



R Shiny in the Cloud – Automating the Process with Extensive Collaboration

PHUSE SDE 2025 - FRANKFURT

Abu Zafar Mohammad Sami, *Principal Statistical Programmer*

www.mainanalytics.de



Deployed and Developed a Shiny App: Next Challenges

```
ui <- fluidPage(
```

```
  titlePanel(<your title>)
```

```
  sidebarLayout
```

```
    sidebarPanel(<input widgets>)
```

```
    mainpanel (<output>)
```

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

```
  <calculations>
```

```
}
```

```
shinyApp(ui=ui, server=server)
```

mainanalytics GmbH

Select Types of Analysis

KAPLAN-MEIER SURVIVAL ANALYSIS

ANCOVA ANALYSIS

SUMMARY STATISTICS

MORE ▾

Km Type

☒ Plot

☐ Summary

☐ Details

Analysis Datasets*

adtte.xpt ▾

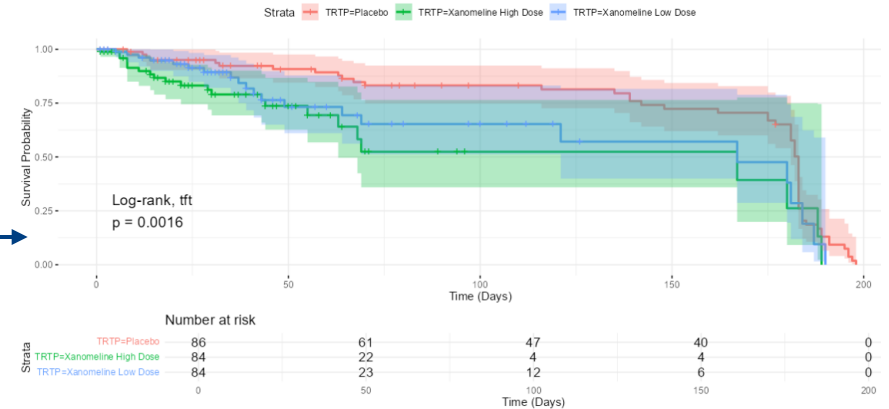
Choose Parameter*

PARAM ▾

Choose Parameter Value*

Time to First Dermatologic Event

PLOT KAPLAN-MEIER



```
fit <- survfit(<extract parameters from user selections>)  
ggsurvplot <- (fit, <draw plot>)
```

The Challenges After Developing and Deploying a Shiny App

- **Validate Packages:** Ensure all utilized R packages are appropriate and functioning as intended.
- **Statistical Review:** Conduct a thorough review of statistical methods and validate results for accuracy.
- **Enhance Test Coverage:** Increase the scope and depth of tests to ensure robust functionality of the application.
- **Streamline CI/CD:** Simplify the complexity of continuous integration, development, and deployment processes for better efficiency.

Imposing good software engineering practices

Agenda



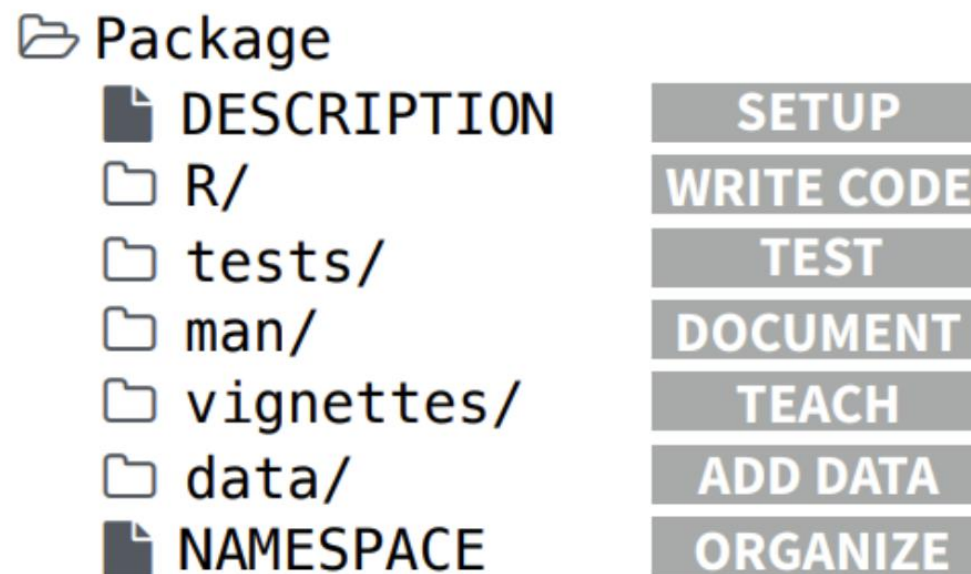
- Shiny App in Cloud
 - Prerequisites
 - Continuous Integration
 - Continuous Delivery and Deployment
 - Cloud Scripting
- Examples
- Conclusion

Shiny App in Cloud

Prerequisite: Follow Standard R Package Convention

There are multiple packages useful for package development, incl. *usethis* (handily automates repetitive tasks).

