salary rate of $4,000.00)

Element	**Cost**	**Value**
Deck	Medium	High
Game Board 1. Bench Pokémon 2. Hand Card 3. Active Pokémon 4. Prize Card 5. Discard Pile	 Low Low Medium Low Low	 Medium Medium High Low Low
Cards 1. Pokémon 2. Energy 3. Trainer	 Medium Low Medium	 High High High
Abilities	High	High

AI Player	High	High
Use Ability	High	High
Add Energy	Low	Medium
Retreat	Medium	Medium
Evolve	Low	Low

Cost and Value in Cards

Element	**Cost**	**Value**
Glameow	Medium	Low
Pikachu Libre	High	Low
Pikachu	Low	Low
Raichu	High	Low
Shellder	Medium	Low
Seaking	Medium	Low
Goldeen	Medium	Low
Frogadier	Medium	Low
Froakie	Low	Low
Cloyster	High	Low
Suicune	High	Low
Swanna	Low	Low
Ducklett	Medium	Low
Purugly	Medium	Low
Manectric	Medium	Low

Electrike	Medium	Low
Electivire	High	Low
Electabuzz	Low	Low
Helioptile	High	Low
Tierno	Low	Low
Potion	Low	Low
Misty's Determination	Low	Low
PokÃ©mon Center Lady	Low	Low
Clemont	Low	Low
Lightning	Low	Low
Water	Low	Low
Meowstic	High	Low
Jynx	Medium	Low
Jirachi	High	Low
Meowth	Medium	Low
Machop	Medium	Low
Doduo	Medium	Low
Dodrio	High	Low
Geodude	Medium	Low
Zubat	Low	Low
Haunter	High	Low
Gastly	Medium	Low
Diglett	Medium	Low

Dugtrio	High	Low
Slowpoke	Medium	Low
Hitmonlee	Medium	Low
Hitmonchan	Medium	Low
Machoke	Medium	Low
Espurr	High	Low
Persian	Medium	Low
Floral	Low	Low
PokÃ© Ball	Low	Low
Shauna	Low	Low
PokÃ©mon Fan Club	Medium	Low
Switch	High	Low
Energy Switch	Medium	Low
Red Card	Low	Low
Wally	Low	Low
Fight	Medium	Low
Psychic	Medium	Low

