

**RConnect – Steps for how to publish WebApp**

**<Version No. 1.0>**

|  |  |  |  |
| --- | --- | --- | --- |
| Document Revision History: | | | |
| Document Version Number | Document Revision Date | Written By | Change Summary  (Reference section[s] changed) |
| 1.0 | 07/31/2020 | CDL Platform team | DRAFT |
| 1.1 | 09/25/2020 | CDL Platform team | Updated Document |
| 1.2 |  |  |  |

Contents

[1.0 Introduction 4](#_Toc51942341)

[1.1. Purpose 4](#_Toc51942342)

[1.2. Constraints 4](#_Toc51942343)

[1.3. Acronyms and Definitions 4](#_Toc51942344)

[1.4. References 4](#_Toc51942345)

[2.0 High level architecture 5](#_Toc51942346)

[3.0 Pre-requisites & Steps 5](#_Toc51942347)

[4.0 Using Libraries from Rstudio package manager 12](#_Toc51942348)

# Introduction

## Purpose

The document covers steps for how to publish the RShiny webapps thru Rconnect in Commercial data lake.

The intended audience of this document is the application development team, IT team, data lake administration team. This document provides the detailed guidelines and steps for how to use RConnect.

## Constraints

RConnect server is not yet setup in CDL Production environment

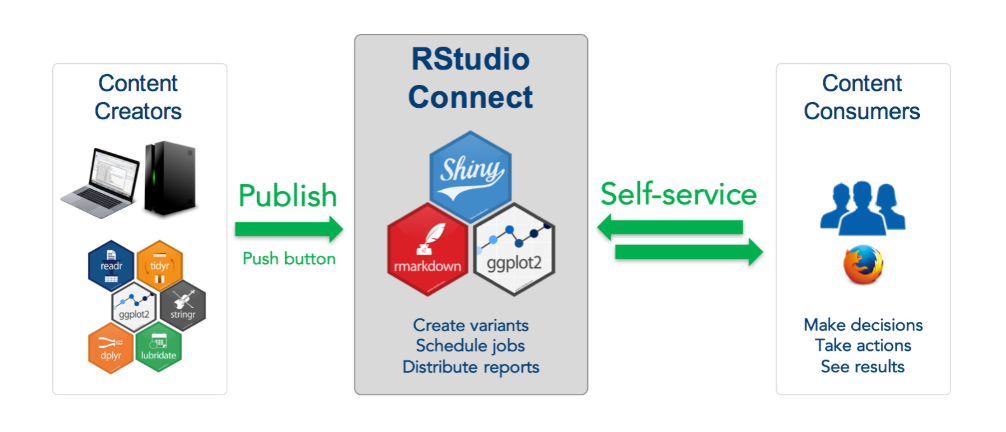
## Acronyms and Definitions

|  |  |
| --- | --- |
| Acronym | Definition |
| AD | Active Directory |
| CDL | Commercial data lake |
|  |  |

## References

| Document Number | Document Title |
| --- | --- |
|  |  |

# High level architecture



# Pre-requisites & Steps

* + 1. **Pre-requisite**

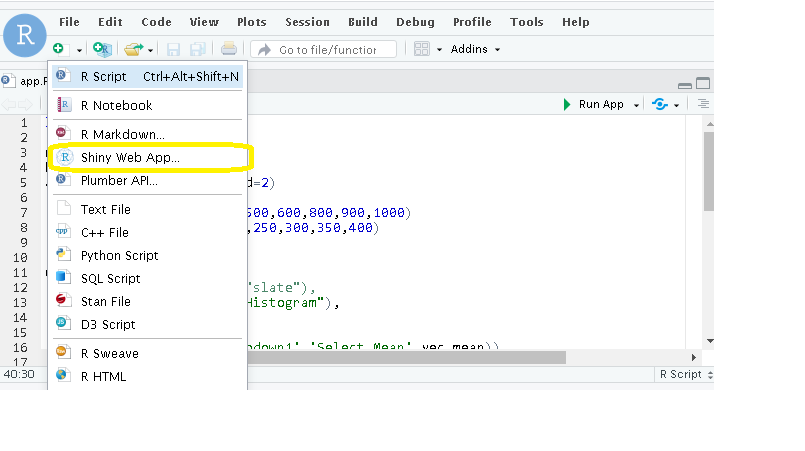
1. User must be added in the VAS AD group
   1. NonProd : AWSApp-cdl-d-hadoop-user
   2. Prod : AWSApp-cdl-p-hadoop-user
2. RStudio Server & RConnect server must be installed and running
3. User must be able to access RStudio Server
   1. NonProd : <https://en01-cop-nonprod.abbvienet.com:8787/>
   2. Prod : <https://analytics-commercial-p.abbvienet.com:8787/>
4. User must be able to access RConnect server
   1. NonProd : <https://en01-cop-nonprod.abbvienet.com:3939/>
   2. Prod : <https://analytics-commercial-p.abbvienet.com:3939/> (In Progress)
      1. **Steps**
5. Install following R packages if its not available in system library

