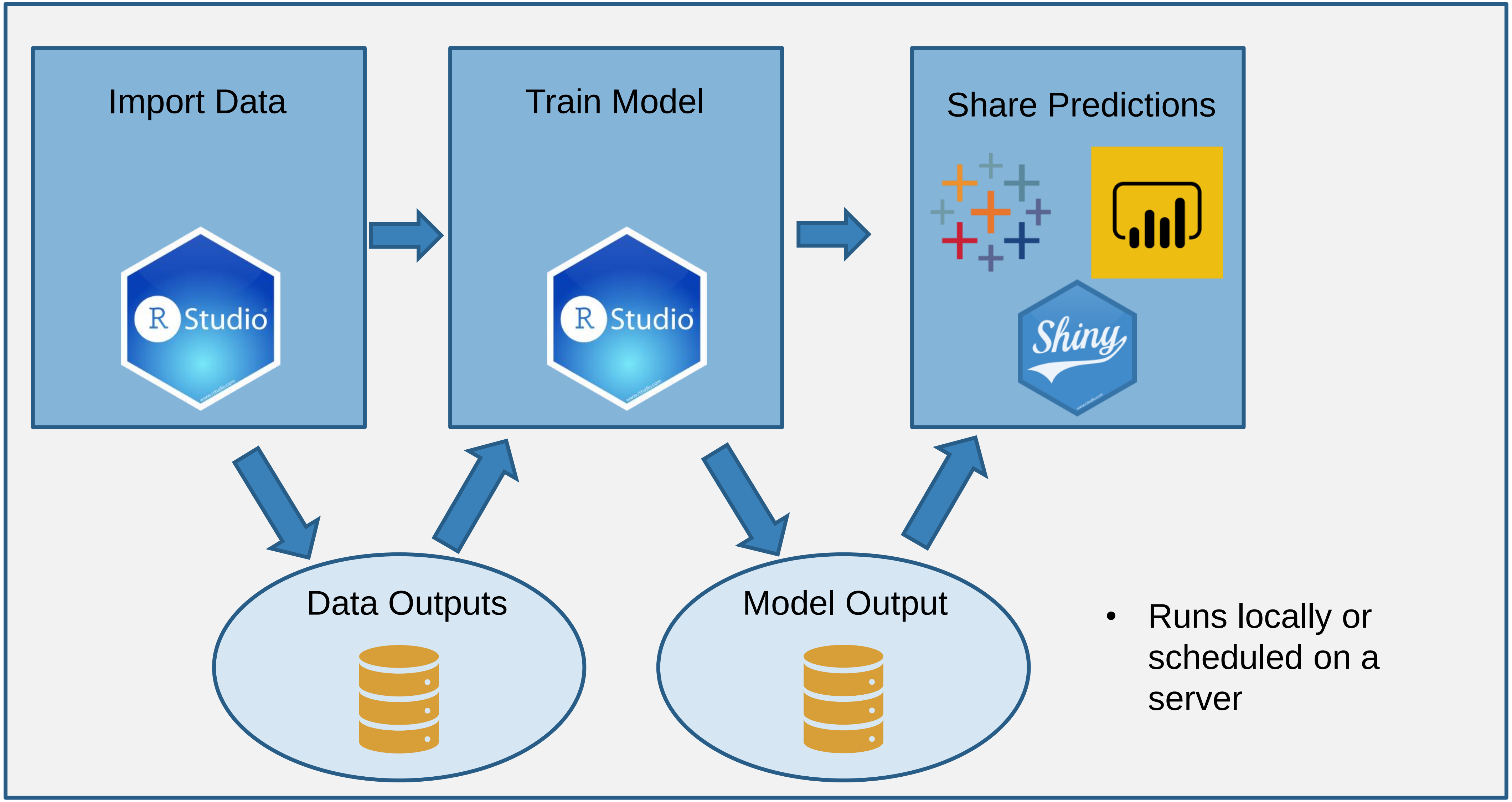
Deploying End-to-End Data Science Production Workflows in R

