Coronapyrus

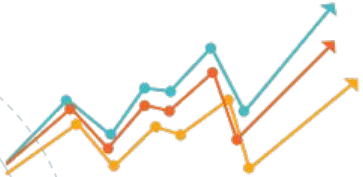
By: Alex Claman, Noah Jaccard, Jake McSweeney

0.

Concept

What are we even doing?

Road to Coronapyrus



Work With COVID Data

Stop COVID Misinformation

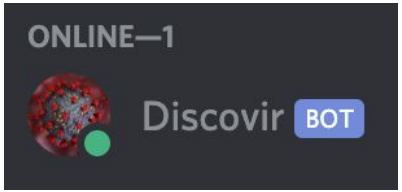
Integrate with Communication Services

Two Part Concept

Coronapyrus - Python package to process and fulfill requests for COVID media coverage or data

```
import coronapyrus
```

Applications - Discord bot/Slack app for interaction with users of various communication services



Concept Images

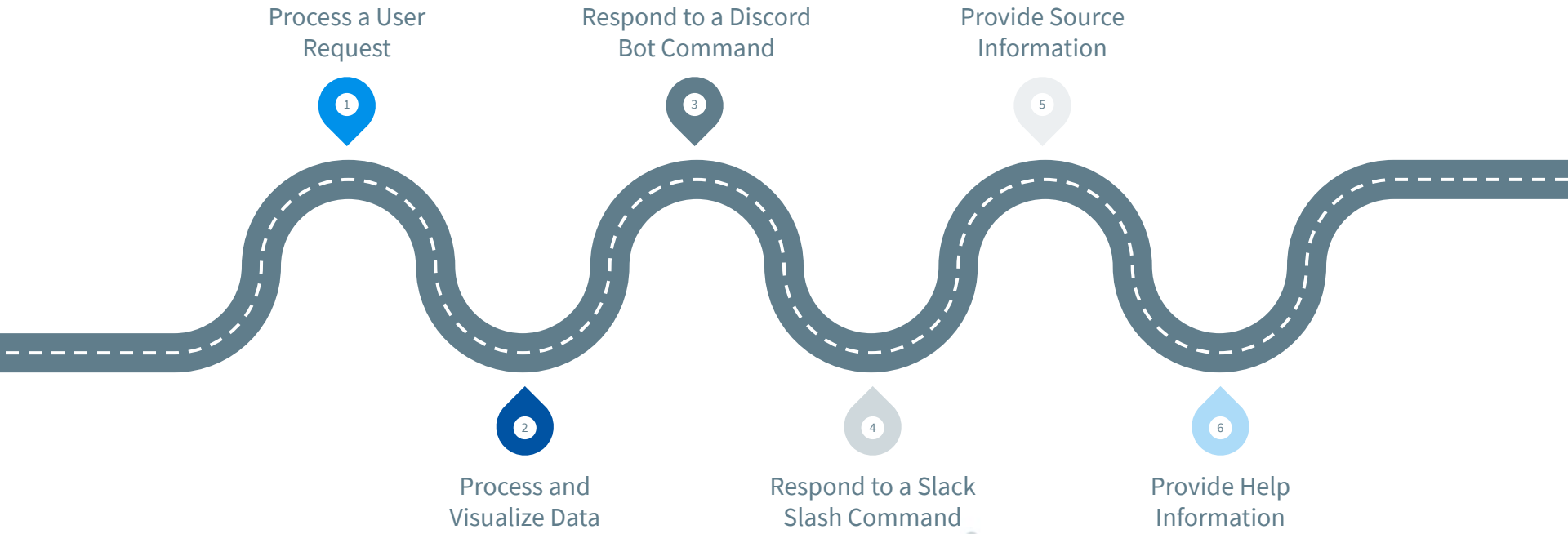
```python
# Imports

import os
import discord
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from discord.ext import commands
from matplotlib.ticker import FixedLocator

# Package Parameter Alterations

plt.rcParams['savefig.dpi'] = 300

# Bot Setup

description = "A bot which provides either COVID media coverage or "
description += "graphical/tabular COVID data upon request."
command_prefix = '!'
bot = commands.Bot(command_prefix=command_prefix, description=description)
# Rewrote predefined help function
bot.remove_command('help')

# Basic Bot Functionality

@bot.event
async def on_ready():
    '''Print bot credentials to CLI upon login'''
    print('Bot logged in: {user} - {uid}.'.format(user=bot.user.name, uid=bot.user.id))

@bot.command(name="help",
    description="Returns all available commands.",
    aliases=['h'])
async def help(ctx, *args):
    '''Send help information as message upon request.'''
    helptext = "```\n"
    helptext += bot.user.name + ": " + bot.description + "\n\n"
    helptext += "Commands:\n"
    for command in bot.commands:
        if command.aliases:
            aliases_string = f", {command_prefix}".join(command.aliases)
            helptext += f"\t{command_prefix}{command} ({command_prefix}{aliases_string}) - {command.description}\n"
```