*install.packages(c("brew","callr","clipr","clisymbols","commonmark","crosstalk","curl","digest","DT","ellipsis","evaluate","fansi","formatR","glue","htmltools","htmlwidgets","ini","knitr","later","praise","prettyunits","processx","promises","ps","purrr","rex","rlang","rmarkdown","tinytex","utf8","uuid","vctrs","withr","xfun","xopen","yaml","acepack","askpass","assertthat","backports","base","base64enc","BH","bindr","bindrcpp","bitops","boot","caTools","checkmate","class","cluster","codetools","colorspace","compiler","config","covr","cronR","curl","data.table","datasets","DBI","dichromat"))*

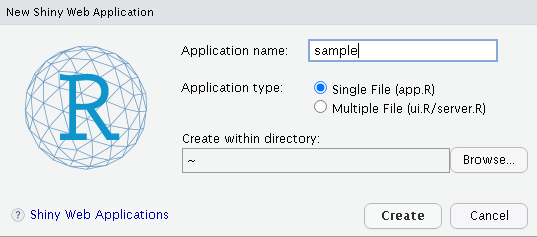
1. Login to Rstudio using the below link

NonProd : <https://en01-cop-nonprod.abbvienet.com:8787/>

1. Select file->+new->



1. Enter the application name and create. Pls refer below example



1. Create a shiny web app. *Below is sample code for reference purpose only*

*Sample Shiny WebApp code :*

*library(shiny)*

*rnorm(500,800,250)*

*hist(rnorm(500,800,250))*

*abline(v=800,col="red",lwd=2)*

*vec\_mean<- c(200,300,400,500,600,800,900,1000)*

*vec\_sd<- c(50,100,150,200,250,300,350,400)*

*ui<- fluidPage(*

*# theme = shinytheme("slate"),*

*headerPanel("RShiny: Histogram"),*

*sidebarPanel(*

*(selectInput('dropdown1','Select Mean',vec\_mean)),*

*(selectInput('dropdown2','Select SD',vec\_sd))*

*),*

*mainPanel(*

*plotOutput("out\_hist",width = "85%")*

*)*

*)*

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

*output$out\_hist<- renderPlot({*

*par(bg="lightskyblue") hist(rnorm(500,as.numeric(input$dropdown1),as.numeric(input$dropdown2)),col="coral",border=F,*