The 7 common parts of an R Package



Shiny App in Cloud

Prerequisite: Package - renv (Reproducible Environments)

System library

Contains all installed packages

Project library

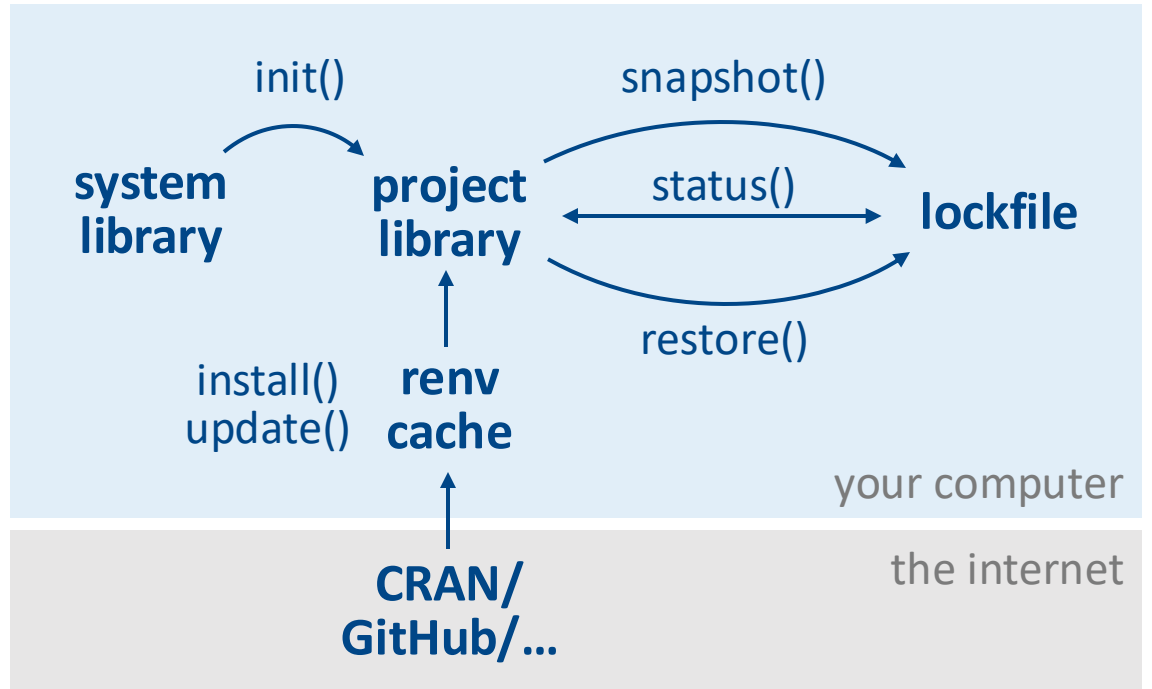
Gives each project its own independent collection of packages

Lockfile

Records metadata about packages to re-install them on a new machine

renv cache

Uses package cache, is shared across all projects



Shiny App in Cloud

Prerequisite: Package - renv (Reproducible Environments)

Advantages

- Resolves version control problem
- Installing and restoring packages is much faster, as renv can find and re-use previously installed packages from the cache

```
{
  "R": {
    "Version": "4.2.2",
    "Repositories": [
      {
        "Name": "CRAN",
        "URL": "https://cran.rstudio.com"
      }
    ]
  },
  "admiral": {
    "Package": "admiral",
    "Version": "1.1.1",
    "Source": "Repository",
    "Repository": "CRAN",
    "Requirements": [
      "R",
      "admiraldev",
      "cli",
      "dplyr",
      "hms",
      "lifecycle",
      "lubridate",
      "magrittr",
      "purrr",
      "rlang",
      "stringr",
      "tidyr",
      "tidyselect"
    ],
    "Hash": "a06f94a38ce50c4745ac7102f161bea4"
  },
}
```

