Shiny

Building web applications in R

Subhabroto Dhar, Monash College

September 2018

Melbourne Users of R Network (MelbURN)

Index

R & Shiny

- Shiny is an R package that makes it easy to build interactive web apps **straight from R**
  - You can host standalone apps on a webpage or
  - Embed them in R Markdown documents or
  - Build dashboards
- You can also extend your Shiny apps with CSS themes, htmlwidgets, and JavaScript actions
- R based Back End Server that renders a Front End in Java Script

Demo

Shiny Showcase

- https://www.rstudio.com/products/shiny/shiny-user-showcase/

My apps @ Monash College

- Simple: Sunburst
- Interactive: Collapsible Tree
- Dashboard: Student enrolment forecasting

Code Structure – app.R

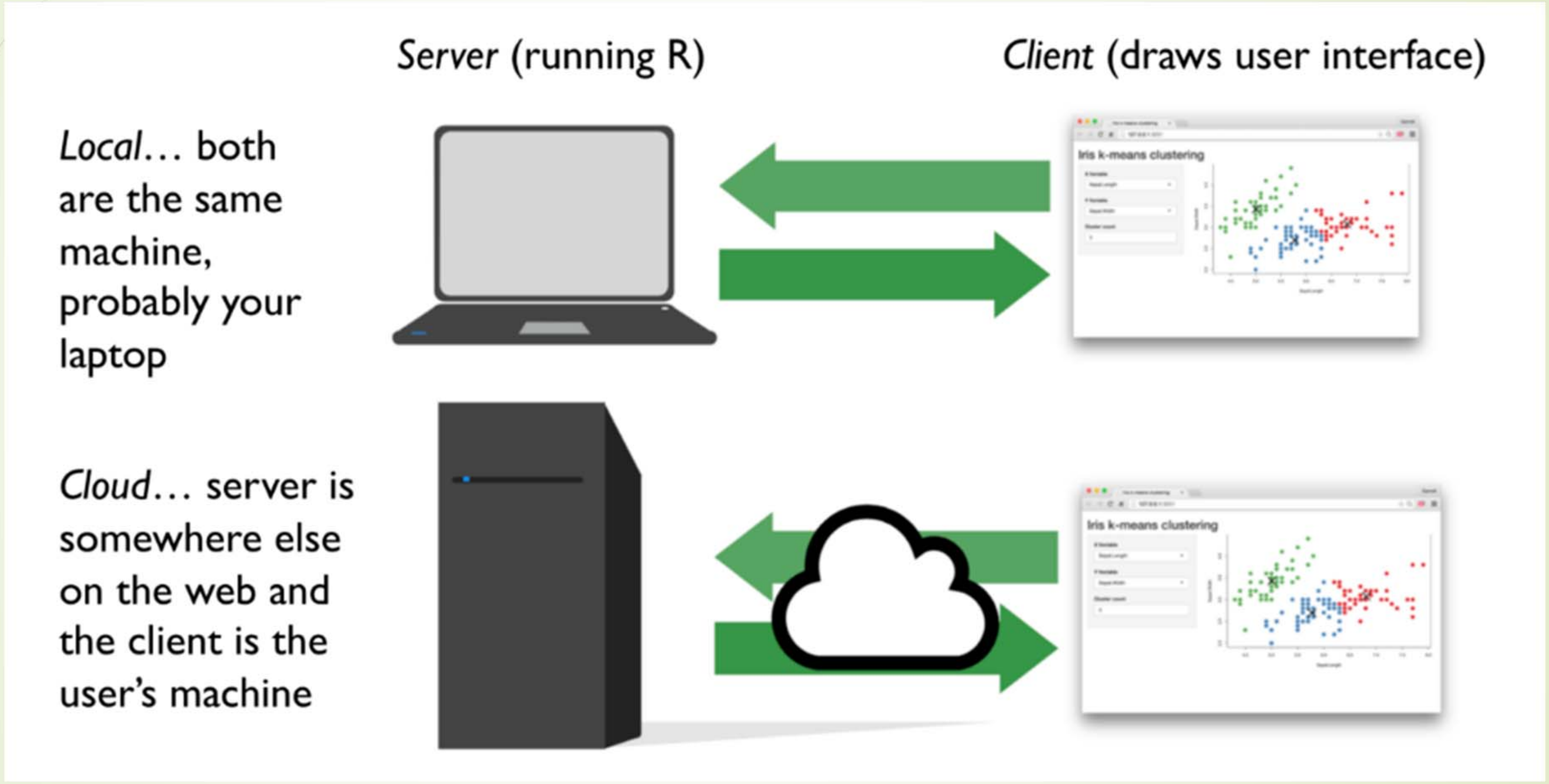
```
library(shiny)

# Create a page with fluid layout
my_ui = fluidPage()

# Create a server function
my_server = function(input, output) { }

# Create a Shiny app object
shinyApp(ui = my_ui, server = my_server)
```

