PokeDex Use Case

**Document Information**

|  |  |
| --- | --- |
| **Document Title** | *Active Pokedex* |
| **Document Owner** | Mike Minassian |
| **Version** | 1.0 |
| **Status** | Pre-phase |
| **Date** | 3/31/2021 |

1. **Brief Description**

*Insert a 1-2 sentence description of this use case. Be sure to include a starts when / ends when statement to clarify the beginning and ending points of the scope of this process or piece of functionality.*

The user should be able to add a pokemon from the web portal as well as view all of those that have been added to the Database from the save web portal.

1. **Actors**

*List any roles or systems involved with this process or use case. A person or system fulfilling a role will be the actor in one of the steps.*

* The user typing in the name of the pokemon (Actor 1)
* System calling out to the internet for details (Actor 2)

1. **Pre-Conditions**

*List anything that must be true before this process or functionality begins. Preconditions should be states that a system can validate to be true. A common example is that a specific Actor has logged into the System.*

* The user has an internet connection

1. **Basic Flow**

*The basic flow is the normal course of events, otherwise called the “happy path.” Ask yourself, what happens most of the time and you’ll discover the steps that belong here. You’ll want your basic flow to cover the full scope of activities between the starts when and ends when.*

*Create a numbered list of each step below. I recommend using the Word “numbered list” functionality to automatically number the list.*

1. User opens application
2. User is prompted to enter the name of a pokemon
3. the user enters a pokemon name
4. user presses enter
5. the application sends out a call to database
6. the application sends the information of the pokemon
7. the application displays information to the user
8. the application then asks for the name of the next pokemon
9. **Alternate/Exception Flows**

*An alternate flow is a variation from the basic flow. Alternatives can be triggered at any step in the basic flow and often reinsert the actors back into the basic flow.*

*An exception flow is an error, or a negative condition. When an exception is encountered, it prevents the process from finishing through to its conclusion until it’s addressed.*

*Number your alternate and exception flows to indicate the step at which the variation occurs. For example, a variation on step 3 could be listed as 3a and a second variation as 3b, and so forth.*

*Describe the alternate functionality and then identify at what step in the basic flow this variation picks back up. For exception flows that result in the use case ending, simply write, “Use Case Ends.”*

3a - User enters the name of pokemon already in database.

Data is entered into the database and confirmation is shown.

1. **Post Conditions**

*Post-conditions indicate what must be true of the state of the system after the steps of the use case are complete. These should be true for the basic flow and all alternate flows. Exception flows may have different post-conditions or none at all.*

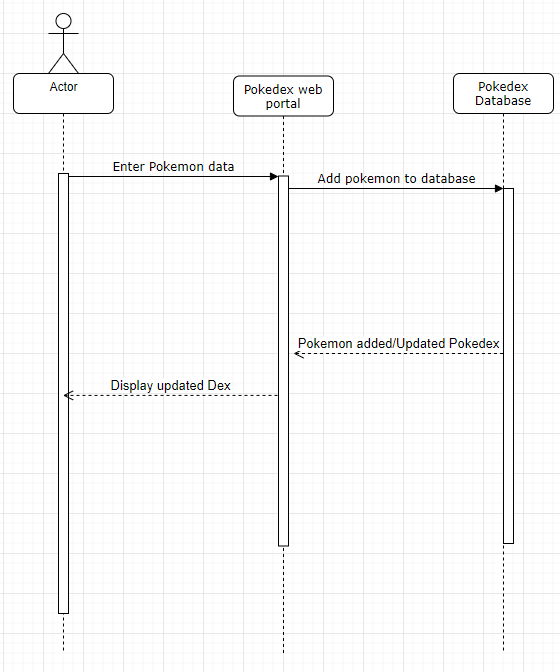
* Application should be ready to accept another name for search

1. **Supplemental Requirements**

*This is a special section I use to hold miscellaneous requirements related to the use case. Often you’ll find BAs including a Business Rules section or other collection of information related to the use case. These may or may not be actual requirements – you’ll want to establish a clear pattern and communicate that clearly and ensure it’s consistent with how your organization documents this type of requirement. I’ve also used this section to capture the most salient decisions and notes so they are stored right with the use case for future consideration.*

1. **Visual Model**

*Many use cases are enhanced by a visual model. A simple work-flow diagram can be used to visually show the sequence of steps and alternate and exception flows. A user interface mock-up can be used to show a possible representation of these user requirements in an interface (or a desired representation). In some organizations, a more formal UML diagram may be appropriate.*



**Revision History**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| V. | Date | Author | Description | Status |
| 1.0 | 4/7/2021 | Mike M | Pokedex application |  |
|  |  |  |  |  |