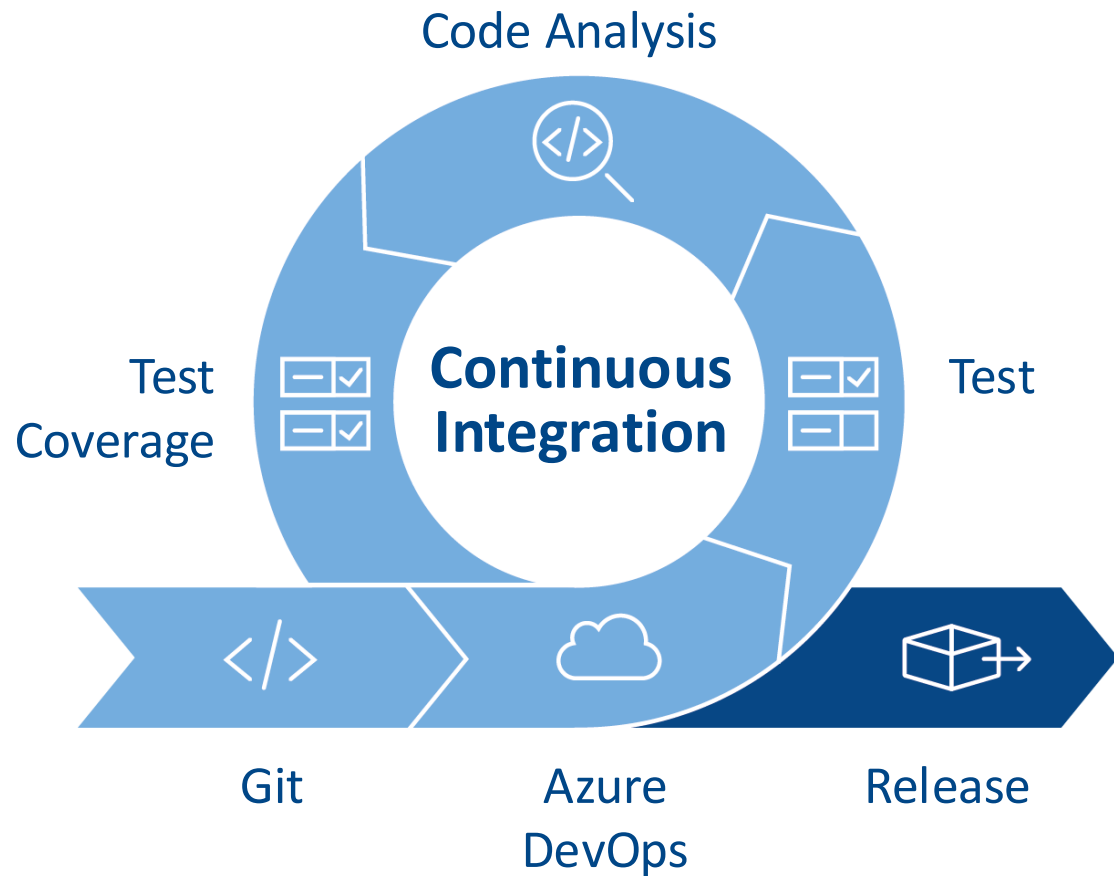
Example: Lockfile

Shiny App in Cloud

Continuous Integration

- Merged code changes into a central repository
- Automatically builds, tests and validates the leveraged code

CI addresses bugs earlier and quicker, ensures code quality and reduce the time it takes to validate and deployment of new software updates.



Shiny App in Cloud

Continuous Delivery and Deployment (CD)

- After passing all CI automated tests, the application can be deployed at any time through a configured automated release process.
- The product is stored as a container image in a private Azure Container Registry. Azure App Service then automatically pulls the latest container image during deployment, allowing authorized collaborators to interact with the Shiny app, facilitating continuous deployment.

CD enables faster and more frequent releases with less/no manual intervention and ensures consistent deployment across environments and reducing the risk of human error.

Shiny App in Cloud

Cloud Scripting: Package - renv (Reproducible Environments)

Configure the YML-File

```
- script: |  
  R -e "install.packages('renv')"  
  displayName: 'Install R Packages with renv'  
  
- task: Cache@2  
  inputs:  
    key: 'renv | "$(Agent.OS)" | renv.lock'  
    path: '~/.cache/R/renv'  
  
- script: |  
  R -e "renv::restore()"  
  displayName: 'Restore R Packages with renv'
```