Architecture

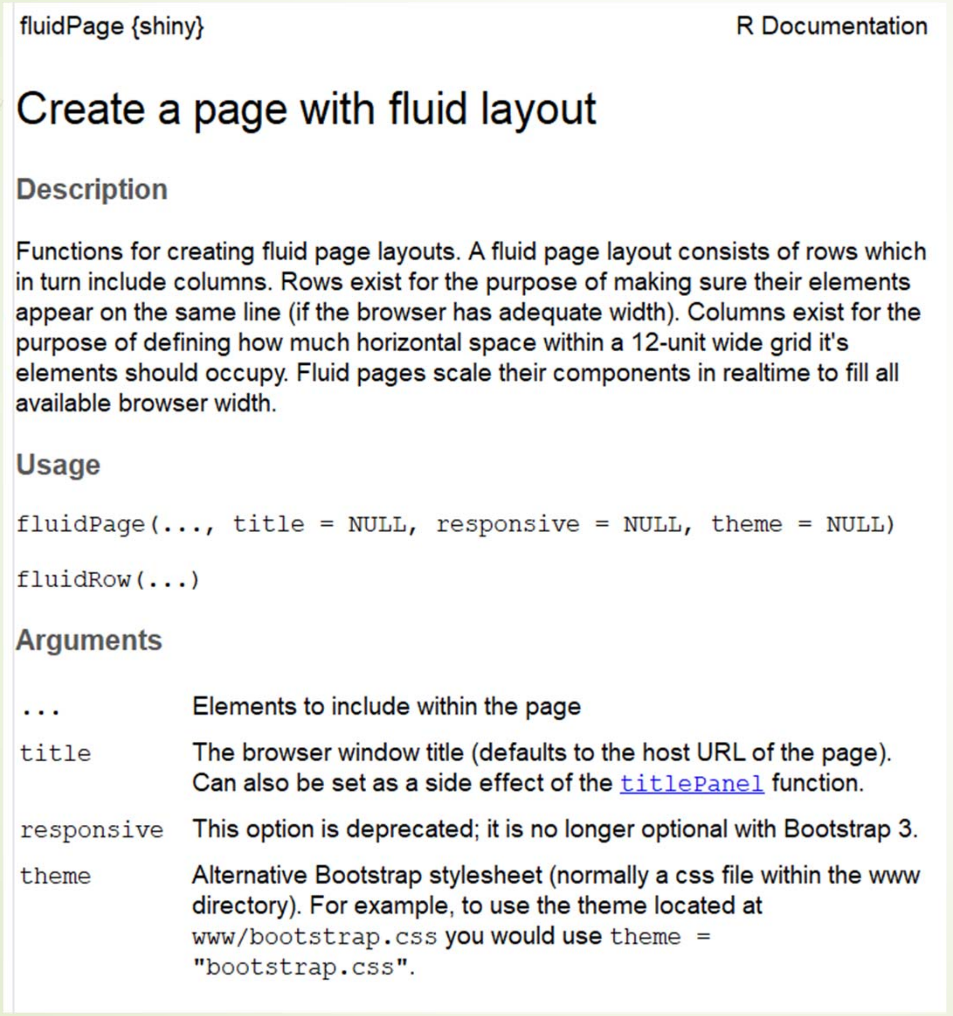


Local… both are the same machine, probably your laptop

Cloud… server is somewhere else on the web and the client is the user's machine

Reference: https://compcogscisydney.org/psyr/shiny.html

UI Layouts



fluidPage {shiny} R Documentation

Create a page with fluid layout

Description

Functions for creating fluid page layouts. A fluid page layout consists of rows which in turn include columns. Rows exist for the purpose of making sure their elements appear on the same line (if the browser has adequate width). Columns exist for the purpose of defining how much horizontal space within a 12-unit wide grid it's elements should occupy. Fluid pages scale their components in realtime to fill all available browser width.

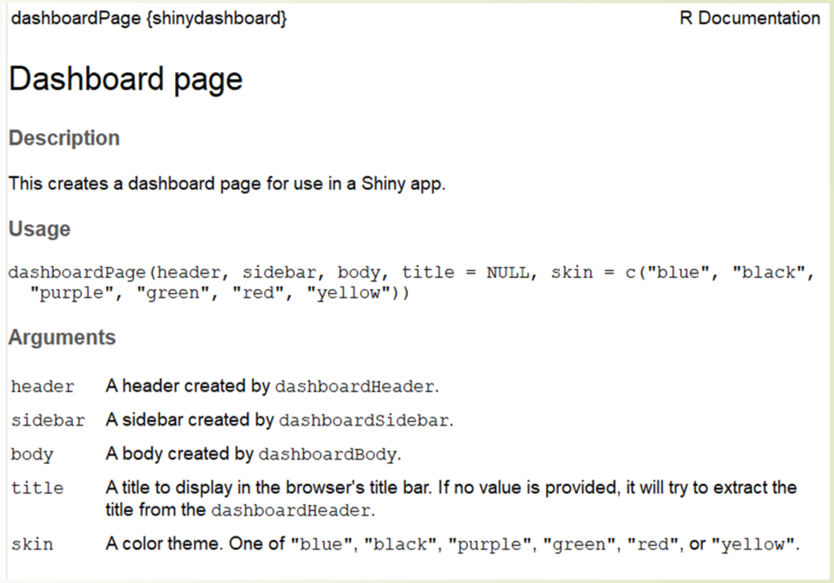
Usage

```
fluidPage(..., title = NULL, responsive = NULL, theme = NULL)

fluidRow(...)
```

Arguments

`...`	Elements to include within the page
`title`	The browser window title (defaults to the host URL of the page). Can also be set as a side effect of the `titlePanel` function.
`responsive`	This option is deprecated; it is no longer optional with Bootstrap 3.
`theme`	Alternative Bootstrap stylesheet (normally a css file within the www directory). For example, to use the theme located at `www/bootstrap.css` you would use `theme = "bootstrap.css"`.



dashboardPage {shinydashboard} R Documentation

Dashboard page

Description

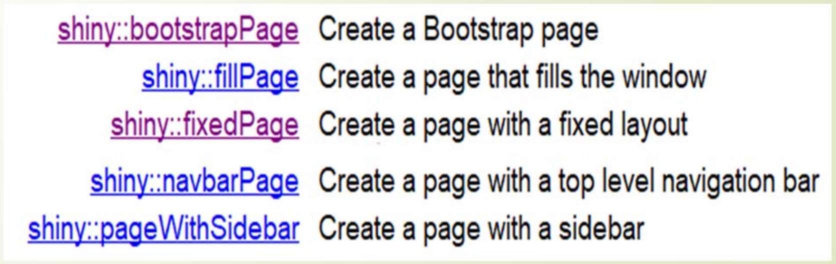
This creates a dashboard page for use in a Shiny app.

