# A picture containing text, clipart Description automatically generated CSC 431 Coronapyrus Software Requirements Specification (SRS)

**Team 11**

|  |  |
| --- | --- |
| Alexander Claman | Scrum Leader |
| Noah Jaccard | Team Member |
| James McSweeney | Team Member |

# Version History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author(s) | Change Comments |
| 1.0 | 02.23.2021 | Team 11 | Rough Draft #1 |
| 1.1 | 03.09.2021 | Team 11 | Rough Draft #2 |
| 1.2 | 05.05.2021 | Team 11 | Rough Draft #3 |

# Table of Contents

Table of Contents 3

1. System Requirements 5

1.1. Functional Requirements 5

1.1.1. Retrieve COVID Information 5

1.1.2. Process and Visualize COVID Information 5

1.1.3. Provide Help Information 6

1.1.4. Respond to a Slack Slash Command 6

1.1.5. Respond to a Discord Bot Command 7

1.1.6. Provide Source Information 7

1.2. Nonfunctional Requirements 8

1.2.1. Retrieve COVID Information 8

1.2.2. Define a User Request’s Scope 8

1.2.3. Define a User Request’s Format 8

2. Constraints 9

2.1. Tool Constraints 9

2.1.1. Required Python Packages 9

2.1.2. Data Availability 9

2.2. Language Constraints 9

2.2.1. Python Constraint 9

2.3. Platform Constraints 9

2.3.1. Python Package Management Platform 9

2.4. Network Constraints 9

2.4.1. Request COVID Information 9

2.5. Deployment Constraints 10

2.5.1. Python Environment 10

2.6. Budget and Schedule Constraints 10

2.6.1. Time Constraint 10

2.6.2. Funding Constraint 10

3. Requirements Modeling 11

3.1. Respond to a User Request 11

3.2. Respond to an Application Command 12

3.3. Class Diagram 13

# System Requirements

## Functional Requirements

### Retrieve COVID Information

|  |  |
| --- | --- |
| ID | FR1 |
| Title | Retrieval of COVID Information |
| Description | A user request must be parsed. Any information with a user-defined scope must be retrieved and made available internally for subsequent formatting. |
| Priority | 0 |
| Precondition(s) | User makes a request for COVID information.  The user request’s scope is properly specified. |
| Basic Flow | Parameters associated with the user request define scope and format.  The scope parameter may include range of dates over which information is required, location of information, and type of information (media/news or data/characteristic).  The information within the given scope is retrieved.  The format parameter may denote returned information as an article link & summary for media, a table for numerical data, or a graph for data series. |
| Postconditions(s) | Information in the requested scope and format is retrieved and available for processing. |
| Use Case Diagram | 3.1. Respond to a User Request |

### Process and Visualize COVID Information

|  |  |
| --- | --- |
| ID | FR2 |
| Title | Processing and Visualization of COVID Information |
| Description | COVID information returned after a user request is made must be processed so it can be effectively visualized as a message, graph, or table. |
| Priority | 0 |
| Precondition(s) | COVID information matching the scope of a user request (see NFR2) have been retrieved and are available for processing.  The user request’s return format is properly specified. |
| Basic Flow | COVID information is processed based on the provided format.  For media information requests, news articles are returned as a list of dictionaries, with each dictionary holding a summary of the article, the title of the article, and a link to the article.  For JHU COVID data requests, data can be returned tabularly, as a Pandas DataFrame containing the requested data, or graphically, as a visualization of data created with Matplotlib. |
| Postconditions(s) | Information in the user-defined format is returned to the user. |
| Use Case Diagram | 3.1. Respond to a User Request |

### Provide Help Information

|  |  |
| --- | --- |
| ID | FR3 |
| Title | Provide Help Information |
| Description | Provide help information for the Discord bot or Slack application upon user request |
| Priority | 1 |
| Precondition(s) | User must be authenticated in their current platform (Discord or Slack)  Must have an active app or bot with Coronapyrus functionality enabled |
| Basic Flow | User issues a platform-dependent command requesting help, a list of viable commands and command parameter formatting is returned to the user. |
| Postcondition(s) | The bot/app presents the help information to the user in a message. |
| Use Case Diagram | 3.2. Respond to an Application Command |