*main=paste0("Histogram: Mean ",input$dropdown1," SD ",input$dropdown2),*

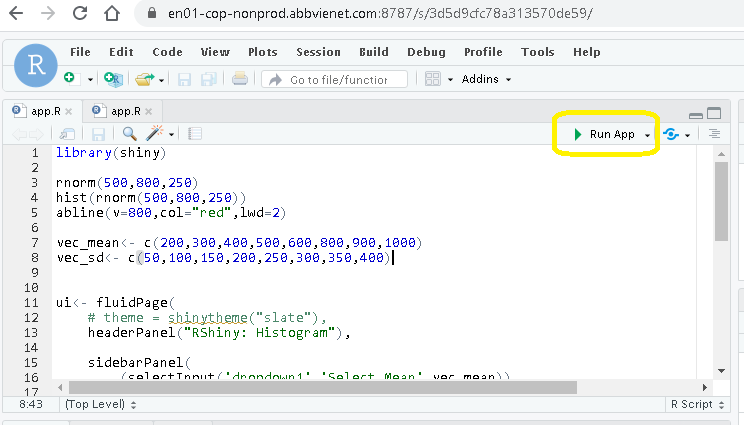
*xlab="Random Sample")*

*})*

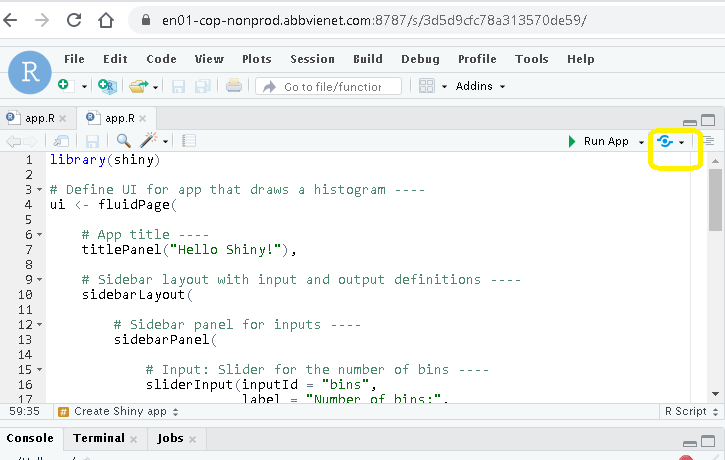
*}*

*shinyApp(ui=ui,server=server)*

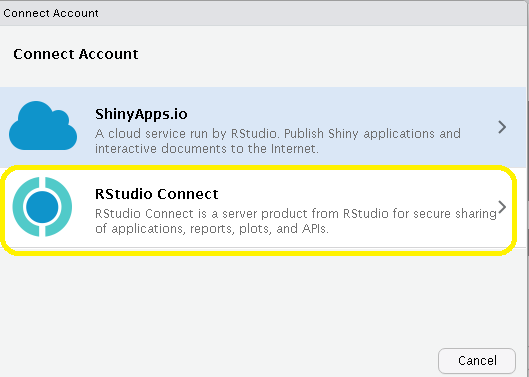
1. Click on the run app



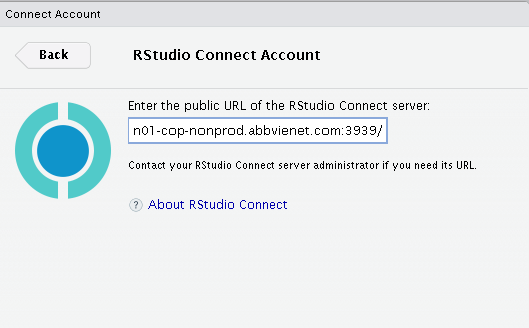
1. Click on publish once the application is run successfully



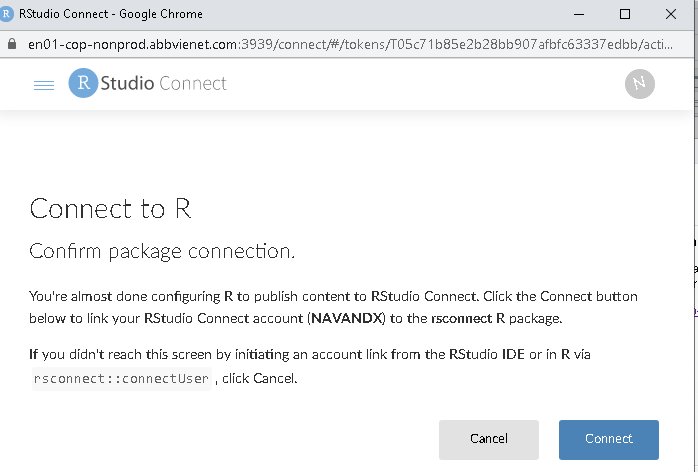
1. Choose RStudio connect



1. Provide Rstudio connect URL to publish the webapp
   * NonProd : <https://en01-cop-nonprod.abbvienet.com:3939/>
   * Prod : <https://analytics-commercial-p.abbvienet.com:3939/> (In Progress)

