Analytic Health.

Optimizing Shiny for enterprise-grade apps

What R we going to talk about?

1. Introduction
2. Before you start coding
3. Writing code like a pro
4. Infrastructure
5. Optimizing Shiny
6. Testing your app
7. Monitoring
    1. Status
    2. Usage

© Analytic Health 2022

2011: Bsc Business Economics
2015: Msc Data Science

Tilburg University

Eneco
2016: Data Scientist

Dr.Reddy's
2018: Data Scientist

HYPEBRIGHT
2019: Data Science consultant

2021: Managing Director & Head of Data Science

Analytic Health.

Shiny JS VR

Analytic Health – who are we?

We develop intelligent and accessible technology which gives organisations the tools they need to accelerate innovation in healthcare.

Pharmly Analytics

Pharmly Cloud Data

https://analytichealth.co.uk

Veerle van Leemput
Managing Director &
Head of Data Science

Greg Mills
Managing Director &
Founder

Jana Kopecna
Head of Operations

Enterprise-grade app

What qualifies as an enterprise-grade app?

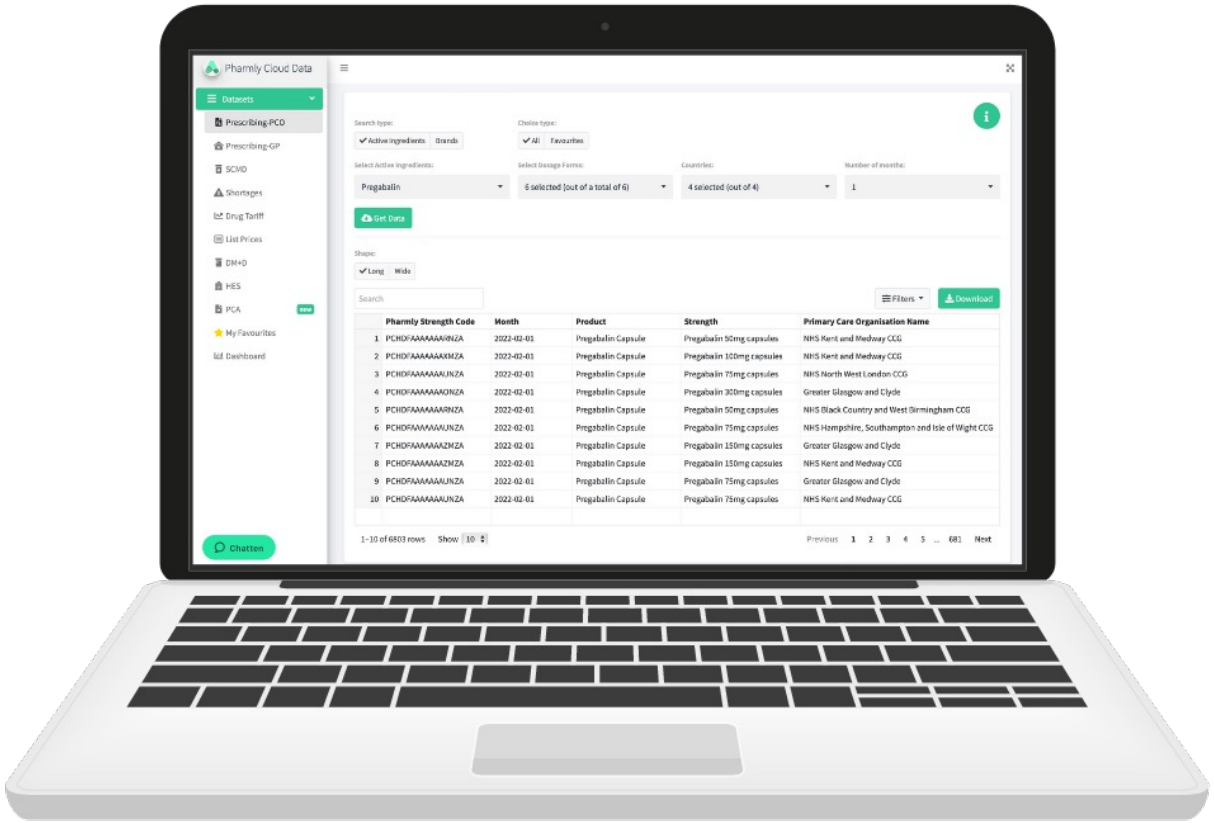
What does it mean to us?