cache renv in cache directory

Shiny App in Cloud

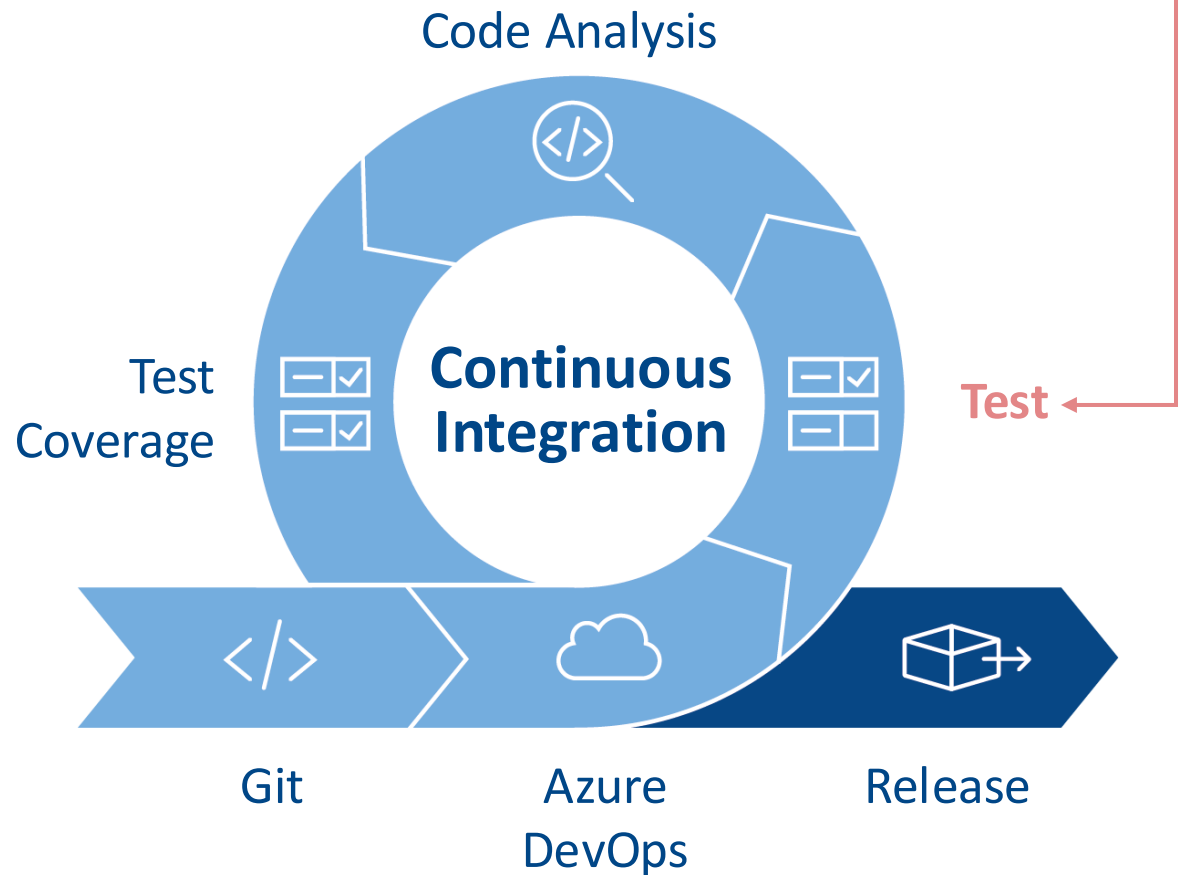
Cloud Scripting

R CMD Checks

CI pipelines run R CMD Checks to verify the quality and structure of an R package

Configure the YML-File:

```
- script: |  
  R -e "install.packages('rcmdcheck')"  
  R -e "rcmdcheck::rcmdcheck(error_on = 'warning')"  
displayName: 'Install and Run R CMD check'
```



Details: <https://r-pkgs.org/r-cmd-check.html>

Shiny App in Cloud

Cloud Scripting

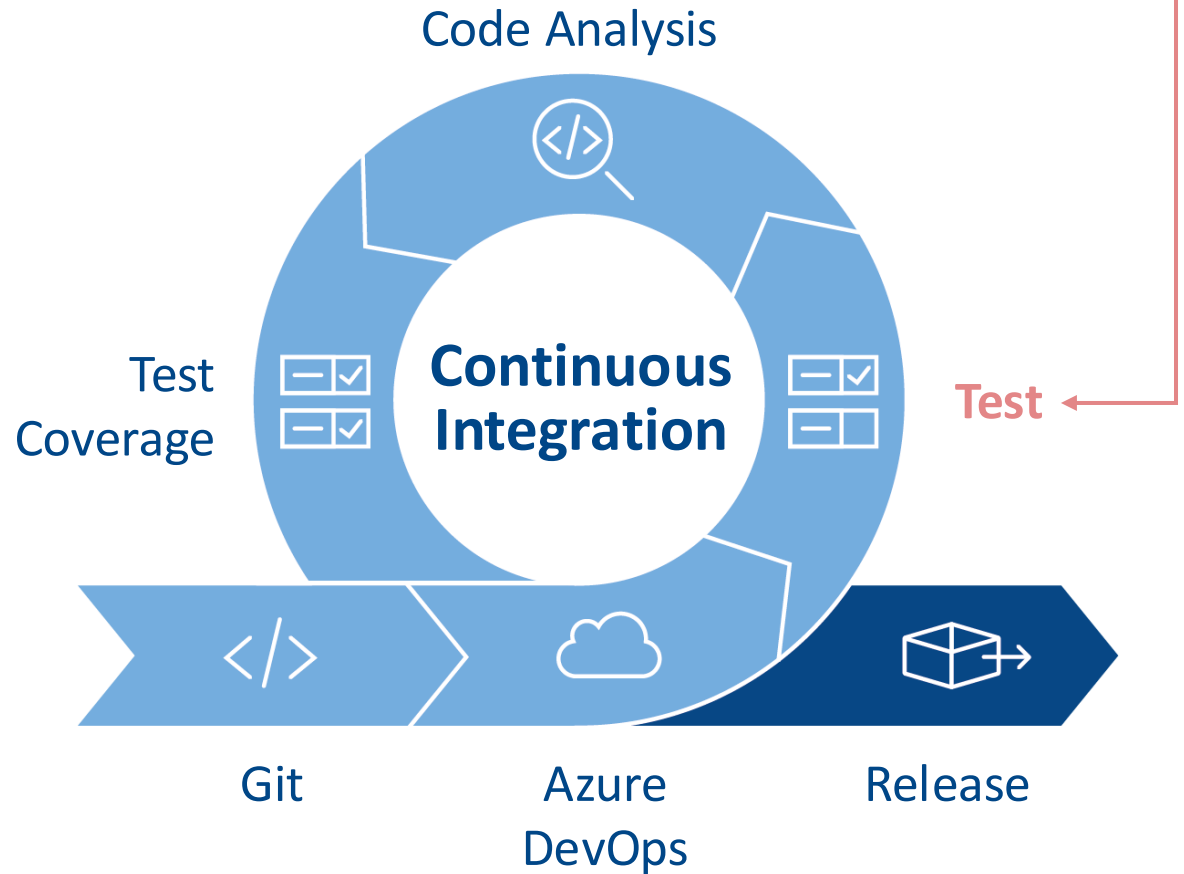
R CMD Checks

Package Structure

- Checking for executable files
- Checking for hidden files and directories.
- Checking for sufficient/correct file permission
- Checking installed package size.
- Checking top-level files.