Alyssa Toay

Old way of working

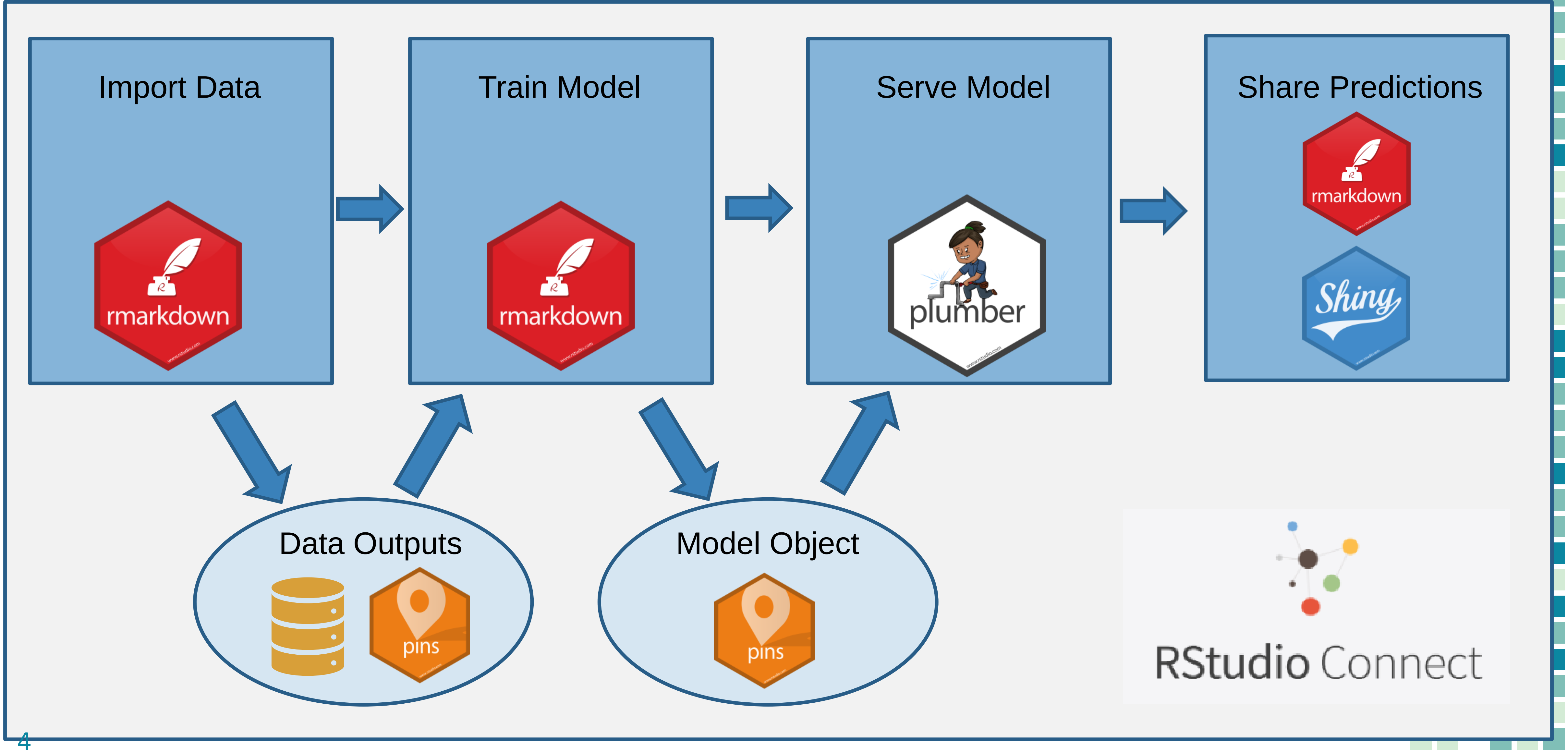


- Runs locally or scheduled on a server

Why something new?

- Easy to collaborate with team members
- Easy to schedule and maintain
- Accessible to business partners
- More reproducible
- Better Security

New way of working



https://solutions.rstudio.com/tour/bike_predict/

R Markdown Overview

- R Markdown documents are flexible frameworks for data science
  - Execute and Save Code
  - Generate static and dynamic reports
- .Rmd files to execute code chunks, display text, and include YAML metadata to help the R Markdown build process
- Supported for SQL, R, and Python
- Reproducible, easy to track in Git, and easy to deploy