Usage

```
dashboardPage(header, sidebar, body, title = NULL, skin = c("blue", "black",
  "purple", "green", "red", "yellow"))
```

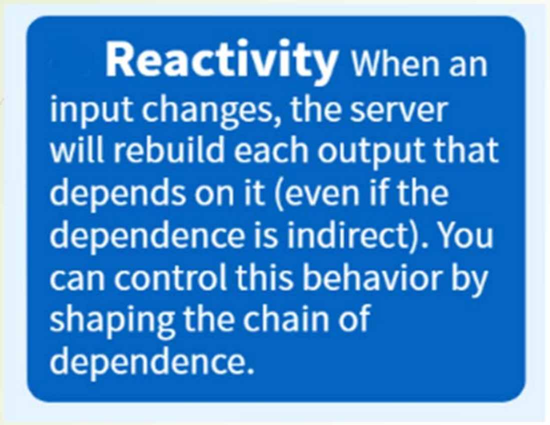
Arguments

`header`	A header created by `dashboardHeader`.
`sidebar`	A sidebar created by `dashboardSidebar`.
`body`	A body created by `dashboardBody`.
`title`	A title to display in the browser's title bar. If no value is provided, it will try to extract the title from the `dashboardHeader`.
`skin`	A color theme. One of `"blue"`, `"black"`, `"purple"`, `"green"`, `"red"`, or `"yellow"`.

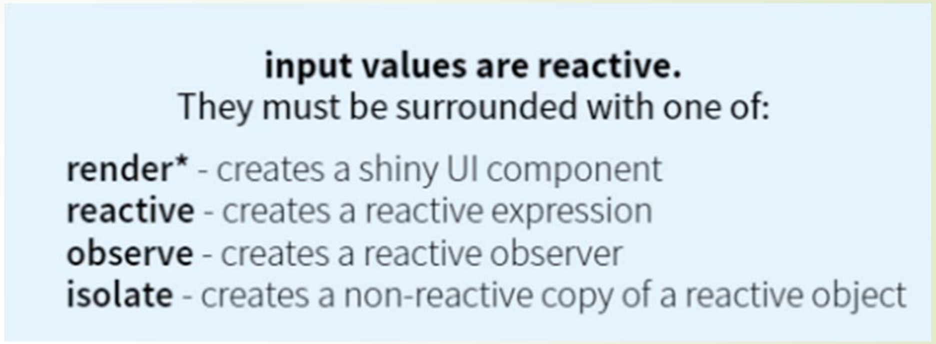


shiny::bootstrapPage	Create a Bootstrap page
shiny::fillPage	Create a page that fills the window
shiny::fixedPage	Create a page with a fixed layout
shiny::navbarPage	Create a page with a top level navigation bar
shiny::pageWithSidebar	Create a page with a sidebar