R code

- Checking R files for non-ASCII characters
- Checking R files for syntax errors
- Checking replacement functions



Shiny App in Cloud

Cloud Scripting

R CMD Checks

Documentation:

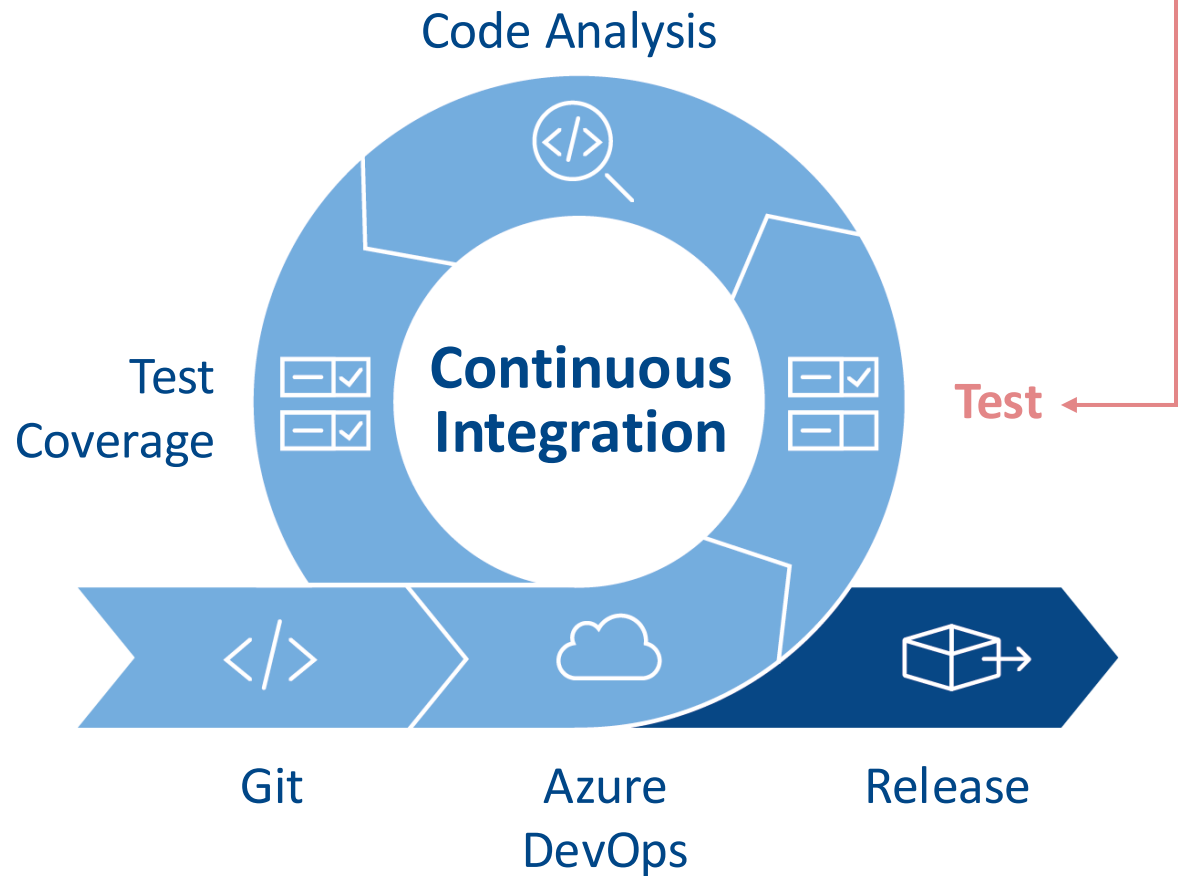
- Checking .Rd files

Tests:

- Checking each file in *tests/* is run

Vignettes:

- Checking running R code from vignettes
- Checking for unstated dependencies in vignettes



Shiny App in Cloud

Cloud Scripting

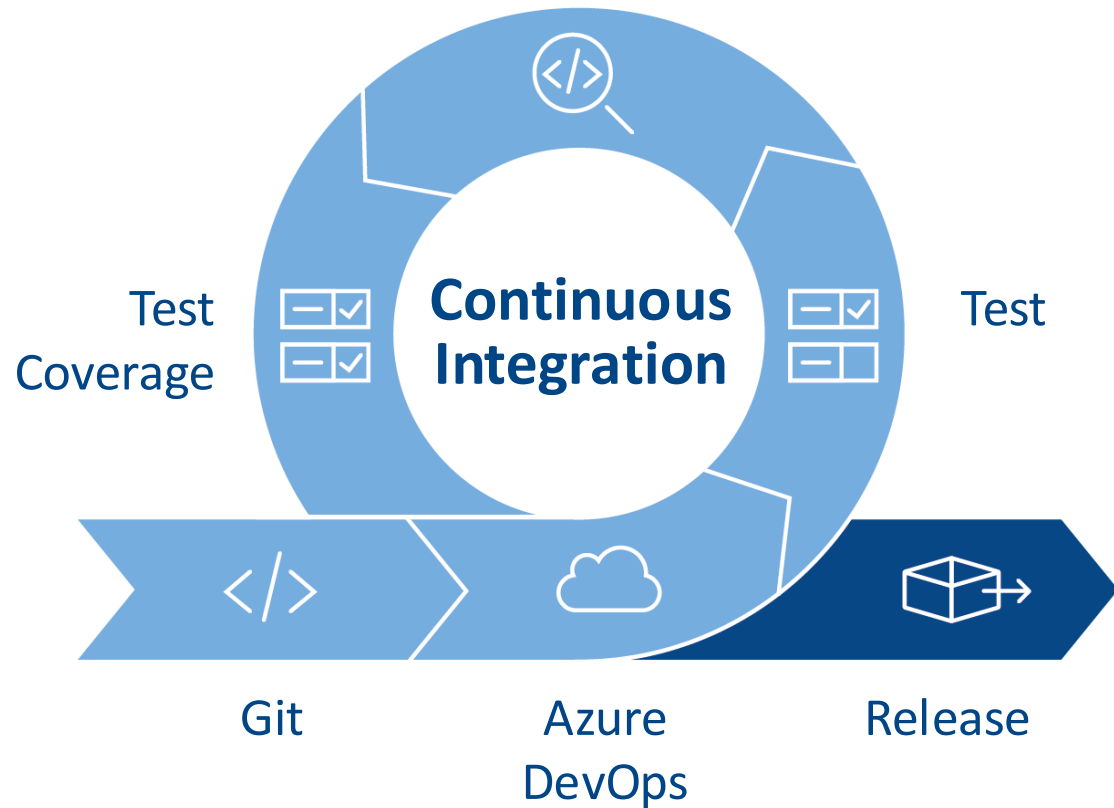
[oxsecurity/megalinter](#) (lintr package) → Code Analysis