### Respond to a Slack Slash Command

|  |  |
| --- | --- |
| ID | FR4 |
| Title | Respond to a Slack Slash Command |
| Description | Develop a Slack app which will use the Coronapyrus package to retrieve and visualize data as requested by a user when a Slash Command is invoked |
| Priority | 2 |
| Precondition(s) | User must be authenticated in Slack.  User command must have valid parameters.  Must have an app with Coronapyrus functionality enabled built and in the Slack App Directory. |
| Basic Flow | User issues a Slash Command requesting COVID information with parameters dictating the scope and format of the request.  The Slack app uses the Coronapyrus package to process the user request.  The correctly formatted COVID information is then returned to the Slack app to be presented to the user. |
| Postcondition(s) | The Slack app presents the data to the user in a message. |
| Use Case Diagram | 3.2. Respond to an Application Command |

### Respond to a Discord Bot Command

|  |  |
| --- | --- |
| ID | FR5 |
| Title | Respond to a Discord Bot Command |
| Description | Develop a Discord bot which will use the Coronapyrus package to retrieve and visualize data as requested by a user when a command is invoked |
| Priority | 2 |
| Precondition(s) | User must be authenticated in Discord.  User command must have valid parameters.  Must have a bot with Coronapyrus functionality enabled on the current Discord server. Bots can be added to Discord servers according to the Discord platform’s instructions and with the appropriate bot link. |
| Basic Flow | User issues a command to the Discord bot requesting COVID information with parameters dictating the scope and format of the request.  The Discord bot uses the Coronapyrus package to process the user request.  The correctly formatted COVID information is then returned to the Discord bot to be presented to the user. |
| Postcondition(s) | The Discord bot presents the information to the user in a message. |
| Use Case Diagram | 3.2. Respond to an Application Command |

### Provide Source Information

|  |  |
| --- | --- |
| ID | FR6 |
| Title | Provide Source Information |
| Description | Provide the source of COVID information for the Discord bot or Slack application upon user request |
| Priority | 3 |
| Precondition(s) | User must be authenticated in their current platform (Discord or Slack)  Must have an active app or bot with Coronapyrus functionality enabled |
| Basic Flow | User issues a platform-dependent command requesting the source of the application’s COVID-19 information, the relevant information is returned to the user. Sources for media information will be provided as part of the processing step. The source for COVID data, the JHU COVID database posted on Github, will be linked if requested. |
| Postcondition(s) | The bot/app presents information about the source to the user in a message. |
| Use Case Diagram | 3.2. Respond to an Application Command |

## Nonfunctional Requirements

### Retrieve COVID Information

|  |  |
| --- | --- |
| ID | NFR1 |
| Title | Retrieve COVID Information |
| Description | Once a user request is made, relevant COVID information must be gathered from reliable sources for the user. The open-source Python package news\_fetch will be used to retrieve information scoped by geographic region and by range of dates. The pandas.read\_csv() function from the open-source Python package Pandas will be used to retrieve COVID data from links to the John Hopkins University Github database. |
| Priority | 0 |
| Applicable FRs | FR1, FR2, FR4, FR5 |

### Define a User Request’s Scope

|  |  |
| --- | --- |
| ID | NFR2 |
| Title | Request Scope |
| Description | A data structure, object, or class will be designed such that a user can properly define the scope of their request. This scope will detail the type of information to be retrieved (raw data or news & media information), the range of dates over which the data should be retrieved for, and the geographic region the data should be retrieved for. |
| Priority | 1 |
| Applicable FRs | FR1, FR2, FR4, FR5 |

### Define a User Request’s Format

|  |  |
| --- | --- |
| ID | NFR3 |
| Title | Request Format |
| Description | A data structure, object, or class will be designed such that a user can properly define their desired response format. For media information requests, news articles will be returned as a list of dictionaries, with each dictionary holding a summary of the article, the title of the article, and a link to the article. For COVID data requests, data can be returned tabularly, as a Pandas DataFrame containing the requested data, or graphically, as a visualization of data created with Matplotlib. |
| Priority | 1 |
| Applicable FRs | FR1, FR2, FR4, FR5 |