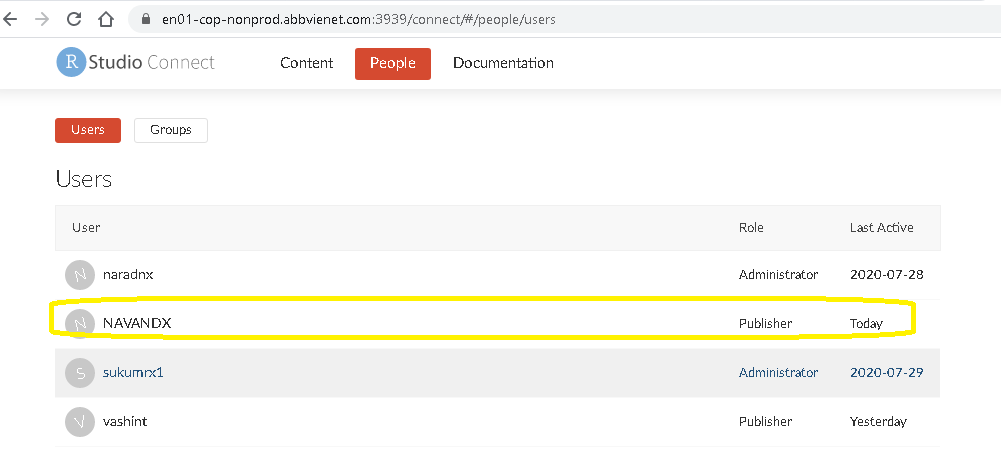


1. It will redirect to authentication page ,click on connect



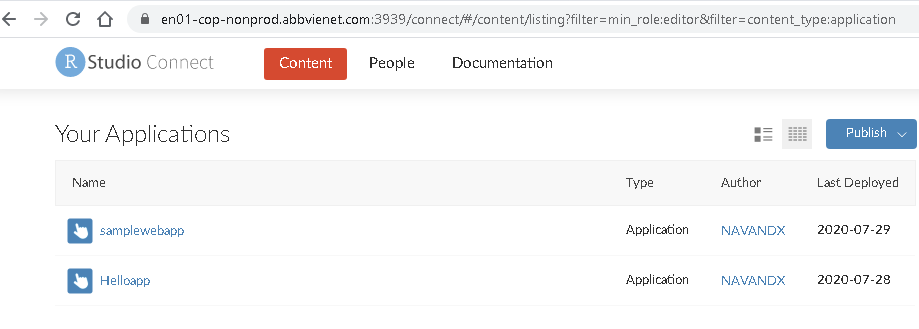
**Note** : User can publish content only if they have publisher access to Rstudio connect