On each push request, automatically analyze all code through:

- Syntax Error Detection
- Semantic Issue Identification
- Style Adherence

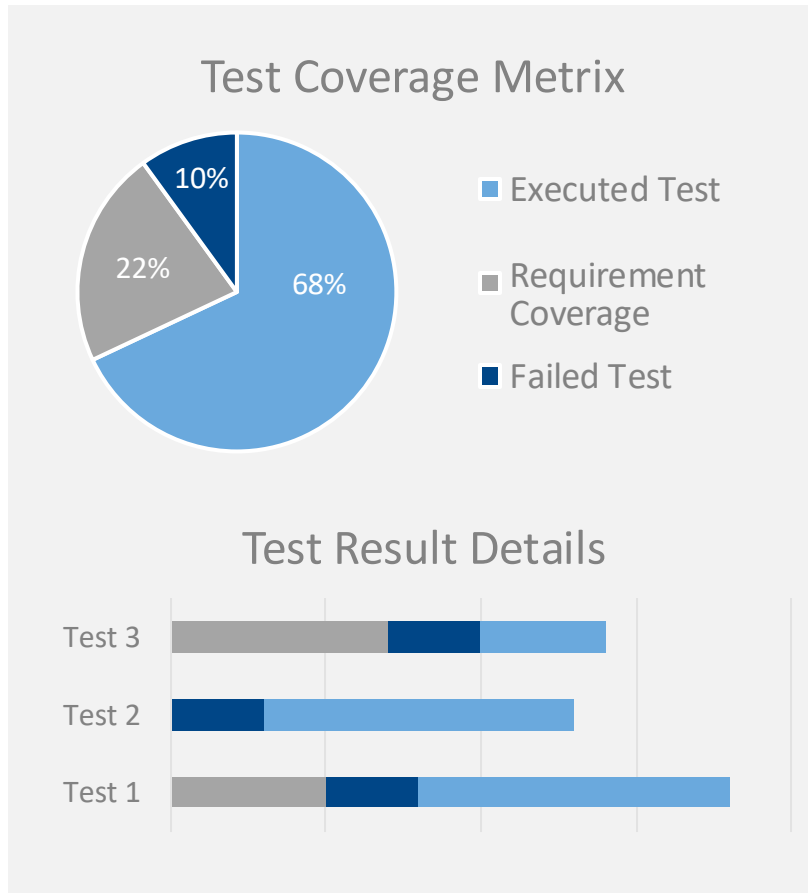
Configure the YML-File:

```
- script: |  
  R -e "install.packages('lintr')"  
  R -e "lintr::lint_package()"  
displayName: 'Install and Run lintr'
```