3. **Domain Model**

Domain Model

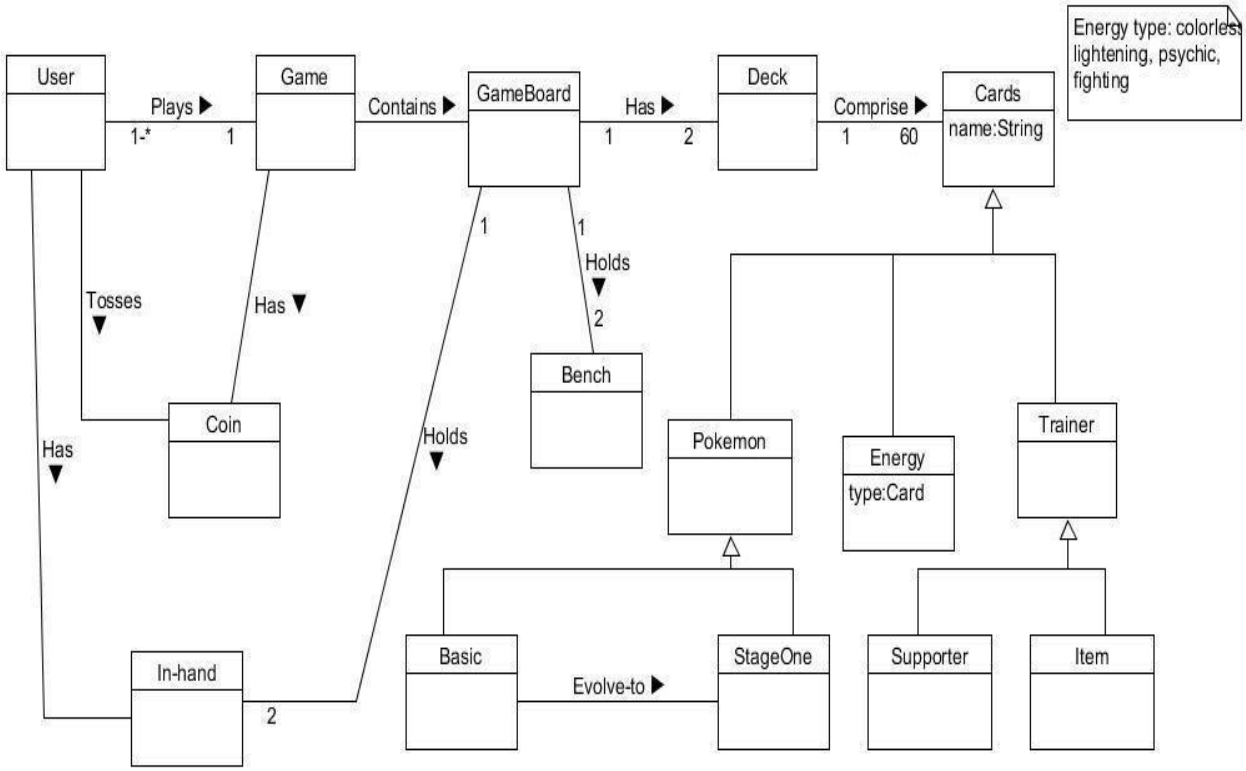

Figure 3.1 Domain Model

4 Specific Requirements

4.1 Functional Requirements

1. The game should be standalone application.
2. The game should have single player mode.
3. Player will play against an AI.
4. Two Decks should be parsed when game starts.
5. Cards should be dealt.
6. One deck should be made for each player with 60 cards. Each deck has three types of cards - Pokémon card, Energy card and Trainer card.
7. Each card type should have further sub categories.***(For more details refer to section 2.1 of supplementary specification document)***
8. Every Pokémon should consist attacking abilities and hit points***(For more details refer to section 2.2 of supplementary specification document)***
9. Every Pokémon may be affected by a status.***(For more details refer to section 2.3 of supplementary specification document)***
10. Mulligan***(For more details refer to section 2.4 of supplementary specification document)***
11. Play and bench Initial Pokémon
12. Draw Card at Turn Start
13. Evolve Pokémon
14. Energy should be placed to play
15. End Turn
16. Retreat Pokémon
17. Knockout Pokémon
18. There should be some Prize Cards
19. Result should be displayed as a Win or lose
20. Hand Sizes, Deck Sizes and Discard Sizes should be displayed
21. Energy on Active Pokémon and Benched Pokémon should be checked
22. Pokémon Ability as well as cards in Hand could be seen
23. See Current and Max Pokémon Health

4.2 Behaviour Requirements

4.2.1 Use Case View

A use case defines a goal-oriented set of interactions between external actors and the system under consideration. Since sometimes, we will not be able to specify completely the behaviour of the system by just State Diagrams.