Reactivity

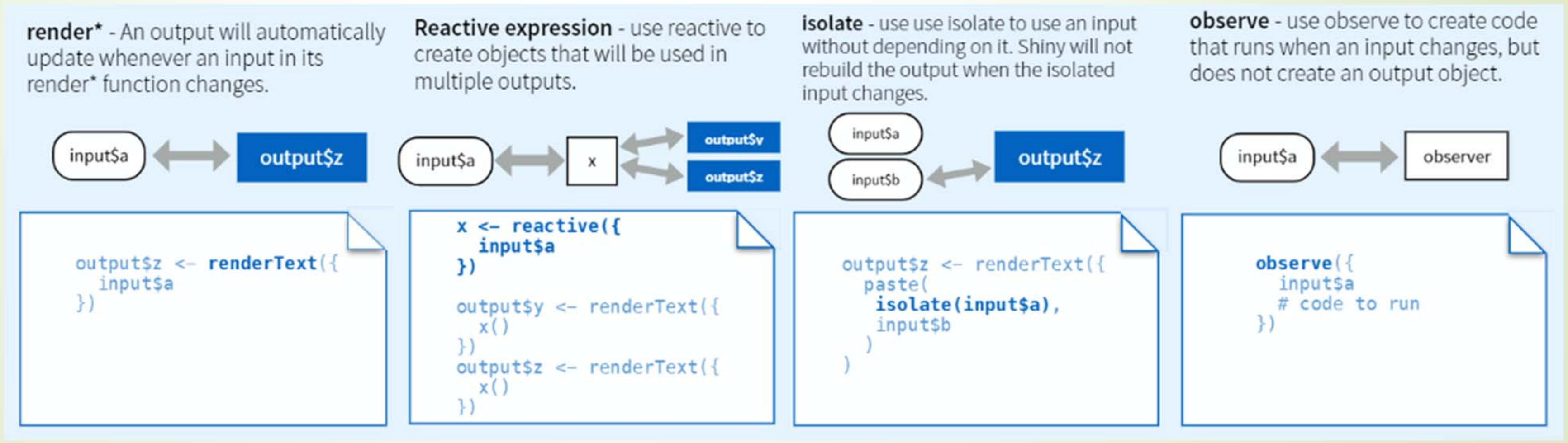


Reactivity When an input changes, the server will rebuild each output that depends on it (even if the dependence is indirect). You can control this behavior by shaping the chain of dependence.



input values are reactive.
They must be surrounded with one of:

render* - creates a shiny UI component
reactive - creates a reactive expression
observe - creates a reactive observer
isolate - creates a non-reactive copy of a reactive object



render* - An output will automatically update whenever an input in its render* function changes.

```
output$z <- renderText({
  input$a
})
```