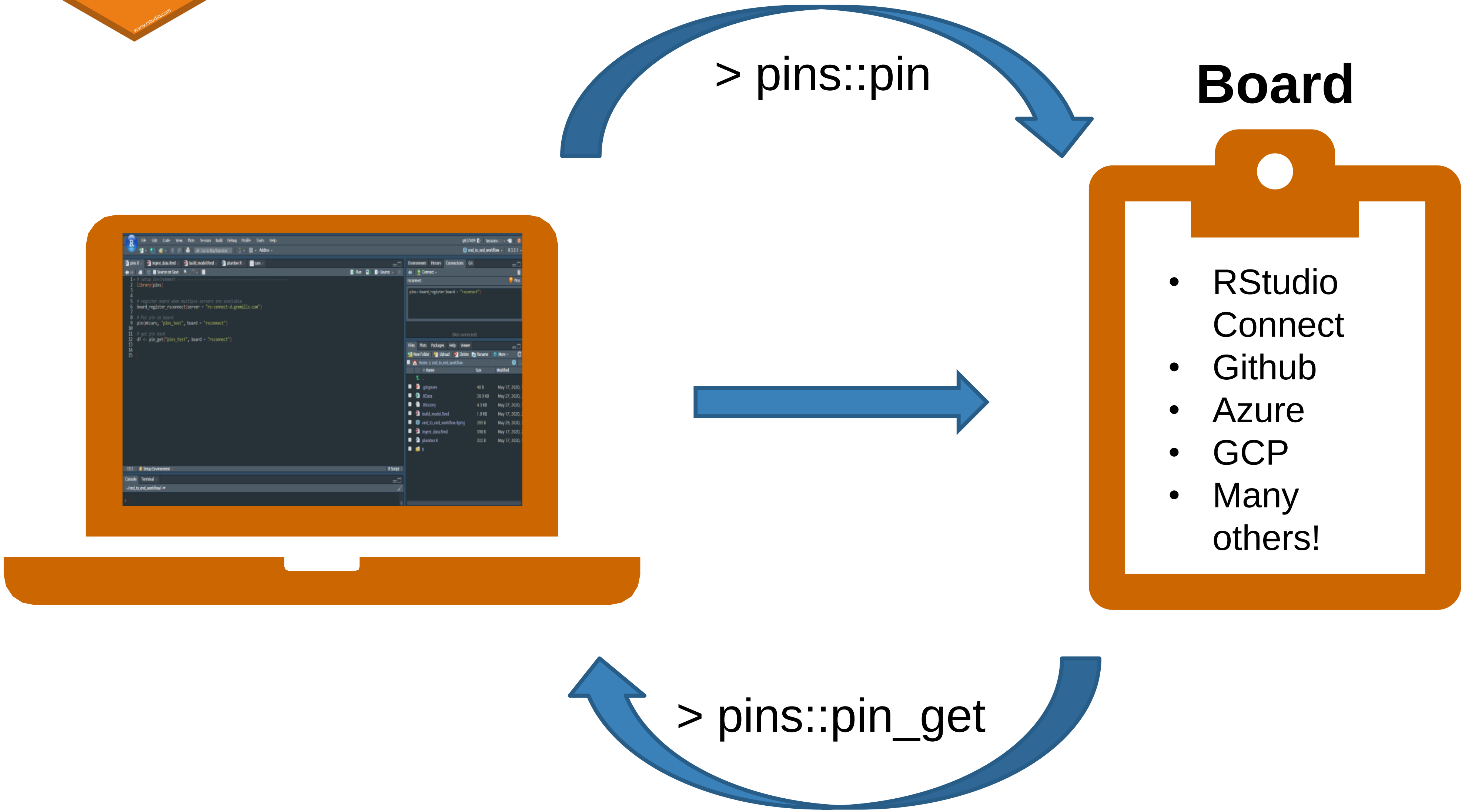
R Markdown Example

R Markdown Best Practices

- Tips to make your R Markdowns functional and reproducible
  - Use parameters or dynamic variables rather than hardcoded variables
  - Name your code chunks
  - Modularize your code chunks
  - Don’t use plain text passwords
  - Don’t include unused packages, code, or dataframes
  - Use relative file paths
  - Use lintr and styler packages to point out style violations and automatically reformat code
  - Use testthat to incorporate unit tests
  - Convert your project to a package ☺

Pins Overview

- Use the pins package to pin remote resources locally, work offline, or cache results
- Share resources in local folders, RStudio Connect, Github, etc.
- Take an R object, dataframe, model object, etc. and pin to a board
- Great for collaboration
  - Instead of emailing objects or saving it to a database → share the name of the board and the pin name
  - Users will always get the newest version



> pins::pin

Board

- RStudio Connect
- Github
- Azure
- GCP
- Many others!

```
# Setup Environment ----------------------------------------------------------
library(pins)


# register board when multiple servers are available
board_register_rsconnect(server = "rs-connect-d.genmills.com")

# Put pin on board
pin(mtcars, "pins_test", board = "rsconnect")

# get pin back
df <- pin_get("pins_test", board = "rsconnect")
```

```
pins::board_register(board = "rsconnect")
```