- Subscription based, B2B
- Access control & Data security
- High availability, speed & data validation demands
- Outstanding UX
- Support

Enterprise-grade has nothing to do with the number of users!

© Analytic Health 2022

Pharmly Cloud Data



© Analytic Health 2022

UI first & UX

Prototype

© Analytic Health 2022

Prototype

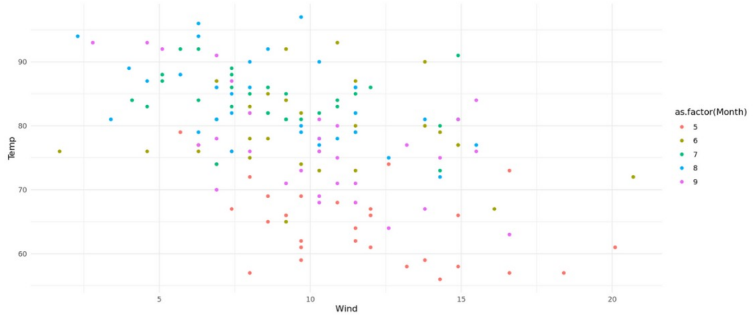
```r
library(shiny)
library(shinipsum)
library(DT)

ui <- fluidPage(

    h2("Welcome to my prototype"),
    textOutput("text"),

    h2("A random ggplot"),
    plotOutput("plot"),

    h2("A random data table"),
    DTOutput("data_table"),

)

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

    output$text <- renderText({
        random_text(nwords = 150)
    })

    output$plot <- renderPlot({
        random_ggplot(type = "point")
    })

    output$data_table <- renderDT({
        random_DT(15, 7)
    })

}

shinyApp(ui, server)
```

Open in Browser Publish

Welcome to my prototype

Lorem ipsum dolor sit amet consectetur adipiscing elit Quisque tellus urna placerat in tortor ac imperdiet sollicitudin mi Integer vel dolor mollis feugiat sem eu porttitor elit Sed aliquam urna sed placerat euismod In risus sem ornare nec malesuada eu ornare quis dui Nunc finibus fermentum sollicitudin Fusce vel imperdiet mi ac faucibus leo Cras massa massa ultricies et justo vitae molestie auctor turpis Vestibulum euismod porta risus euismod dapibus Nullam facilisis ipsum sed est tempor et aliquam sapien auctor Aliquam velit ligula convallis a dui id varius bibendum quam Cras malesuada nec justo sed aliquet Fusce urna magna malesuada tempus pharetra in fringilla iaculis mi Morbi rutrum iaculis convallis Cras et elit velit Suspendisse eget elit blandit egestas odio at ullamcorper lectus Nulla metus diam elementum ac arcu vitae pharetra dignissim neque Etiam at elit dignissim suscipit nulla ac gravida eros Donec feugiat mi sed purus egestas convallis Vestibulum

A random ggplot



A random data table

Show 10 entries Search:

	GNP.deflator	GNP	Unemployed	Armed.Forces	Population	Year	Employed
1947	83	234.289	235.6	159	107.608	1947	60.323
1948	88.5	259.426	232.5	145.6	108.632	1948	61.122
1949	88.2	258.054	368.2	161.6	109.773	1949	60.171
1950	89.5	284.599	335.1	165	110.929	1950	61.187
1951	96.2	328.975	209.9	309.9	112.075	1951	63.221
1952	98.1	346.999	193.2	359.4	113.27	1952	63.639
1953	99	365.385	187	354.7	115.094	1953	64.989