Reactive expression - use reactive to create objects that will be used in multiple outputs.

```
x <- reactive({
  input$a
})

output$y <- renderText({
  x()
})
output$z <- renderText({
  x()
})
```

isolate - use use isolate to use an input without depending on it. Shiny will not rebuild the output when the isolated input changes.

```
output$z <- renderText({
  paste(
    isolate(input$a),
    input$b
  )
)
```

observe - use observe to create code that runs when an input changes, but does not create an output object.

```
observe({
  input$a
  # code to run
})
```

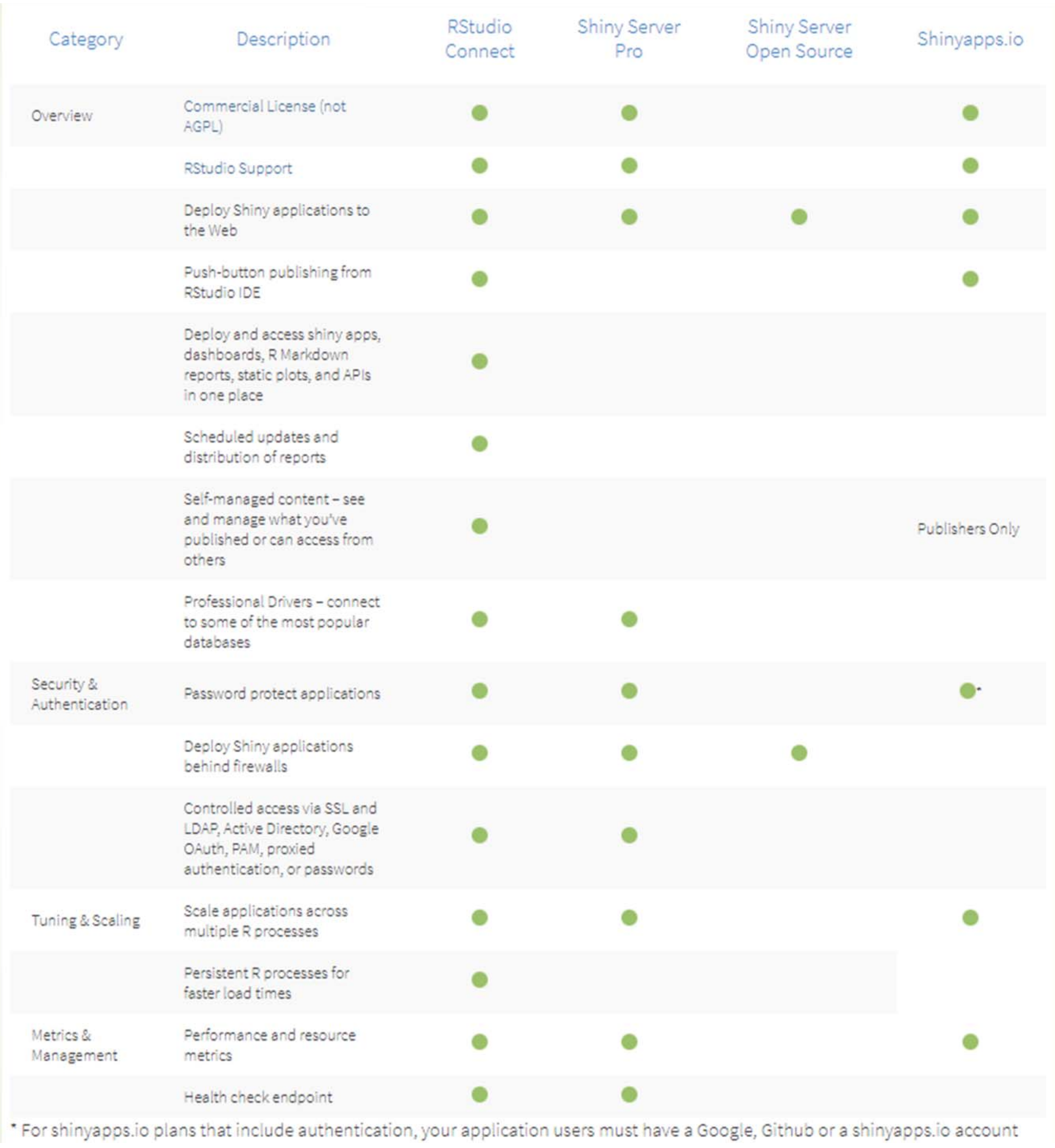
Reference: https://github.com/chendaniely/2015-04-15-SPDC-shiny/blob/master/docs/shiny_cheatsheet.pdf