Name	Size	Modified
.gitignore	40 B	May 17, 2020,
.RData	28.9 KB	May 27, 2020,
.Rhistory	4.3 KB	May 27, 2020,
build_model.Rmd	1.8 KB	May 17, 2020,
end_to_end_workflow.Rproj	205 B	May 29, 2020,
ingest_data.Rmd	598 B	May 17, 2020,
plumber.R	332 B	May 17, 2020,
R		

File Edit Code View Plots Session Build Debug Profile Tools Help
pins.R
ingest_data.Rmd
build_model.Rmd
plumber.R
cars
Source on Save
Run
Source
Environment History Connections Git
Connect
rsconnect
Pins
(Not connected)
Files Plots Packages Help Viewer
New Folder Upload Delete Rename More
Home › end_to_end_workflow
end_to_end_workflow
R 3.5.1
Console Terminal
R Script

> pins::pin_get

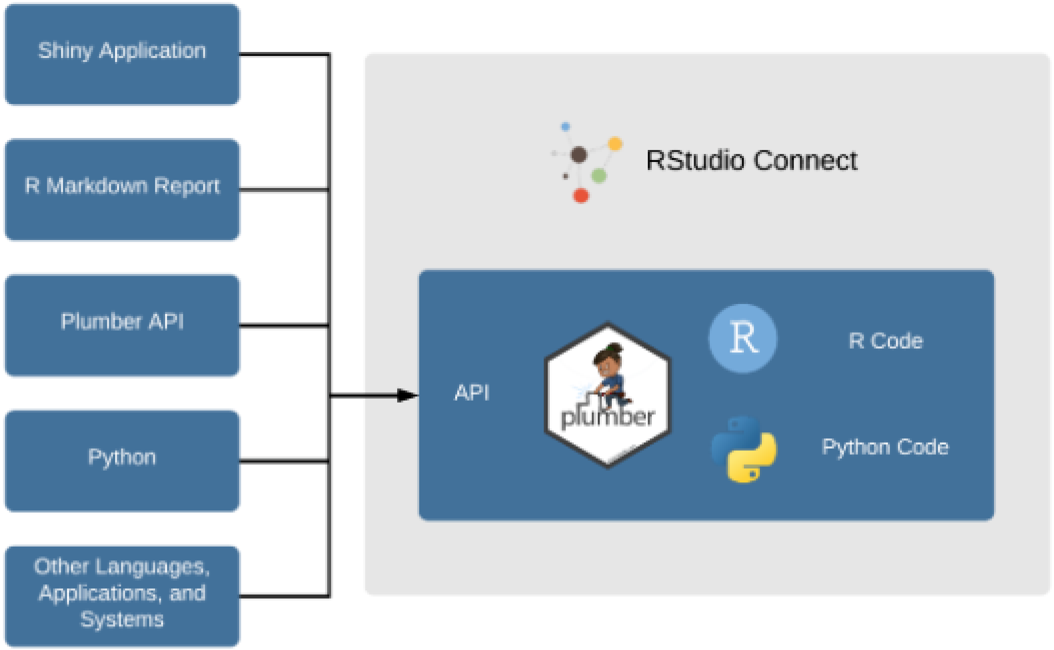
Pins Example

Pins Best Practices

- Pins is not a database replacement
- Pins are:
  - Small – 1GB or Less
  - Re-used
  - Re-created
  - Current – you just want the newest version
  - Shareable (+ you can secure them)
  - Great for objects that are computationally expensive and you’d like to cache the results
  - Great for model objects – simple to update the model independently from an API that serves model predictions

Plumber Overview

- Deploy machine learning models as a RESTful API built in R
- You can log details about API requests and responses
- You can call Plumber APIs from other languages



```
library(plumber)

#* Echo back the input
#* @param msg The message to echo
#* @get /echo
function(msg = "") {
    list(msg = paste0("The message is: '", msg, "'"))
}

plumb(file = "overview/plumber.R")$run()
```

https://solutions.rstudio.com/examples/rest-apis-overview/

Plumber Example

Plumber + Shiny??

- Plumber is great to host model APIs and get predictions
- Shiny is intended for human consumption
- Shiny will scale well as the data is not duplicated in memory for each user
- The data is housed in one location (Plumber API & pins)

- The underlying data can be large, which is not suitable for a shiny app
- The API can be updated without redeploying the shiny application.