© Analytic Health 2022

Code

✨ Maintainable ✨ code

Easy to understand and debug

- Coding style
- Comments
- KISS

Easy to modify and enhance

- DRY
- Separation is key

Easy to test

- Automated unit tests

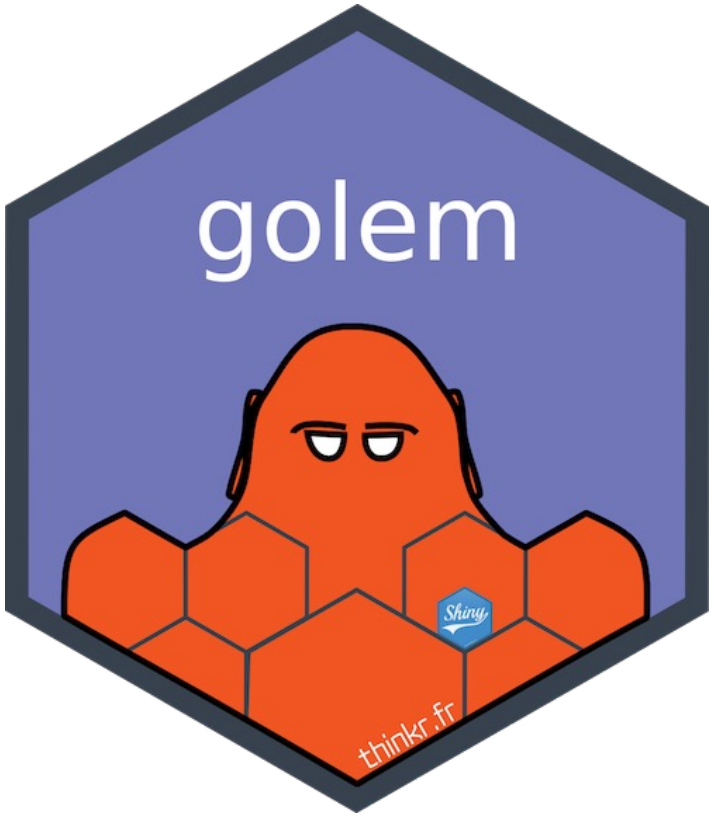
Building an app in Shiny is not different from developing an app in any other framework!

But there are specific Shiny ways to help you out

© Analytic Health 2022

Golem & the almighty Modules

- Developed by thinkR
- Framework for “production-grade” shiny applications
- Provides you with a basic structure
- Your app as an R package
  - Documenting
  - Testing
  - Sharing
- Makes use of Shiny modules



© Analytic Health 2022

… or at least functionize your code

- Functions are the fundamental unit of abstraction in R
- You can write UI-generating functions
- You can write functions to be used in the server
    - Define outputs
    - Create reactive expressions
- Functions don't address the namespacing problem
    - You need to make sure that none of the input and output IDs collide
- While modules add one level of abstraction beyond functions and take care of the namespacing problem, using functions will get you closer to an enterprise-grade app