Use Case Diagram

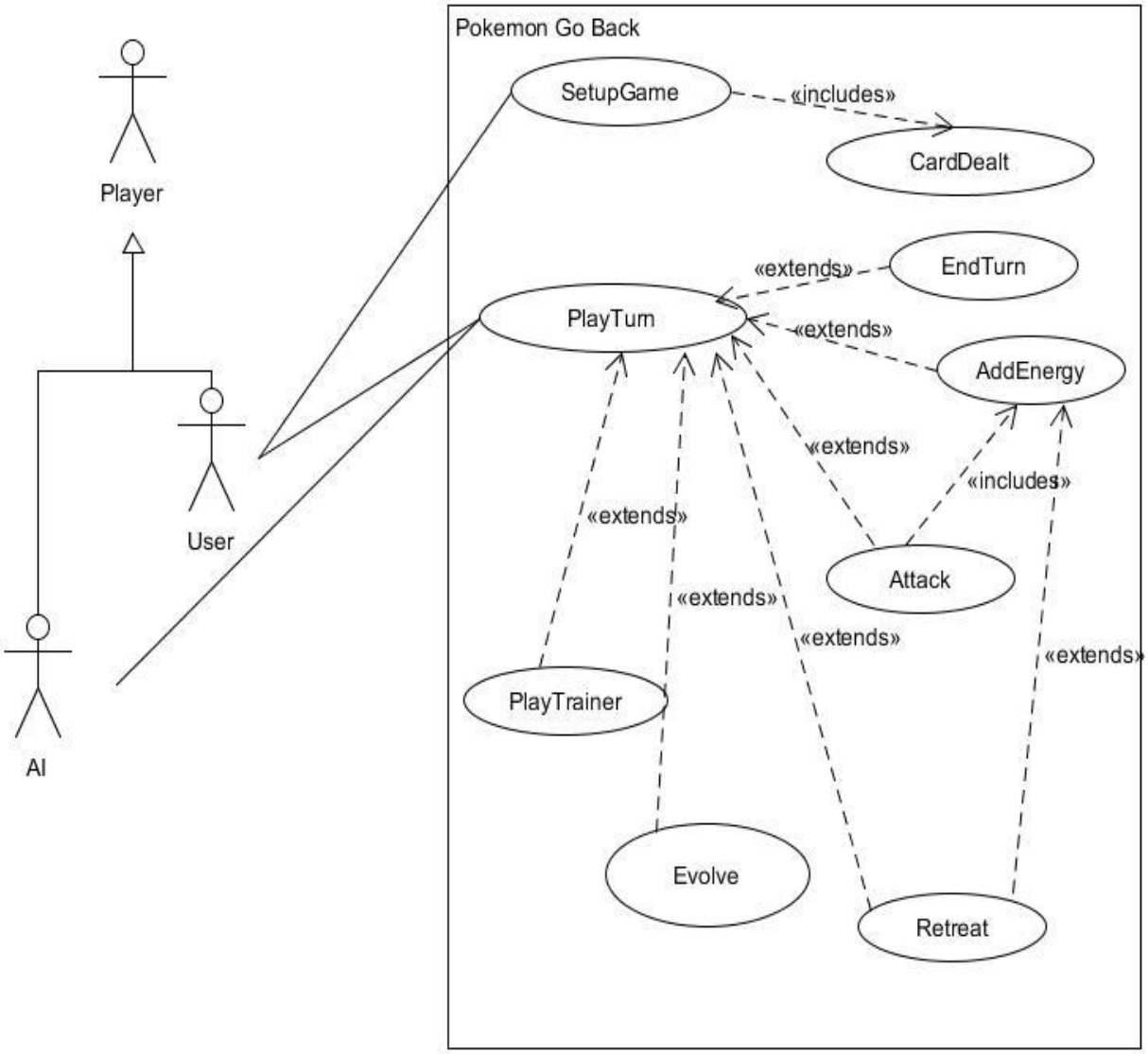

Figure 4.1 Use Case Diagram

4.2.1.1 Setup Game

Use Case	UC1
Name:	Setup Game
Priority	High
Primary Actor	Player
Actor's goal	To start the game
Secondary Actor	N/A
Precondition	Player should have game
Post Condition	Game started
Main Scenario	→ 1. Player will start the game. ← 2. System will parse the cards. ← 3. System will ask for toss. → 4. Player will choose head or tail. ← 5. Sequence of turn will be decided. ← 6. System will dealt cards. UC2:Cards Dealt → 7. Player will decide the initial active Pokémon. → 8. Player will choose bench Pokémon.
Alternative Scenario	7.a There is mulligan state(No basic Pokémon). b. Player will end the turn UC 9: End Turn.