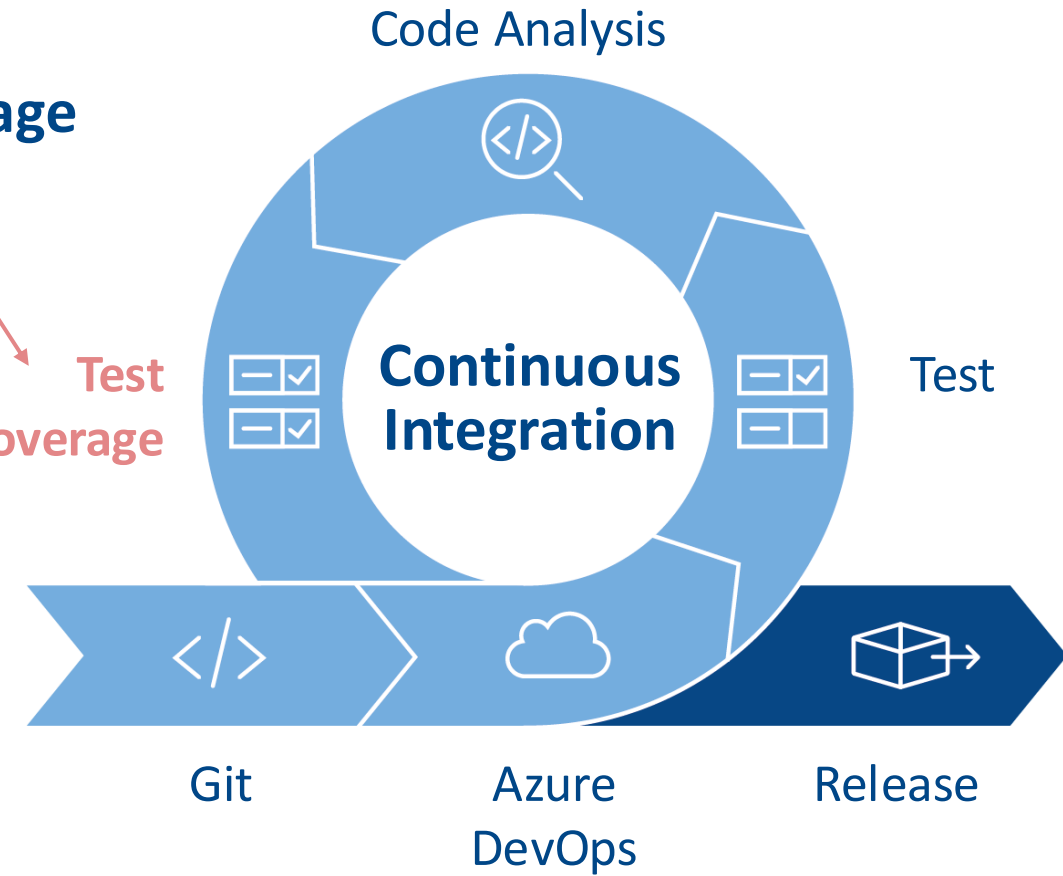
Shiny App in Cloud

Cloud Scripting



covr Package

Test Coverage



Examples

ANCOVA Module Output

Type III ANCOVA of Diastolic Blood Pressure (mmHg) by Planned Treatment					
adjusted for Baseline Diastolic Blood Pressure (mmHg)					
	n	Mean	SE	95% CI	p-value
Placebo					
Model Estimates	72	75	0.88	[74, 77]	
Xanomeline Low Dose					
Model Estimates	60	75	0.96	[73, 77]	
Contrast / Placebo		-0.5	1.3	[-3.1, 2.1]	0.698
Xanomeline High Dose					
Model Estimates	56	76	0.99	[74, 78]	
Contrast / Placebo		0.41	1.3	[-2.2, 3]	0.755
Contrast / Xanomeline Low Dose		0.92	1.4	[-1.8, 3.6]	0.507
Treatment p-value: 0.800					
Covariate p-value: <0.001					

A corresponding forest plot should also be added to the Shiny application

Examples

DevOps

phuseeuconnect2024 +

Overview
Boards
Repos
Files
Commits
Pushes
Branches
Tags
Pull requests
Advanced Security
Pipelines
Test Plans
Artifacts

master

Pushes

8/27/2024 (4 updates)

- > **AS** Updated to 1f40f794: Updated renv.lock
1f40f794 Abu Sami Tue at 3:40 PM ✖ failed
- > **AS** Updated to edf83680: Detect Error
edf83680 Abu Sami Tue at 3:38 PM
- > **AS** Updated to fb6a07c2: Update azure-pipelines.yml for Azure Pipelines
fb6a07c2 Abu Sami Tue at 1:36 PM
- > **AS** Created at ccc7a4af: firstcommit
ccc7a4af Abu Sami Tue at 1:25 PM ✖ failed

Continue tracking the results of each cloud push.

Examples

DevOps

Windows PowerShell

```
PS C:\Users\abu.sami\Desktop\phuseeuconnect2024> git init
Reinitialized existing Git repository in C:/Users/abu.sami/Desktop/phuseeuconnect2024/.git/
PS C:\Users\abu.sami\Desktop\phuseeuconnect2024> git add .
PS C:\Users\abu.sami\Desktop\phuseeuconnect2024> git commit -m "Resolved CI Error"
PS C:\Users\abu.sami\Desktop\phuseeuconnect2024> git config http.postBuffer 524288000
PS C:\Users\abu.sami\Desktop\phuseeuconnect2024> git push -u origin1 main
```



Push request of the updated version of the application to the cloud via version control system

Examples

DevOps

#20240831.6 • Resolved CI Error

phuseeuconnect2024

Cancel

:

Summary

Code Coverage

Triggered by

AS Abu Sami

View change

Repository and version

Time started and elapsed

Related

Tests and coverage

phuseeuconnect2024

Just now

0 work items

Get started

main c7465964

10s

0 artifacts

Jobs

Name	Status	Duration
Check R Package	Queued	
Lint	Queued	

Examples

DevOps

Repository and version
phuseeuconnect2024
master 1f40f794

Time started and elapsed
Tue at 3:40 PM
42m 29s

Related
0 work items
4 published

Tests and coverage
75% passed
0.96% covered

Errors 2 Warnings 4

- Bash exited with code '1'.
Check R Package • Install and check package
- There are one or more test failures detected in result files. Detailed summary of published test results can be viewed in the Tests tab.
Linter • Publish MegaLinter Results

[View documentation for troubleshooting failed runs](#)

Jobs

Name	Status	Duration
Check R Package	Failed	37m 22s
Linter	Failed	4m 46s

- 100% passed is required
- Covered should be improved

Examples

DevOps

Repository and version
phuseeuconnect2024
main 5c19393d

Time started and elapsed
Today at 6:39 AM
1h 0m 7s

Related
0 work items
4 published

Tests and coverage
100% passed
0.96% covered

Warnings 3

Task '[To be deprecated] Publish code coverage' version 1 (PublishCodeCoverageResults@1) is deprecated.
Check R Package • Initialize job

The PublishCodeCoverageResults@1 is deprecated. Users are recommended to switch to task version 2. For more details, see <https://devblogs.micros...>
Check R Package • Initialize job

