MIAMI

CSC 431

Coronapyrus

Software Requirements Specification (SRS)

Team 11

Alexander Claman	<Role>
Noah Jaccard	<Role>
James McSweeney	<Role>

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

Table of Contents

1. System Requirements

1.1. Functional Requirements

1.1.1. Process a User Request

ID	FR1
Title	User Request Processing
Description	A user request must be parsed. Any information with a user-defined scope must be retrieved, processed and/or visualized, and returned in a user-defined format.
Priority	0
Precondition(s)	User made a request for COVID information.
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. The retrieved information is processed and visualized in the format requested.
Postconditions(s)	Information in the requested scope and format is returned to the user.
Use Case Diagram	3.1. Process a User Request

1.1.2. Process and Visualize Data

ID	FR3
Title	Data Processing and Visualization
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)	User made a request for COVID information within a particular scope and format. COVID data matching the scope have been retrieved and are available for processing.
Basic Flow	COVID information is processed based on the provided format. Information from news articles is 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. Information from JHU COVID data can be returned multiple ways. It can be returned as a Pandas DataFrame containing the requested data, or a visualization of data can be created with Matplotlib and returned.
Postconditions(s)	Information in the user-defined format is returned.
Use Case Diagram	3.1. Process a User Request

1.1.3. Respond to a Discord Bot Command

ID	FR3
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 Must have a bot with Coronapyrus functionality enabled on the current Discord server
Basic Flow	User issues a command to the bot with parameters dictating the scope and format of the request, the bot uses the Coronapyrus package to process the user request, the relevant information is returned to the user
Postcondition(s)	The bot presents the data to the user in a message.
Use Case Diagram	3.2. Respond to an Application Command

1.1.4. 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 Must have an app with Coronapyrus functionality enabled built and in the Slack App Directory
Basic Flow	User issues a Slash Command with parameters dictating the scope and format of the request, the app uses the Coronapyrus package to process the user request, the relevant information is returned to the user
Postcondition(s)	The app presents the data to the user in a message.
Use Case Diagram	3.2. Respond to an Application Command

1.1.5. Provide Source Information

ID	FR5
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
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

1.1.6. Provide Help Information

ID	FR6
Title	Provide Help Information
Description	Provide help 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 help, the relevant information 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

1.2. Nonfunctional Requirements

1.2.1. 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.
Priority	1
Applicable FRs	FR1, FR2, FR3, FR4

1.2.2. 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.
Priority	1
Applicable FRs	FR1, FR2, FR3, FR4

1.2.3. 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/medium.
Priority	1
Applicable FRs	FR1, FR2, FR3, FR4

2. Constraints

2.1. Tool Constraints

2.1.1. Required Python Packages

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

2.1.2. 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

2.2. Language Constraints

2.2.1. 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

2.3. Platform Constraints

2.3.1. 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

2.4. Network Constraints

2.4.1. Request COVID Information

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

2.5. Deployment Constraints

2.5.1. 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

2.6. Budget and Schedule Constraints

2.6.1. 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

2.6.2. Funding Constraint

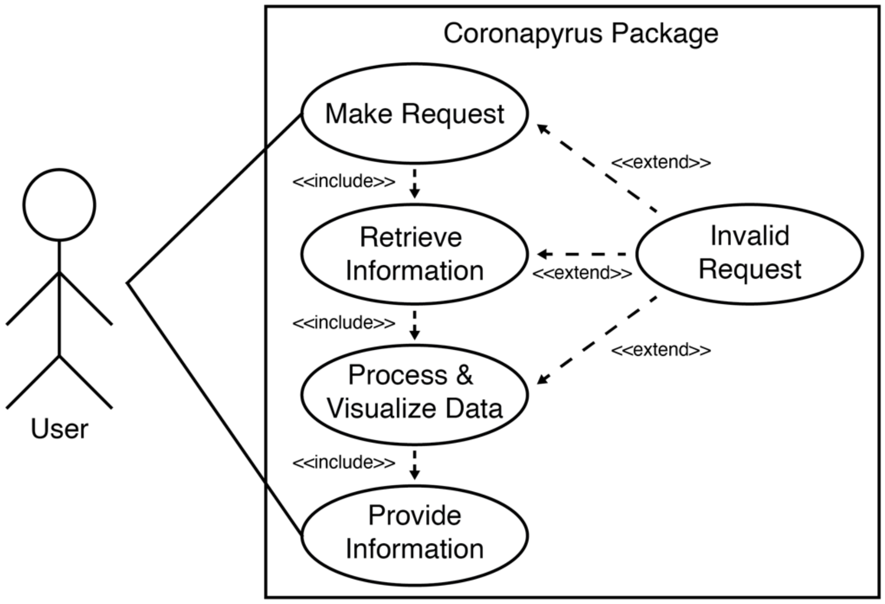
Title	Funding Constraint
Description	There is no funding for this project being sourced from a client at this time; as such, this will be open source.
Priority	5

3. Requirements Modeling

3.1. Process a User Request

Name	Process 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.