© Analytic Hcalth 2022

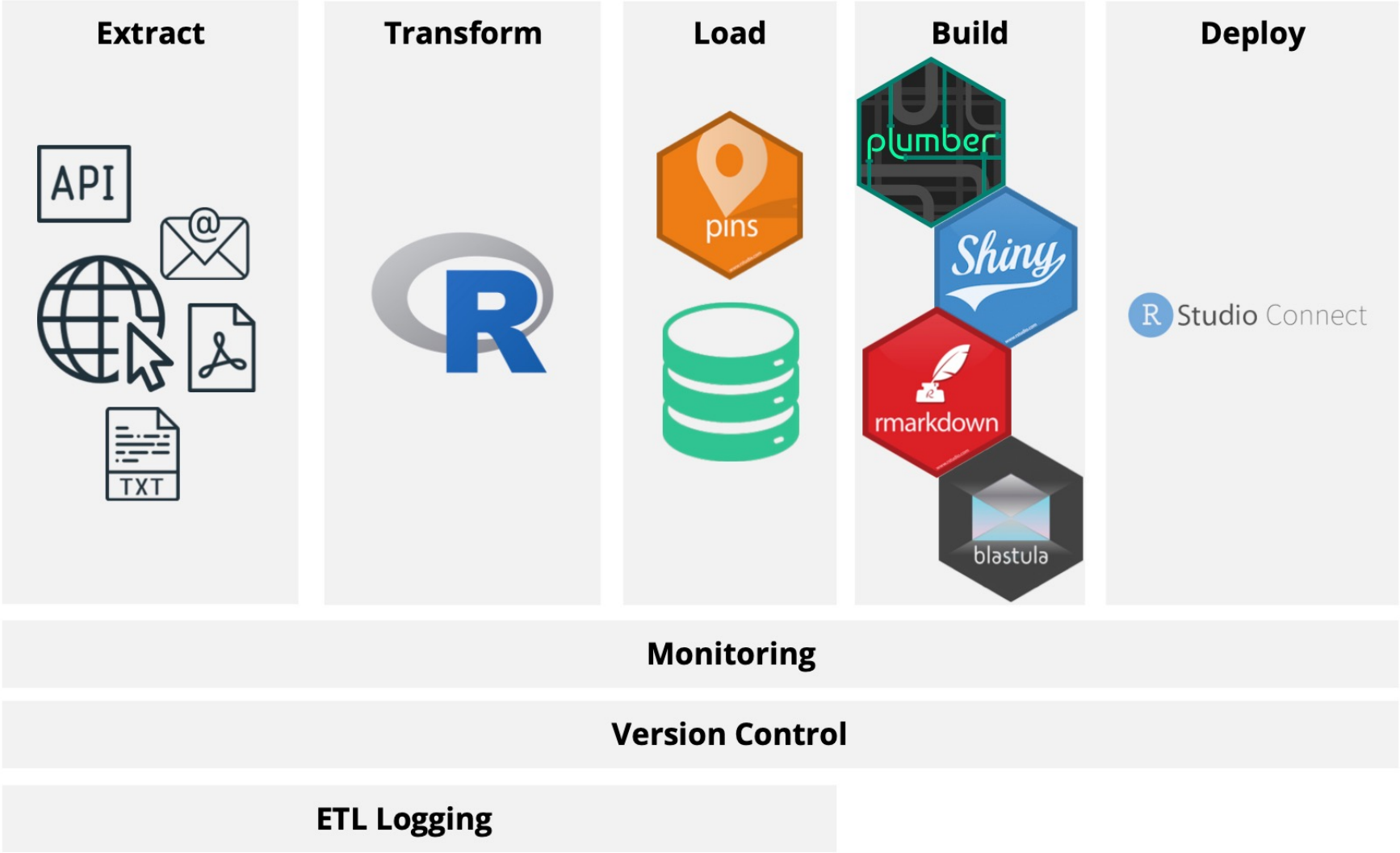
New kid in town: Rhino

- Developed by Appsilon
- Framework allowing you to “develop enterprise-grade Shiny apps at speed”.
- Integrating software best practices and development tools
- Helps to structure your code
- Uses **box** to modularize code