1.

Functional Requirements

How a user interacts with Coronapyrus

Functional Requirements



Functional Requirement 1

Process A User Request

- A user request must be parsed.
- All user defined information must be retrieved, processed, visualized, and returned.
- Generalized request for Covid information from source.

- Defines scope and format parameters.

Functional Requirement 2

Process and Visualize Data

- The data needs to be processed.
- Depending on the format defined by the user, the data will be visualized as such.
- This could be a message, graph, data table, etc.

- Pandas returns a data table, Matplotlib does the graphs and charts.

Functional Requirement 3 and 4

Respond to a Discord/ Slack Command

- User issues a command to the bot with parameters dictating the scope and format of the request.
- The command interacts with the bot
- The relevant information is then displayed

Functional Requirement 5

Provide Source information

- Coronapyrus will be able to tell users where the information was pulled from.
- Our default place to access data from is the John Hopkins University Covid Database

Functional Requirement 6

Provide Help information

- Helps users understand how to use the bot
- Shows relevant commands
- Provide a description of what the bot does

2.

Nonfunctional Requirements

Behind the scene mechanics

Nonfunctional Requirements

Retrieve COVID Info

Once a user request is made, relevant COVID information must be gathered from reliable sources.

Define User Scope

A data structure designed such that a user can properly define their scope.

Define User Format

A data structure designed so the user can properly define their format.

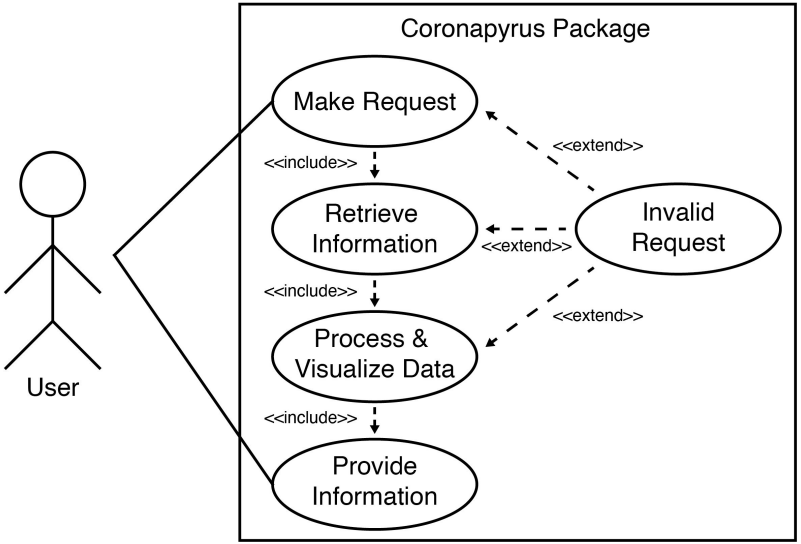
3.

Use Cases

How might Coronapyrus be used?
How will we implement an app?

