…Load the following packages to get started:

```
library(shiny)
library(plotly)
library(COVID19)
```

COVID19

The COVID19 R package provides a seamless imtegration with COVID-19 Data Hub via the `covid19()` function. Type `?covid19` for the full list of arguments. Here we are going to use:

- `country`: vector of country names or ISO codes.
- `level`: granularity level; data by (1) country, (2) region, (3) city.
- `start`: the start date of the period of interest.
- `end`: the end date of the period of interest.

Define UI

Define the following inputs…

- `country`: the country name. Note that the options are automatically populated using the `covid19()` function.
- `type`: the metric to use. One of `c("confirmed", "tests", "recovered", "deaths")`, but many others are avaibale. See here for the full list.
- `level`: granularity level (country – region – city).
- `date`: start and end dates.

…and the output:

- `covid19plot`: plotly output that will render an interactive plot.

Wrap everything into a `fluidPage`:

```
# Define UI for application
ui <- fluidPage(

    selectInput("country", label = "Country", multiple = TRUE, choices = 
unique(covid19()$administrative_area_level_1), selected = "Italy"),
    selectInput("type", label = "type", choices = c("confirmed", "tests", 
"recovered", "deaths")),
    selectInput("level", label = "Granularity", choices = c("Country" = 1, 
"Region" = 2, "City" = 3), selected = 2),
    dateRangeInput("date", label = "Date", start = "2020-01-01"),

    plotlyOutput("covid19plot")

)
```

Server logic

After defining the reactive inputs in the UI, we connect such inputs to the `covid19()` function to fetch the data. The following code snippet shows how to render an interactive plot(ly) that automatically updates when any of the input is changed. Note that the `covid19()` function uses an internal **memory caching** system so that the data are never downloaded twice. Calling the function multiple times is highly efficient and user friendly.

```
# Define server logic
```

```
server <- function(input, output) {

    output$covid19plot <- renderPlotly({
        if(!is.null(input$country)){
            x <- covid19(country = input$country, level = input$level, start =
input$date[1], end = input$date[2])
            color <- paste0("administrative_area_level_", input$level)
            plot_ly(x = x[["date"]], y = x[[input$type]], color = x[[color]])
        }
    })

}
```

Run the application

The sample application is available at
https://guidotti.shinyapps.io/h83h5/

```
# Run the application
shinyApp(ui = ui, server = server)
```