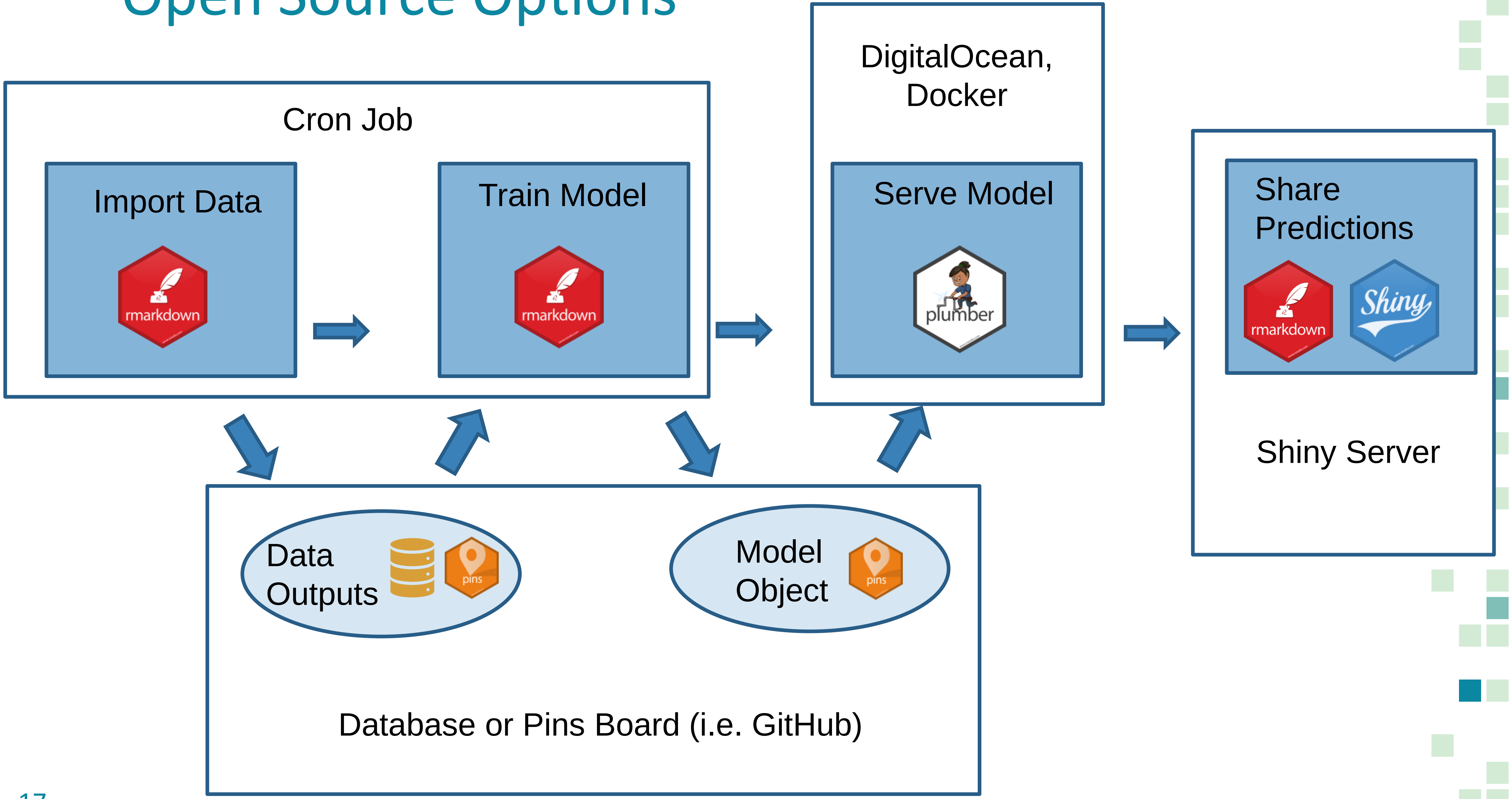
Shiny Overview

- A web application framework for R that allows you to turn your data into an interactive web app
    - User interface script that controls the layout and appearance by converting R code into HTML
    - Server script that contains the code needed to build the app

- Connect Shiny to Plumber APIs to get predictions from your model or monitor model performance

Shiny Example

Open Source Options



- Finally, remember to use **testthat** in your workflows!

Helpful Links

- R Markdown: https://rmarkdown.rstudio.com/
- Pins: http://pins.rstudio.com/
- Plumber: https://www.rplumber.io/
- Plumber: https://rstudio.com/resources/webinars/plumbing-apis-with-plumber/
- Rest APIs overview: https://solutions.rstudio.com/examples/rest-apis-overview/
- Shiny: https://shiny.rstudio.com/
- Testhat: https://testthat.r-lib.org/
- Putting R in Prod: https://putrinprod.com/
- Another End to End Example: https://solutions.rstudio.com/tour/bike_predict/