## **Step 1: Identify domain logic**

```
server <- function(input, output, session) {</pre>
 downloads <- reactive({</pre>
   cranlogs::cran downloads(
     input$package,
     from = Sys.Date() - 365,
     to = Sys.Date()
 })
downloads rolling <- reactive({</pre>
   validate(need(sum(downloads()$count) > 0, "Input a valid package name"))
   downloads() %>%
     mutate(count = zoo::rollapply(count, 7, mean, fill = "extend"))
 } )
 output$plot <- renderPlot({</pre>
   ggplot(downloads rolling(), aes(date, count)) + geom line()
 })
```

## **Step 1: Identify domain logic**

```
server <- function(input, output, session) {</pre>
 downloads <- reactive({</pre>
   cranlogs::cran downloads(
     input$package,
     from = Sys.Date() - 365,
     to = Sys.Date()
                                              Only applies to Shiny,
                                                 don't export it!
 })
 downloads rolling <- reactive({</pre>
   validate(need(sum(downloads()$count) > 0, "Input a valid package name"))
   downloads() %>%
     mutate(count = zoo::rollapply(count, 7, mean, fill = "extend"))
 })
 output$plot <- renderPlot({</pre>
   ggplot(downloads rolling(), aes(date, count)) + geom line()
 })
```