New V2 version of task publishing code coverage results is available to all our customers now. We highly recommend to stop using the V1 version an...
Check R Package • Publish code coverage

Jobs

Name	Status	Duration
Check R Package	Success	55m 2s
Lint	Success	4m 23s

100% passed is achieved

Examples

DevOps

Summary **Tests** Code Coverage

Summary

3 Run(s) Completed (3 Passed, 0 Failed)

88

Total tests

+88



88 ● Passed
0 ● Failed
0 ● Others

100%

Pass percentage

↑ 100%

8s 10ms

Run duration ⓘ

↑ +8s 10ms

0

Tests not reported

Examples

DevOps

Summary Tests Code Coverage

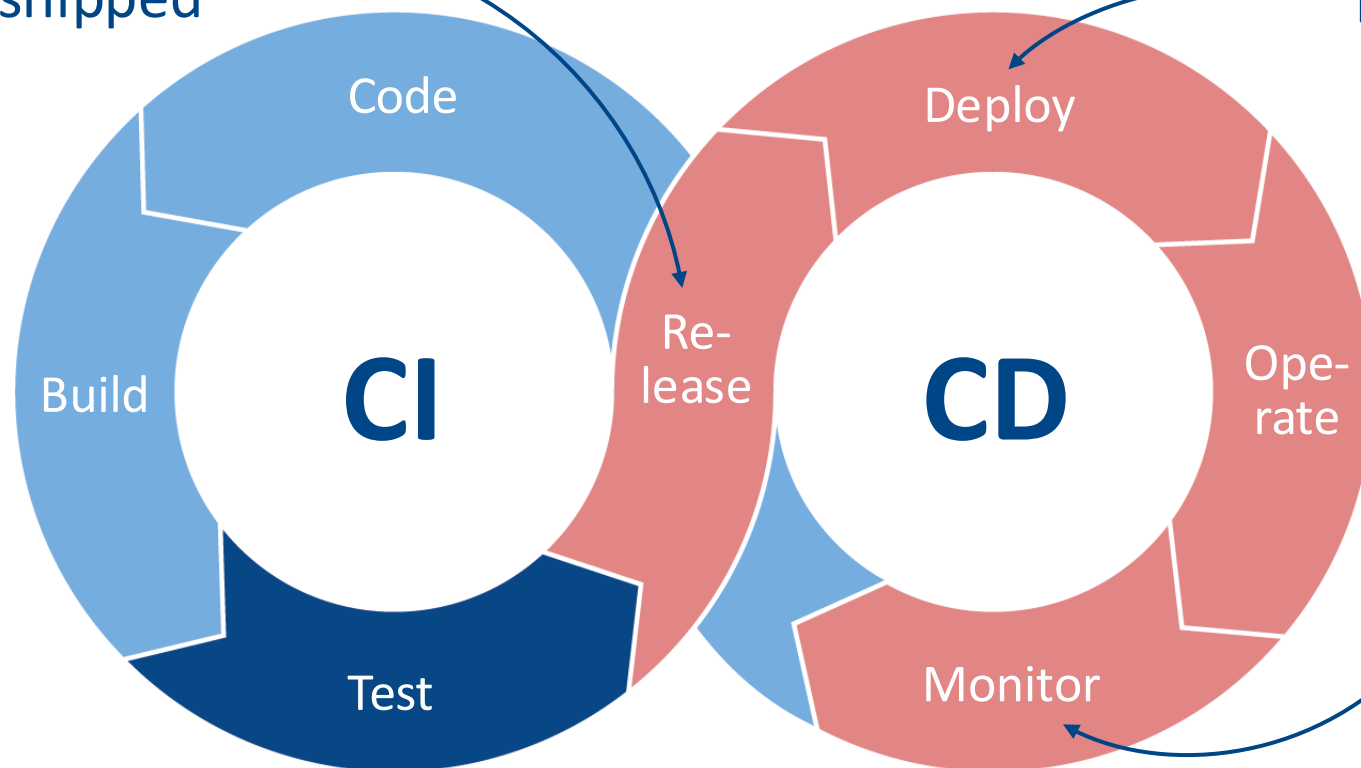
	Line coverage					
Name	Covered	Uncovered	Coverable	Total	Percentage	
shiny.portfolio	22	2272	2294	3722	0.9%	
app-main.R	0	1	1	17	0%	
form-engine.R	0	57	57	157	0%	
import.R	20	7	27	104	74%	
module-dsfilter.R	0	415	415	603	0%	
module-form-anova.R	0	160	160	206	0%	
module-form-numbycat.R	0	92	92	132	0%	
module-form-numbynum.R	0	86	86	126	0%	
module-form-surv.R	0	112	112	151	0%	
module-stat-anova-ancova.R	0	443	443	556	0%	

Example

Ready for Delivery and Deployment (CD)

Build container,
ready to be shipped

To Azure cloud



Activity logs
in Azure

Conclusion

Strengths

- In response to each update, minimal or no human intervention is required to release a new version
- Foster better collaboration among teams by providing version control, build automation, testing, and deployment, all in one place.

Challenges

- Improvements in unit tests are required. My colleague, Féat Yann, will be discussing this topic, 'Analysis Results Standard for Test-Driven Development,' at the 2025 CDISC+TMF Europe Interchange.



Thank you. Any questions?

abu.sami@mainanalytics.de



www.mainanalytics.de