Contact admin team to get publisher access

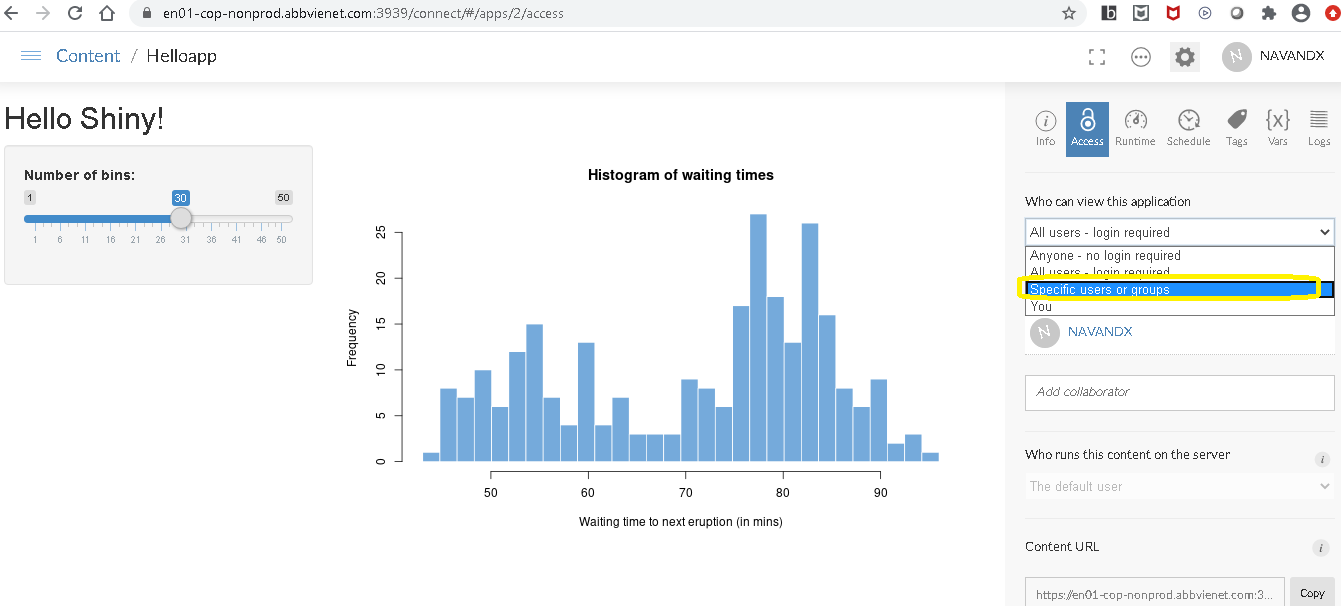


1. Once authenticated we should be able to see the content from Rstudio connect URL

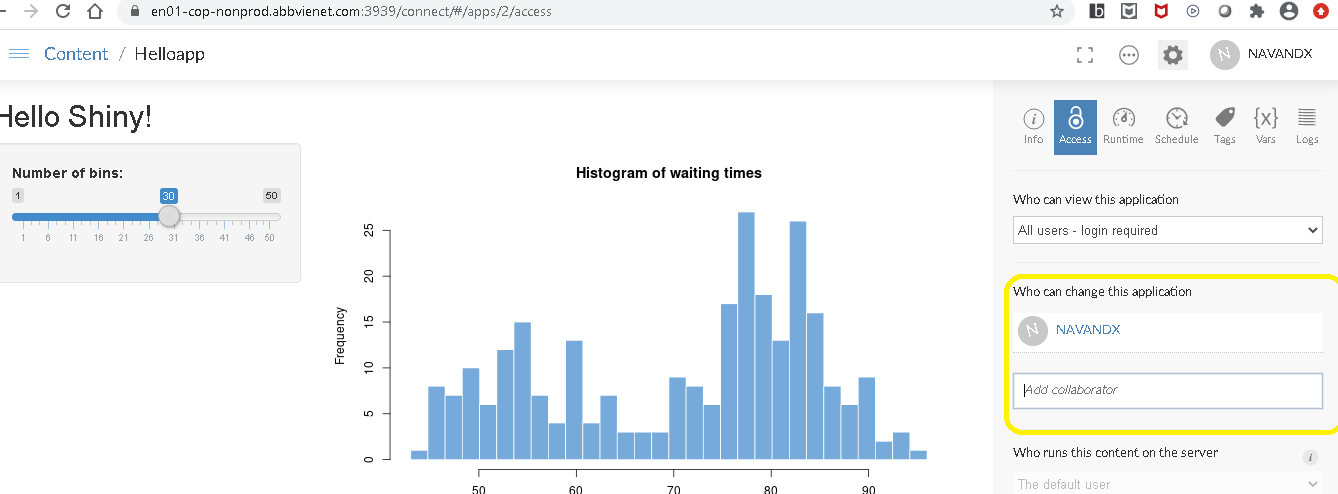
e.g. <https://en01-cop-nonprod.abbvienet.com:3939/connect/#/content/>



1. Select the app and choose access to specific users or groups and add the group who can view the application. (Contact admin team in case to create group, etc)



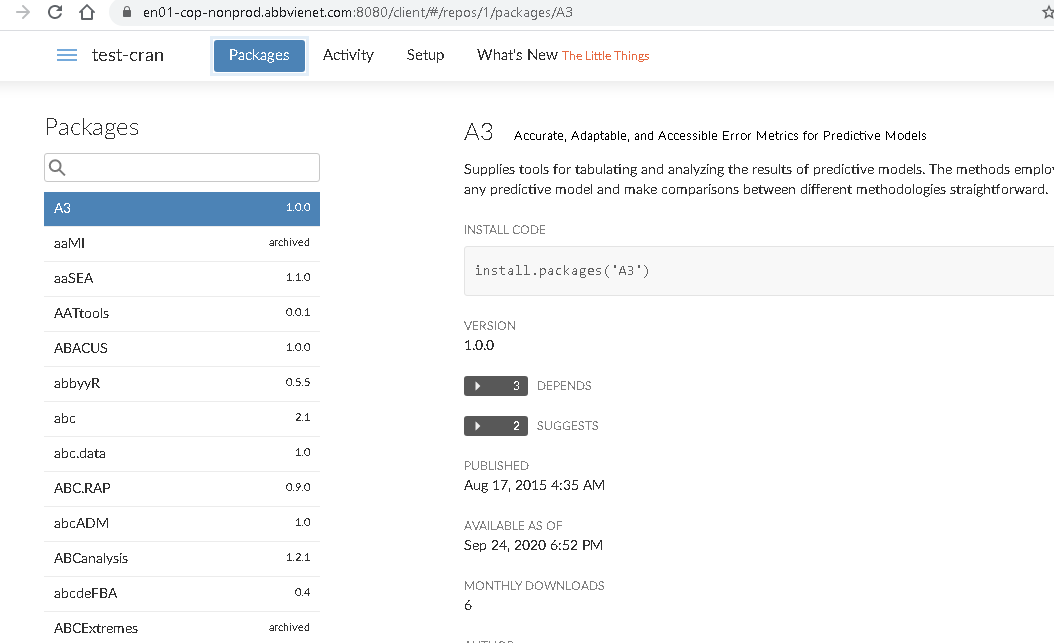
1. To provide access to users who can collaborate to the web application, add their userids in the below column