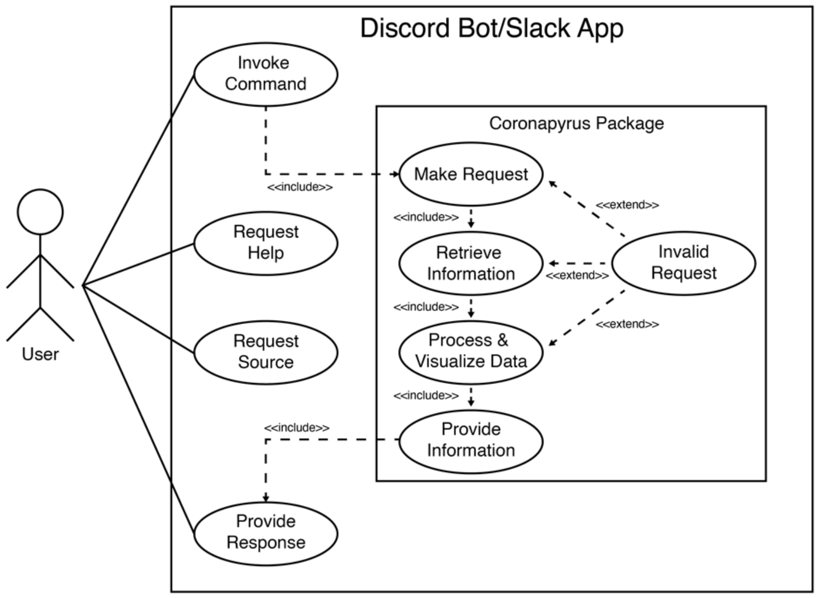
Figure 1 – Process a User Request Use Case Diagram (Rough Draft)



3.2. Respond to an Application Command

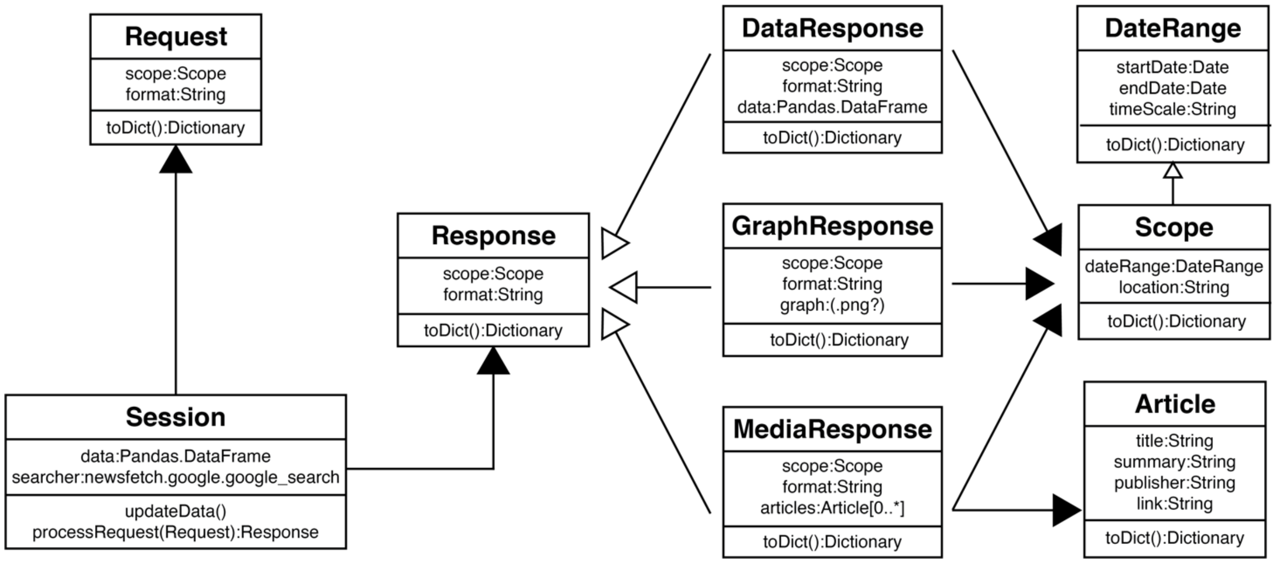
Name	Respond to an Application Command Use Case
Description	This is the primary use case for the Slack app/Discord bot.
Actors	The User.
Trigger	This use case is initiated when the User issues a command to the app/bot requesting information about COVID-19.
Precondition(s)	The User must be authenticated 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)



3.3. Class Diagram

Figure 3 – Class Diagram (Rough Draft)


Request
scope:Scope
format:String
toDict():Dictionary
Session
data:Pandas.DataFrame
searcher:newsfetch.google.google_search
updateData()
processRequest(Request):Response
Response
scope:Scope
format:String
toDict():Dictionary
DataResponse
scope:Scope
format:String
data:Pandas.DataFrame
toDict():Dictionary
GraphResponse
scope:Scope
format:String
graph:(.png?)
toDict():Dictionary
MediaResponse
scope:Scope
format:String
articles:Article[0..*]
toDict():Dictionary
DateRange
startDate:Date
endDate:Date
timeScale:String
toDict():Dictionary
Scope
dateRange:DateRange
location:String
toDict():Dictionary
Article
title:String
summary:String
publisher:String
link:String
toDict():Dictionary