© Analytic Health 2022

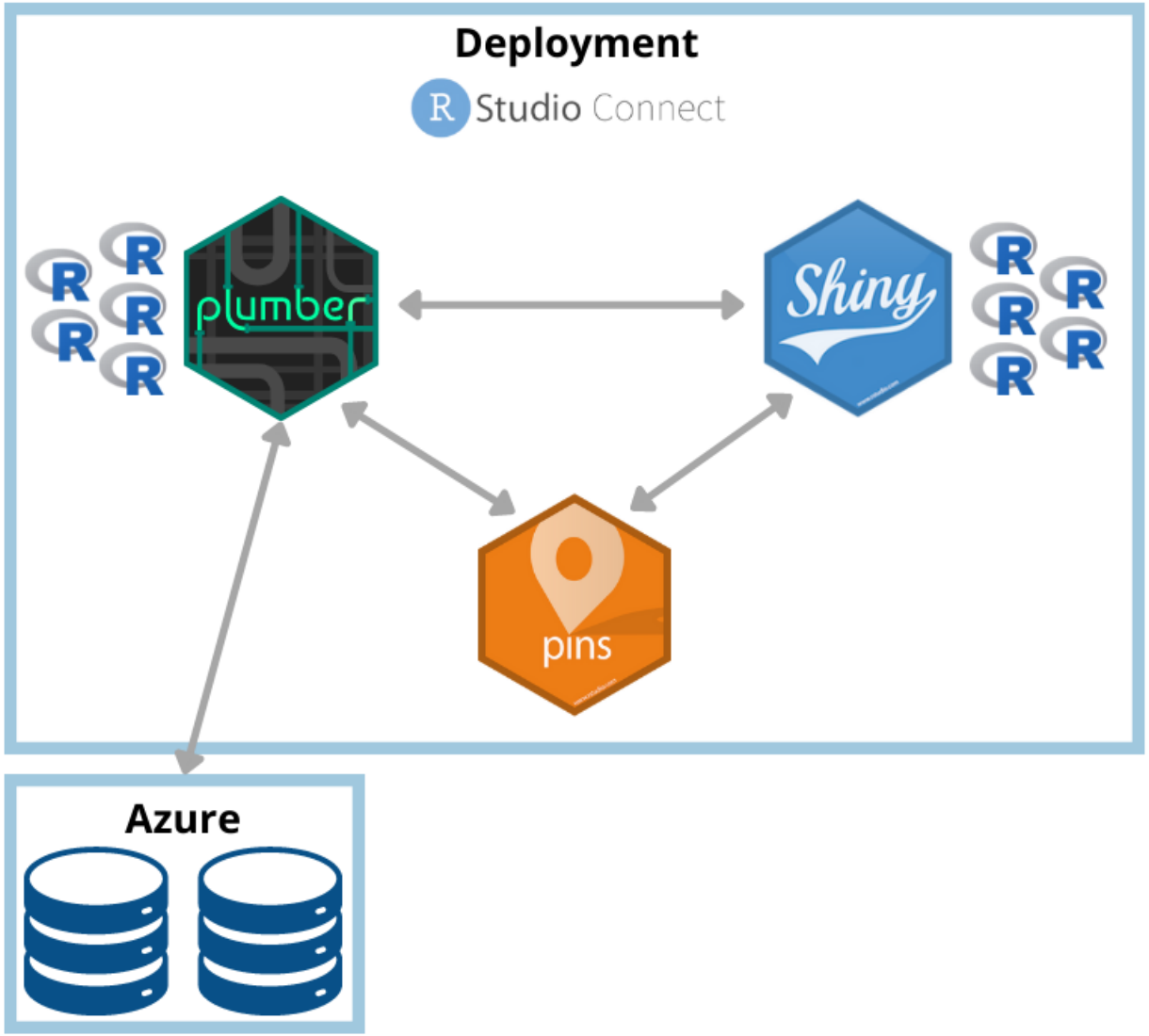
Infrastructure

Our infrastructure



© Analytic Health 2022

Our infrastructure



© Analytic Health 2022

Alternatives

- Load balancing & high availability (Connect)
- Docker + Azure App Service
- Docker + ShinyProxy
- Shiny Server Open Source
- … or solutions from any other cloud service provider

© Analytic Health 2022

Optimizing Shiny

Process configuration

- A Shiny app requires an R process
- **Max processes:** higher means more resource consuming, lower means more chance of rejection
- **Min process:** higher means more resource consuming, lower means more start-up time
- **Max connections per process:** lower means less load per R process, but also more resource consuming
- **Load factor:** lower means many new processes

© Analytic Health 2022

Smart coding: cache

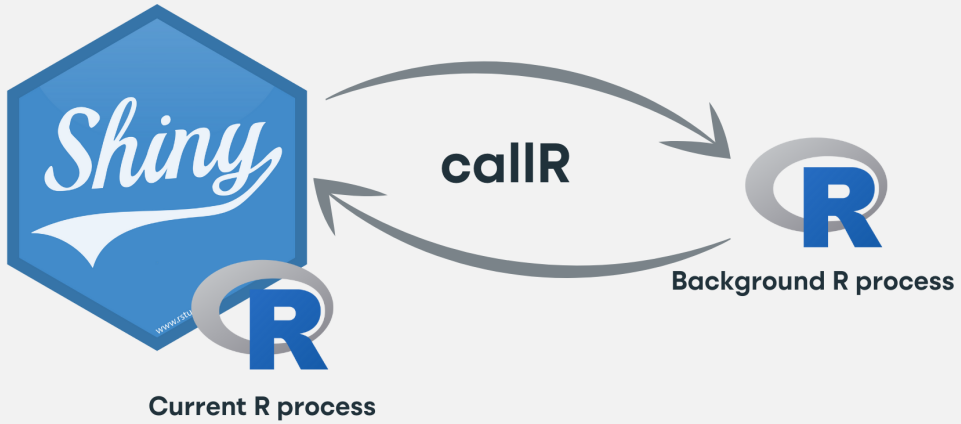
- Caching helps when you're doing the same lengthy calculation which results in the same output
- Two ways to implement caching:
  - Memoise
  - Shiny's bindCache() function
- Disclaimers:
  - Only useful when output does not change regularly
  - It requires space
  - Think about sensitive data

© Analytic Health 2022

Smart coding pt 2: callR

- **Asynchronous execution** of code is important when developing an app for multiple users
- A user shouldn't have to wait for another user
- And a user should also be able to do other things *while waiting*
- **Solution**: spin up a background R process!
- Think about queries to a database