# Constraints

## Tool Constraints

### Required Python Packages

|  |  |
| --- | --- |
| Title | Required Python Packages |
| Description | The Pandas, Matplotlib, and newsfetch packages must be installed for Coronapyrus to function properly. |
| Priority | 0 |

### Data Availability

|  |  |
| --- | --- |
| Title | Data Availability |
| Description | Any numerical data used will be retrieved from publicly available John Hopkins University databases (<https://github.com/CSSEGISandData/COVID-19/tree/master/csse_covid_19_data>). If this data becomes unavailable, an alternative source of COVID data will need to be used. |
| Priority | 0 |

## Language Constraints

### Python Constraint

|  |  |
| --- | --- |
| Title | Python Constraint |
| Description | The only supported language for the Coronapyrus package at this time is Python for both development and use. |
| Priority | 0 |

## Platform Constraints

### Python Package Management Platform

|  |  |
| --- | --- |
| Title | Python Package Management Platform |
| Description | Independent of operating system, a Python package management strategy or system (such as pip) is required. |
| Priority | 0 |

## Network Constraints

### Request COVID Information

|  |  |
| --- | --- |
| Title | Request COVID Information |
| Description | A proper connection to the network is required to download recent COVID information. |
| Priority | 0 |

## Deployment Constraints

### Python Environment

|  |  |
| --- | --- |
| Title | Python Environment |
| Description | The Coronapyrus package will be retrievable from the pip package manager and from Github. It will be deployable in any Python development environment. A Python distribution such as Anaconda is required to create applications or scripts using Coronapyrus. |
| Priority | 0 |

## Budget and Schedule Constraints

### Time Constraint

|  |  |
| --- | --- |
| Title | Time Constraint |
| Description | This project must be completed by the end of the Spring 2021 University of Miami school semester. |
| Priority | 0 |

### Funding Constraint

|  |  |
| --- | --- |
| Title | Funding Constraint |
| Description | There is no funding for this project being sourced from a client at this time; as such, this will be open source. |
| Priority | 5 |

# Requirements Modeling

## Respond to a User Request

|  |  |
| --- | --- |
| Name | Respond to a User Request Use Case |
| Description | This is the primary use case for the Coronapyrus package. |
| Actors | The User. |
| Trigger | This use case is initiated when the User requests information about COVID-19. |
| Precondition(s) | None. |
| Basic Flow | 1. The User’s request is processed by the Coronapyrus package and any needed data or media is retrieved based on the User’s defined scope. 2. The data or media retrieved are processed into responses or visualized then converted into responses based on the User’s defined format. 3. A response is returned to the User containing the information within the scope they defined, in the format they requested it in. |
| Exceptions | The Coronapyrus package will raise errors if the requested data cannot be found or cannot be retrieved from the network. This includes requests for data that does not exist (such as COVID statistics from future dates) or a lack of internet connection. |
| Postcondition(s) | The User has received a properly formatted response containing the information they requested. |

Figure 1 – Process a User Request Use Case Diagram (Rough Draft)

Diagram

Description automatically generated

## Respond to an Application Command

|  |  |
| --- | --- |
| Name | Respond to an Application Command Use Case |
| Description | This is the primary use case for the Slack app/Discord bot. |
| Actors | The User. |
| Trigger | This use case is initiated when the User issues a command to the app/bot requesting information about COVID-19. |
| Precondition(s) | The User must be authenticated in whichever application (Discord/Slack).  The app/bot must be enabled in the user’s environment (Discord server/Slack App Directory). |
| Basic Flow | 1. If the User’s command is a request for help with the app/bot or for information about the sources for COVID information, these are returned. 2. If the User’s command is a request for COVID information, that request is passed to and processed by the Coronapyrus package; any needed data or media is retrieved and visualized based on the User’s defined scope and format. 3. The returned information is then presented to the user as media, graphs, tables, a message, links, or an error message depending on the Coronapyrus package’s response. |
| Exceptions | None. |
| Postcondition(s) | The User has received a properly formatted response containing the information they requested or an error message. |

Figure 2 – Respond to an Application Command Use Case Diagram (Rough Draft)

Diagram

Description automatically generated

## Class Diagram

Figure 3 – Class Diagram (Rough Draft)

Diagram

Description automatically generated