4.2.1.2 Cards Dealt

When the game starts the system picks seven cards from deck and places it in player's hand. When player's turn starts, one more card is added to the hand. If no cards are present in the deck, system cannot place cards in player's hand.

4.2.1.3 Play Turn

Use Case	UC3
Name:	Play Turn
Priority	High
Primary Actor	Player
Actor's goal	To play the game
Secondary Actor	N/A
Precondition	Player should have active basic Pokémon
Post Condition	Player took its turn

Main Scenario	→ 1. Player can choose the following options according: - A. Player adds energy to active basic Pokémon (Include Add Energy (UC4).) B. Player uses trainer card. (Include Play Trainer (UC5).) C. Player evolves to stage 1 Pokémon from active basic Pokémon. Include: Evolve (UC6). D. Player retreats active Pokémon with available bench Pokémon. Include: Retreat (UC7). E. Player uses the attack from his active Pokémon. Include: Attack (UC8). 1. Player will place Pokémon on bench. 2. System will pass turn to opponent.
Alternative Scenario	1.a. Player will choose active Pokémon if active Pokémon is knocked out 1. A. a. Player adds energy to any Pokémon currently on bench. 1.A. b. Player's in-hand cards does not contain any energy card. 1.B. a. Player's in-hand cards does not contain any trainer card. 1. C. a. Player's in-hand cards does not contain any stage 1 card.

4.2.1.4 Add Energy

When Pokémon card is there. Player chooses energy card from In-hand and adds it to the Pokémon. Energy card is removed from In-hand.

4.2.1.5 Play Trainer

Player will select trainer card. It will play card ability and card will go to discard pile.

4.2.1.6 Evolve

Use Case	UC6
Name:	Evolve
Priority	Medium
Primary Actor	Player
Actor's goal	To evolve the Pokémon
Secondary Actor	N/A
Precondition	Player should have Basic Pokémon on Game Board
Post Condition	Pokémon type changed to Stage One
Main Scenario	→ 1. Player selects Stage One Pokémon from in-hand cards → 2. Player places it on basic Pokémon

	← 3. Pokémon type changes to Stage One
Alternative Scenario	1.a No Stage One Pokémon in-hand

4.2.1.7 Retreat

Use Case	UC7
Name:	Retreat
Priority	Medium
Primary Actor	Player
Actor's goal	To retreat the Pokémon
Secondary Actor	N/A
Precondition	Player should have Pokémon in bench
Post Condition	Pokémon retreated

Main Scenario	→ 1. Player selects retreat option → 2. Player selects one of the Bench Pokémon's to be replaced as active Pokémon ← 3. System discards energies of Pokémon to be retreated. ← 4. Card exchanges positions.
Alternative Scenario	1.a No Pokémon Bench

4.2.1.8 Attack

Use Case	UC8
Name:	Attack
Priority	Medium
Primary Actor	Player
Actor's goal	To attack the opponent's Pokémon on Game Board
Secondary Actor	N/A
Precondition	Pokémon should have sufficient energy to attack

Post Condition	Opponent Pokémon's damage increased
Main Scenario	→ 1. Player attacks opponents Pokémon ←2. Opponent's damage value increases
Alternative Scenario	←2 a. Opponent's Pokémon will be knocked out 2. b Player will win.

4.2.1.9 End Turn

Player do not want to attack opponent or do not have enough energy to attack or there is a mulligan state then player will end his turn and turn will pass to opponent.

5 Conclusion

This document has introduced how different types of requirements can be organized. The categorization of requirements makes it easier to develop the list of questions and to identify gaps in knowledge. The aim is to ensure that the product to be developed is fully understood from all angles. Brief idea on budget is also proposed in this document by selecting 4000$ as the average monthly wage. It explains various techniques to smoothly run a software engineering process from initial to final phase.