With callR you can create background processes that keep your Shiny app alive for current and concurrent users



© Analytic Health 2022

Smart coding pt 3: API & feather

- Game changer for our app!
- Faster API == Faster Shiny App
- Reading feather files is almost 2.5x times faster in comparison to JSON
- We could reduce file size by 33%
- In some cases, especially with very large volumes of data, we could reduce API response time with as much as 50% 🚀

Efficient data transfer with feather: a binary columnar serializer for data frames

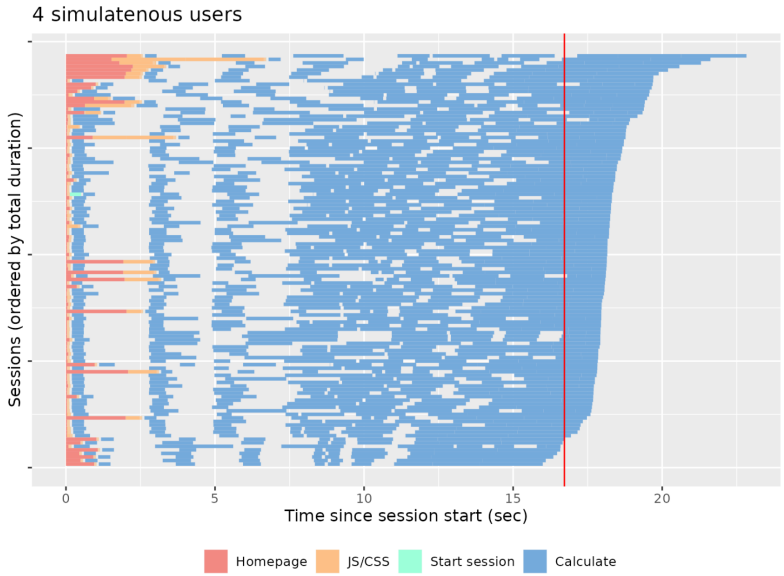
- Implementation of Apache ARROW
- Efficient reading/writing of data
- Language agnostic so easy to exchange
- Maintains meta data, like date types

© Analytic Health 2022

Testing your app

Testing for load: shinycannon 💣

- How many users can your Shiny app support?
- **Shinyloadtest** package and **shinycannon** command line tool
- … with the command line tool you can also automate replaying recordings 👀



shinyloadtest and shinycannon can be used to load test deployed Shiny applications

Record

Record a typical user session in your Shiny app

Simulate

Replay the recordings in parallel to simulate user sessions

Analyse

Analyse the results and identify performance bottlenecks to guide changes to infrastructure, configuration, or code

© Analytic Health 2022

Testing for outcome: shinytest2

- Create and run automated tests
- Using the **testthat** strategy
- Record a test and your interactions will be translated to code 🪄.
- Successor of **shinytest**
- Uses **chromote** to launch a headless Chrome browser

A consistent and scalable way to test your Shiny applications

© Analytic Health 2022

Monitor app performance & usage

Status reports with blastula

Server Monitor

Log files

server	latest_log_date
Server 1	Sun May 22 05:05:02 UTC 2022
Server 2	Sun May 22 05:04:01 UTC 2022

Disk space

server	partition	filesystem	percentage_used
Server 1	/	sdc1	79%
Server 1	/data-drive	sdb1	2%
Server 1	/mnt	sda1	1%
Server 2	/	sdb1	59%
Server 2	/mnt	sda1	1%
Not applicable	fileshare		65%

Server 1 - Status root disk (sdc1): ok
Server 1 - Status data-drive: ok
Server 1 - Status mnt disk (sda1): ok
Server 2 - Status root disk (sdc1): ok
Server 2 - Status mnt disk (sda1): ok
Fileshare - Status fileshare: ok

Status of services

server	service	status
Server 1	nginx	Active
Server 1	RSC	Active
Server 1	RSPM	Active
Server 2	nginx	Active
Server 2	RSW	Active