Code walk through

- https://shiny.rstudio.com/gallery/tabsets.html

Deployment options

Reference: https://www.rstudio.com/products/shiny/shiny-server/



Category	Description	RStudio Connect	Shiny Server Pro	Shiny Server Open Source	Shinyapps.io
Overview	Commercial License (not AGPL)	●	●		●
	RStudio Support	●	●		●
	Deploy Shiny applications to the Web	●	●	●	●
	Push-button publishing from RStudio IDE	●			●
	Deploy and access shiny apps, dashboards, R Markdown reports, static plots, and APIs in one place	●			
	Scheduled updates and distribution of reports	●			
	Self-managed content – see and manage what you've published or can access from others	●			Publishers Only
	Professional Drivers – connect to some of the most popular databases	●	●		
Security & Authentication	Password protect applications	●	●		●*
	Deploy Shiny applications behind firewalls	●	●	●	
	Controlled access via SSL and LDAP, Active Directory, Google OAuth, PAM, proxied authentication, or passwords	●	●		
Tuning & Scaling	Scale applications across multiple R processes	●	●		●
	Persistent R processes for faster load times	●			
Metrics & Management	Performance and resource metrics	●	●		●
	Health check endpoint	●	●		

* For shinyapps.io plans that include authentication, your application users must have a Google, Github or a shinyapps.io account

Deploying with Shiny Server Open Source

- Fetch the rpm

```
[user]# wget
https://download3.rstudio.org/centos5.9/x86_64/shiny-server-
1.5.4.869-rh5-x86_64.rpm
```

- Install the rpm

```
[user]# yum install shiny-server-1.5.4.869-rh5-x86_64.rpm
```