1. Click on the logs tab to view the detailed logs

# Using Libraries from Rstudio package manager

1. Rstudio pacakage manager can be accessed using the below link <https://en01-cop-nonprod.abbvienet.com:8080/>



1. For POC purpose test repo has been created with the name of “test-cran” and all the default packages are available under this repo
2. While creating a shiny webapp we can utilize the packages which is available from Rstudio package manager by using the “options” parameter

options(repos = c(REPO\_NAME = "https://en01-cop-nonprod.abbvienet.com:8080/test-cran/latest"))

install.packages('A3')

1. Create a new shiny webapp in Rstudio and include the packages which needs to be installed from Rpackage manager

Sample shiny webapp code fro reference :

#

# This is a Shiny web application. You can run the application by clicking

# the 'Run App' button above.

#

# Find out more about building applications with Shiny here:

#

# http://shiny.rstudio.com/

#

library(shiny)

options(repos = c(REPO\_NAME = "https://en01-cop-nonprod.abbvienet.com:8080/test-cran/latest"))

install.packages('A3')

# Define UI for application that draws a histogram

ui <- fluidPage(

# Application title

titlePanel("Old Faithful Geyser Data"),

# Sidebar with a slider input for number of bins

sidebarLayout(

sidebarPanel(

sliderInput("bins",

"Number of bins:",

min = 1,

max = 50,

value = 30)

),

# Show a plot of the generated distribution

mainPanel(

plotOutput("distPlot")

)

)

)

# Define server logic required to draw a histogram

server <- function(input, output) {

output$distPlot <- renderPlot({

# generate bins based on input$bins from ui.R

x <- faithful[, 2]

bins <- seq(min(x), max(x), length.out = input$bins + 1)

# draw the histogram with the specified number of bins

hist(x, breaks = bins, col = 'darkgray', border = 'white')

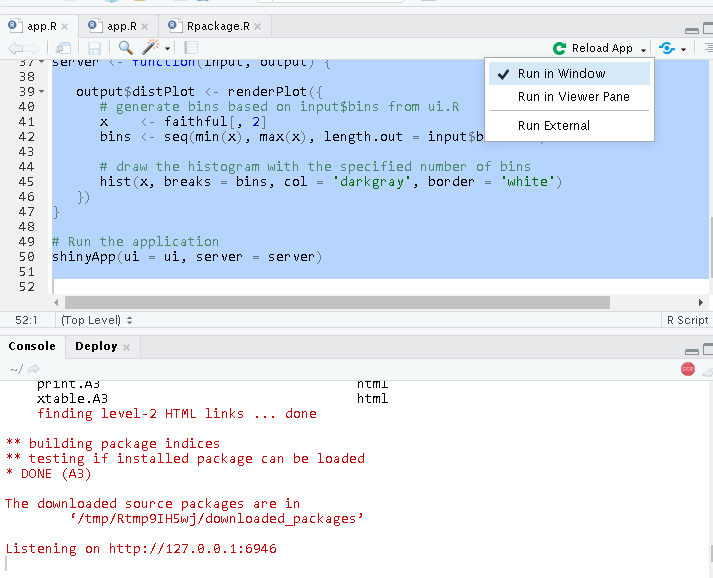
})

}

# Run the application

shinyApp(ui = ui, server = server)

1. Run the application ,In console log you will be able to view the packages getting downloaded from Rpackage manager and installed .



1. Once Run is successful, we can go-ahead and publish the application