Response time for application URLs

app	URL	reponse_time
Pharmly Analytics	/pharmly_analytics	0.018706n
Pharmly Cloud Data	/pharmly_cloud_data	0.017347n
		0.045448n

blastula

© Analytic Health 2022

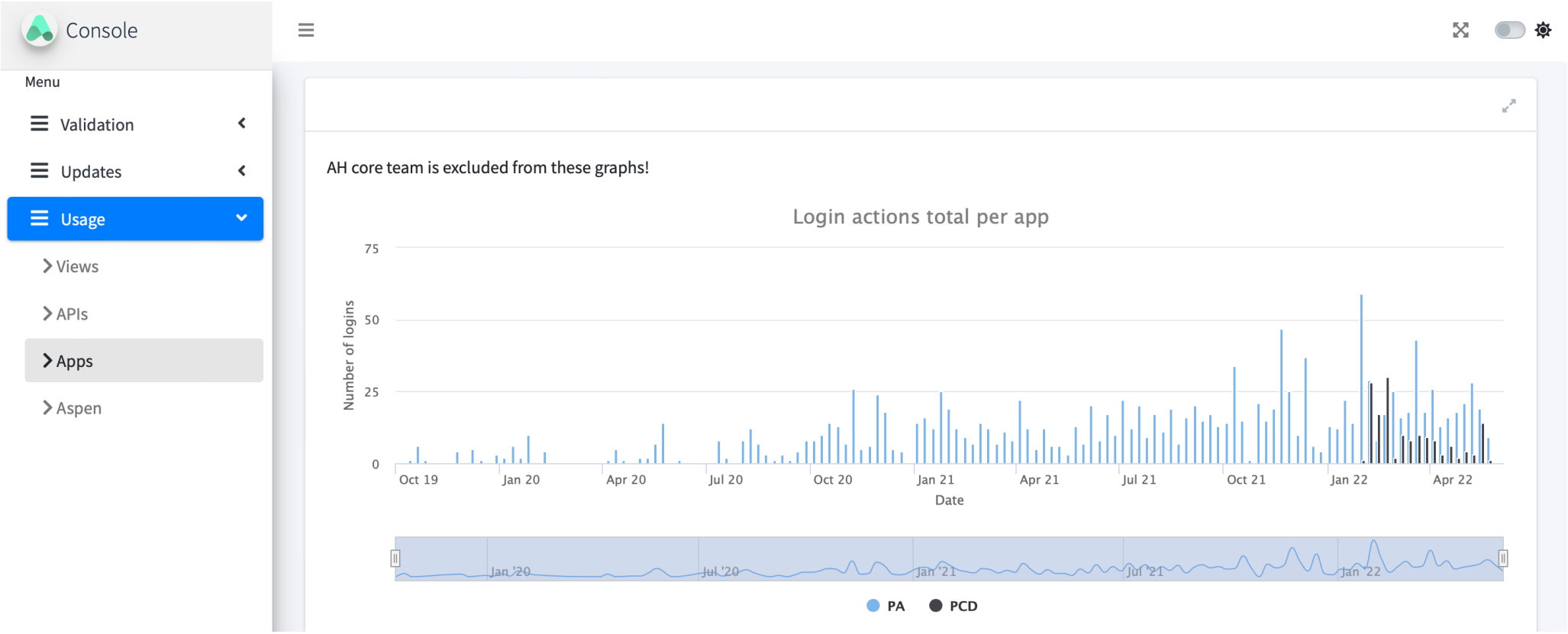
Hey, shinycannon again!

- Is my app behaving the same way as yesterday?
- The command line tool, shinycannon, can also be used to to run a single recording
- Number of workers = 1, loaded duration = 0 → worker will run a single session
- Output can be used to give status reports

```
.
├── detail.log
├── recording.log
└── workers
    ├── 0_0_0.csv
    └── 1_1_0.csv
```


© Analytic Health 2022

Usage: our console



© Analytic Health 2022

Usage: RStudio Connect

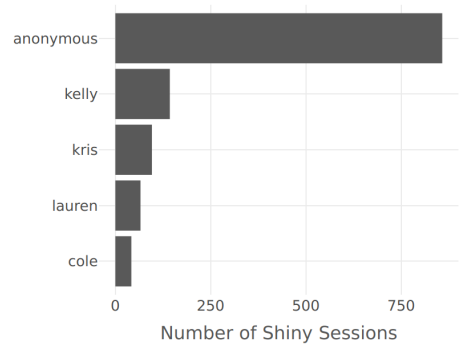
https://solutions.rstudio.com/data-science-admin/tracking/

RStudio Connect Usage - Last 30 Days

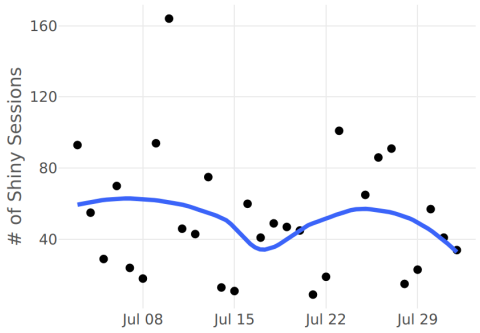
This content summary may contain privileged information. The report is generated using the RStudio Connect Server API and the source code is available online if you'd like to customize your analysis. Data is limited to the last 30 days.