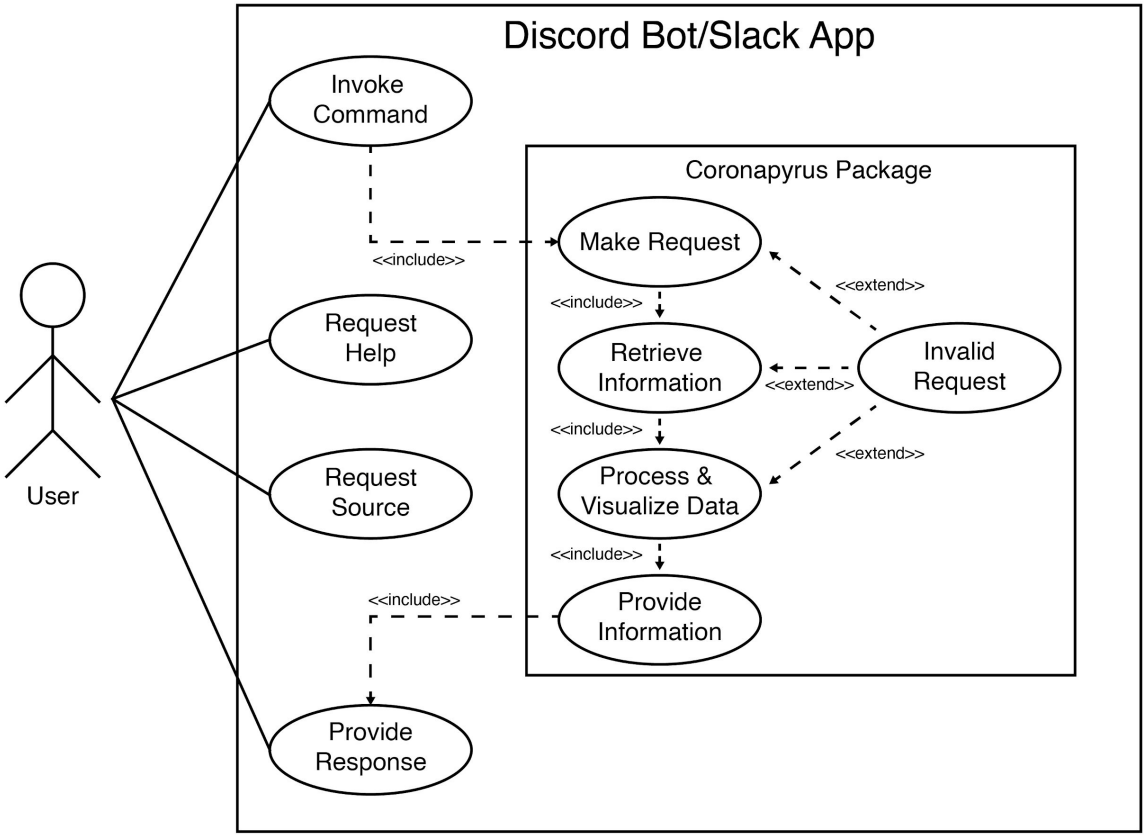
Coronapyrus Use Case:

Respond to User Request



App Use Case:

Respond to User Command



4.

Constraints

Defining the six types of constraints

1. Tool Constraints

Required Python Packages

The Pandas, Matplotlib, and newsfetch packages must be installed for Coronapyrus to function properly.

Data Availability

Any data used will be retrieved from John Hopkins University databases. If this data is unavailable, a different source of data will be used.

Source:https://github.com/CSSEGISandData/COVID-19/tree/master/csse_covid_19_data

2. Language Constraints

Python Constraint

The only language allowed is Python.

3. Platform Constraints

Python Package Management Platform

Independent of the operating system, a Python package manager is required.

4. Network Constraints

Request Covid Information

A proper internet connection is required.

5. Deployment Constraints

Python Environment

It will be deployable in any Python development environment. A Python distribution such as Anaconda is required to create applications or scripts using Coronapyrus.

6. Budget and Schedule Constraints

Time Constraint

The final deadline is May 1st 2021

Budget Constraint

There is no budget tied to the project. It is also an open source project

Thanks!

Any questions?