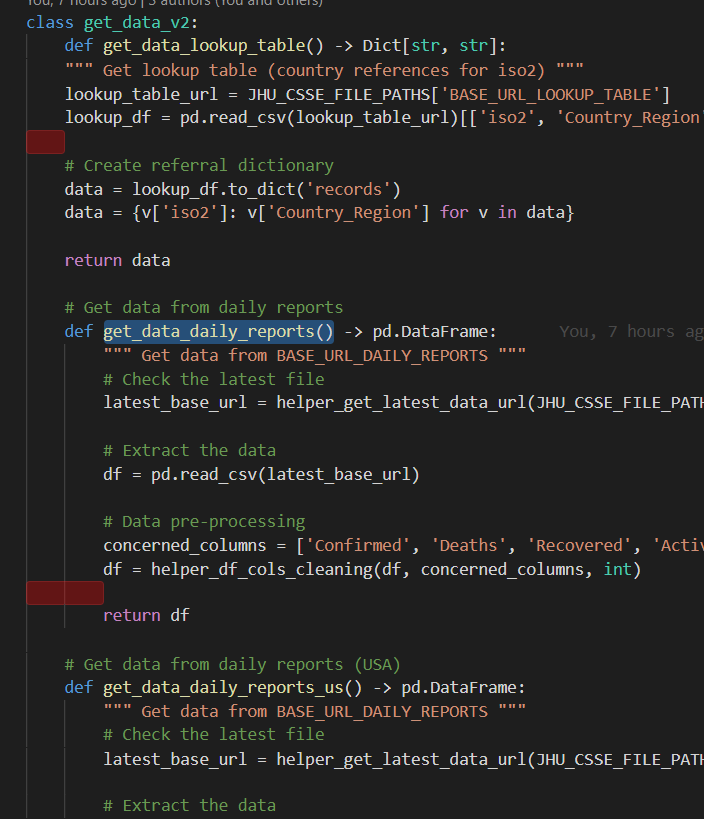
|  |  |
| --- | --- |
| **Student Name** | Kanisha Patel |
| **Student ID** | 216006595 |
| **GitHub User ID** | Kanisha14 |
| **GitHub Email** | Kanishapatel14@gmail.com |
| **Chosen Project** | Proj\_2: COVID19 API |
| **Project Repository url** | <https://github.com/Kanisha14/covid19-api> |
| **First Pull Request Number** | #143, branchName = “Versions\_intoClass” |
| **Second Pull Request Number** | #146 branchName = “Integrator\_docFixes” |

Aggregate pattern in Domain-Driven Design usually refers the several similar objects grouped together that can be treated as one unit for the purpose of the data changes.

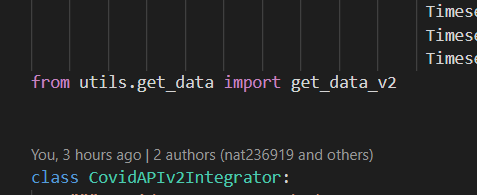
The **first pull request**, “Wrapper Class for V1 & V2” makes the changes in the file *get\_data.py*. It have several methods like get\_data\_daily\_reports(), get\_data\_time\_series(), get\_data() for our API version 1 and version 2. Two wrapper class get\_data\_v1 and get\_data\_v2 which groups the methods of version 1 in one class and version two in different classes, respectively. The aggregates are formed when classes form a “Has-a” relationship with its method.

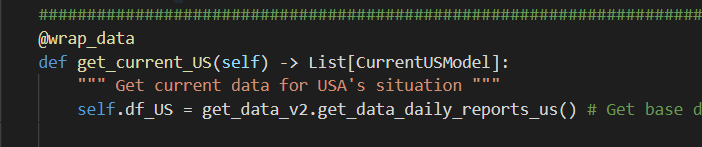
 

Hence, get\_data\_v1 has-a method get\_data() that functions for the API version 1. Similarly, get\_data\_v2 has-a methods like get\_data\_lookup\_table(), get\_data\_daily\_reports() etc.

Each Aggregates has a root, i.e in our case will be our two classes, get\_data\_v1 and get\_data\_v2 and it has a boundary i.e all the methods inside each class.

Furthermore, the pull request also makes changes in the file *covid\_api\_v2\_Integrator.py* that initially imported few of the methods for version 2, but now import the class get\_data\_v2. Hence it also further makes changes to all the place where the method is called. Now it calls the same method but via our class for version 2.





Similar changes need to be made in *covid\_api\_v1\_Integrator.py* file for version 1.

The name get\_data\_v1 and get\_data\_v2 comes from the filename get\_data.py and v1, v2 represents version 1 and version 2 respectively. Since its easier to understand what the class i.e., our root represents and what its functions are.

The **Second pull request**, “Separate Wrapper Classes for method in Integrator file” makes changes in the file *covid\_api\_v2\_Integrator.py.* this file initially had several methods like get\_country(), get\_current(), get\_current\_US(), get\_confirmed(), get\_deaths() and get\_time\_series() and many more. The pull request made group all the method with similar functionality into one class, hence forming a cluster of objects that can be used as single unit. Hence the classes being the root of the aggregate pattern formed and its respective methods being the boundary of that aggregate. Below table shows all the classes and its subsequent methods.

|  |  |  |
| --- | --- | --- |
| Roots | Boundary | Reason for the Name |
| CovidAPIv2Integrator | **def \_\_init\_\_(self)**  **def wrap\_data(func)**  **get\_country()** | **Class name is similar to the file name as it is the main class of the file.** |
| CovidAPIv2Integrator\_current | **get\_current()**  **get\_current\_US()** | **This class groups all the methods that pulls the current data. Hence the current name appears on the end.** |
| CovidAPIv2Integrator\_Status | **get\_confirmed()**  **get\_deaths()**  **get\_recovered()**  **get\_active()**  **get\_total()** | **Similarly, this class get the total STATUS by summing all the cases together for respective fields. Hence status is added at the end.** |
| CovidAPIv2Integrator\_timeseries | **get\_time\_series()**  **\_\_extract\_time\_series()**  **\_\_extract\_time\_series\_global()**  **get\_US\_time\_series()**  **\_\_extract\_US\_time\_series()** | **This class retrieves the timeseries for different regions and hence the name.** |