The report uses the environment variables `RSTUDIO_CONNECT_SERVER` and `RSTUDIO_CONNECT_API_KEY` to collect the data. To limit the results to a single publisher, use a publisher API key.

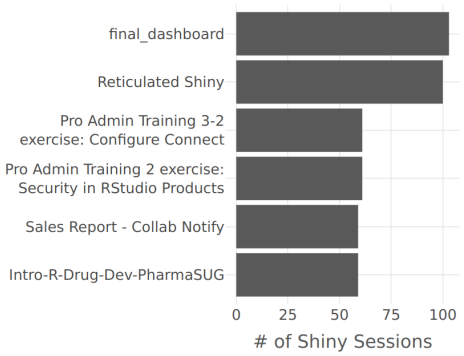
Shiny Sessions by User (Top 5)



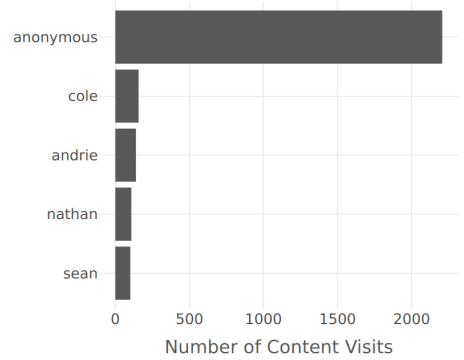
Shiny Sessions Over Time



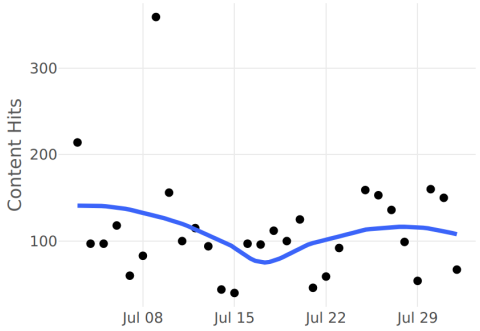
Top Applications



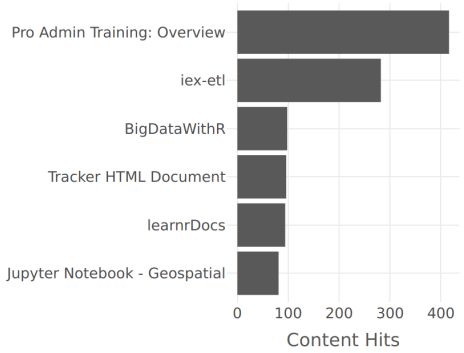
Static Content Hits by User (Top 5)



Static Content Visits Over Time



Top Static Content



© Analytic Health 2022

Enterprise-grade apps with Shiny?

Yes you can!

Questions?

Let’s stay in touch!

https://www.linkedin.com/in/veerlevanleemput/

Package overview & Resources

Mentioned packages & other resources

Packages

- Blastula (https://github.com/rstudio/blastula)
- Shinipsum (https://github.com/ThinkR-open/shinipsum)
- Rhino (https://appsilon.github.io/rhino)
- Golem (https://thinkr-open.github.io/golem)
- Shinycannon (https://github.com/rstudio/shinycannon)
- Memoise (https://memoise.r-lib.org)
- callR (https://github.com/r-lib/callr)
- arrow (https://arrow.apache.org/docs/r/reference/)
- Shinyloadtest (https://rstudio.github.io/shinyloadtest/)
- Shinytest2 (https://github.com/rstudio/shinytest2)

Other resources

- R Studio Connect (https://www.rstudio.com/products/connect/)
- R Studio Workbench (https://www.rstudio.com/products/workbench/)
- My GitHub (https://github.com/hypebright)

© Analytic Health 2022