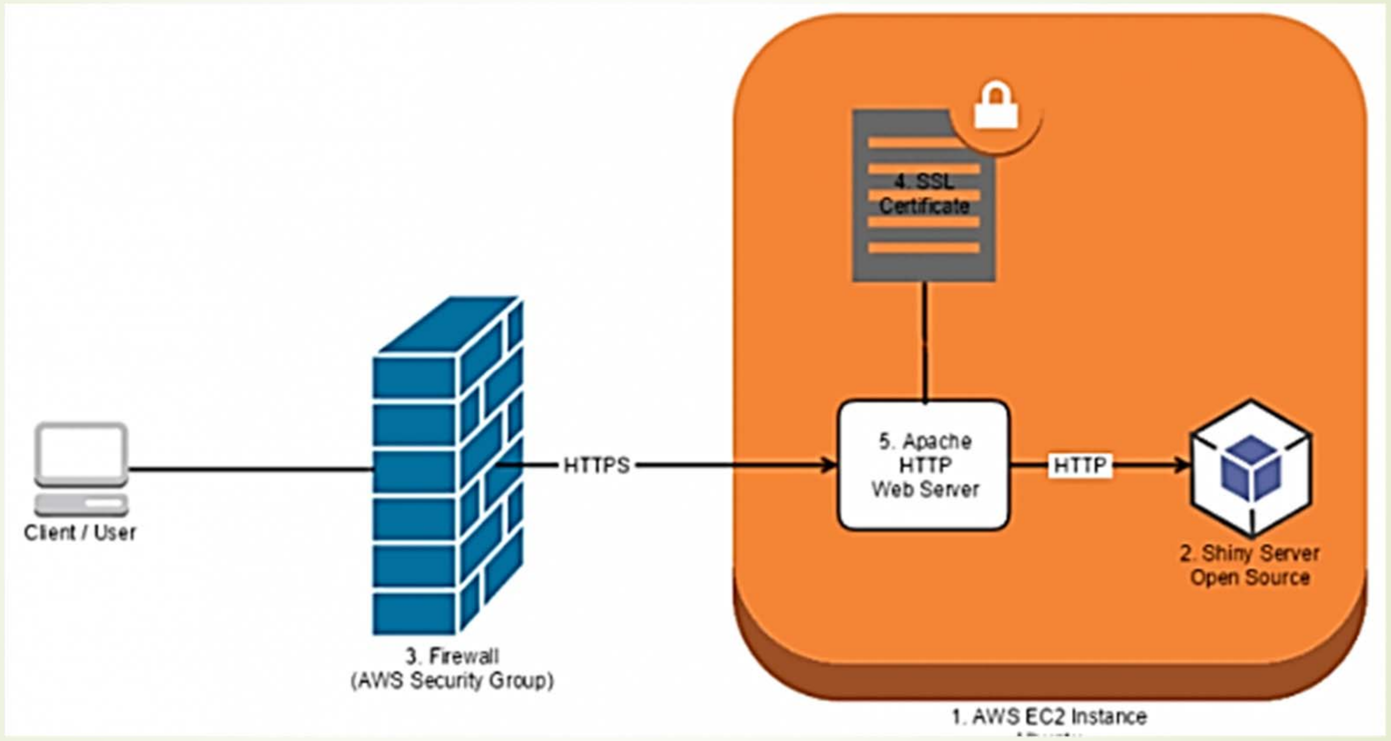
- Configure the port number on which shiny server will listen

```
[user]# vi /opt/shiny-server/config/default.config
```

- Start shiny-server

```
[user]# systemctl start shiny-server
```

Securing – Shiny server open source



Client / User
3. Firewall (AWS Security Group)
HTTPS
4. SSL Certificate
5. Apache HTTP Web Server
HTTP
2. Shiny Server Open Source
1. AWS EC2 Instance

Reference: https://ipub.com/shiny-https/

Resources

- Basic parts of a Shiny app
  https://shiny.rstudio.com/articles/basics.html
- Tutorial
  https://shiny.rstudio.com/tutorial/
- UI Layout guide
  https://shiny.rstudio.com/articles/layout-guide.html
- Reactivity overview
  https://shiny.rstudio.com/articles/reactivity-overview.html
- Cheat sheet
  https://shiny.rstudio.com/images/shiny-cheatsheet.pdf
- Advanced Shiny tips
  https://deanattali.com/blog/advanced-shiny-tips/
- 2016 Shiny Developer Conference Videos
  https://www.rstudio.com/resources/webinars/shiny-developer-conference/