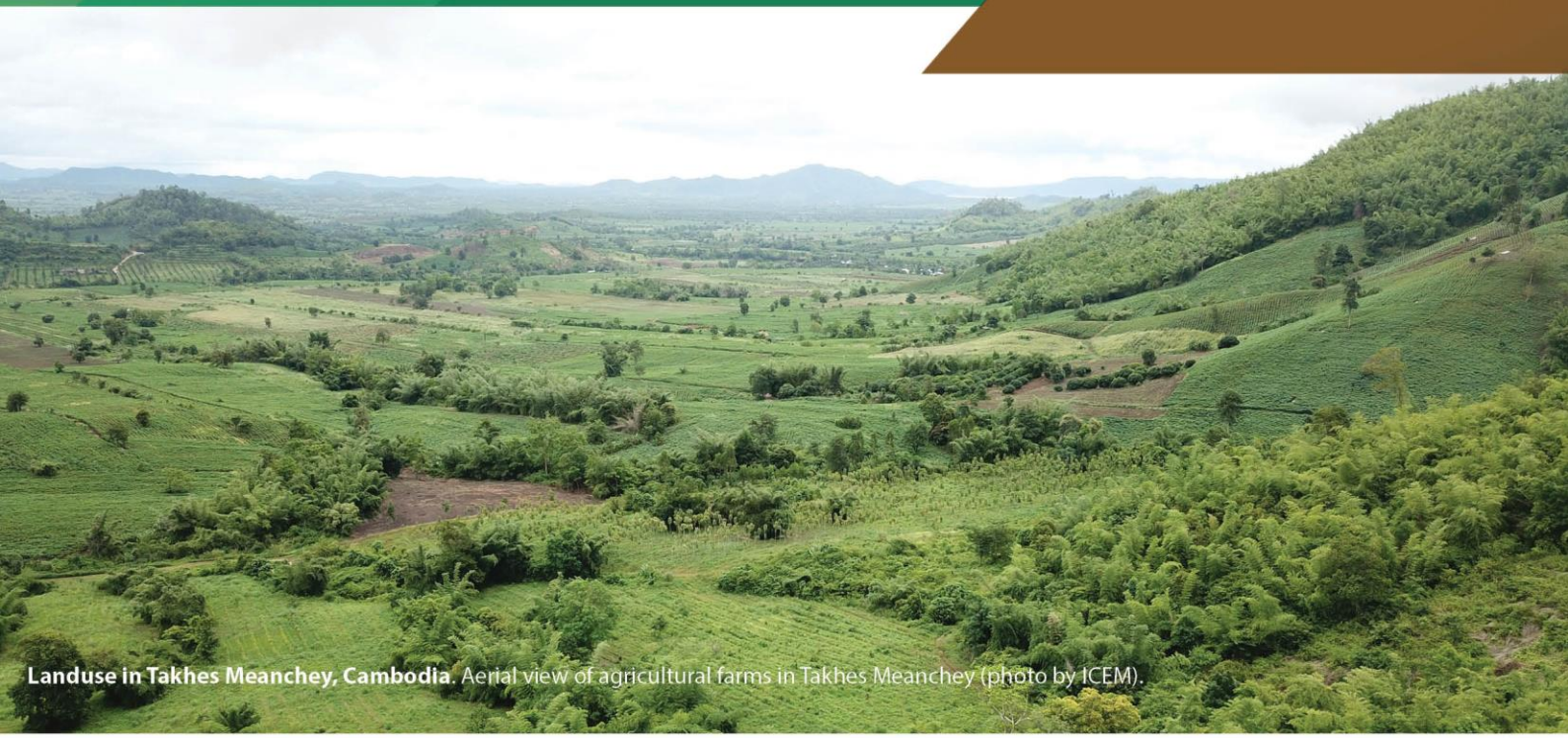
Administrative Manual: Sangker River Basin Decision Support System

Investing in Climate Change Adaptation through Agroecological Landscape Restoration: A Nature-Based Solution for Climate Resilience
(Technical Assistance 6539)

May 2024



Landuse in Takhes Meanchey, Cambodia. Aerial view of agricultural farms in Takhes Meanchey (photo by ICEM).

ADB

JFPR Japan Fund for Prosperous and Resilient Asia and the Pacific

From the People of Japan

icem

World Agroforestry

DISCLAIMER

This document was prepared for the Asian Development Bank (ADB) by a joint venture of the International Centre for Environmental Management (ICEM) and World Agroforestry (ICRAF). The views, conclusions, and recommendations in this document are not to be taken to represent the views of ADB.

Prepared by	ICEM and ICRAF
Prepared for	ADB
Suggested citation	ICEM and ICRAF. 2024. *Administrative Manual: Sangker River Basin Decision Support System (Knowledge Product 8).* Investing in Climate Change Adaptation through Agroecological Landscape Restoration: A Nature-Based Solution for Climate Resilience (Technical Assistance 6539). Prepared for Asian Development Bank. Hanoi.
Deliverable summaries	TA-6539 Investing in Climate Change Adaptation through Agroecological Landscape Restoration: A Nature-based Solution for Climate Resilience led to the preparation of the following knowledge products: • KP1 (1): Landscape Restoration Country Profile: Philippines • KP1 (2): Landscape Restoration Country Profile: Cambodia • KP3: Business Models to Encourage Private Sector Participation in Sustainable Land and Forest Landscape Management • KP4 (1): Climate Change Risk and Adaptation Options Assessment – Sangker River Basin, Cambodia • KP4 (2): Climate Change Risk and Adaptation Options Assessment – Manupali Watershed, Mindanao River Basin, the Philippines • KP5: Good practices manual on biodiverse forest and landscape restoration • KP6: Community-based Climate Vulnerability Assessment and Adaptation Planning for Resilient Agroecosystems • KP 7: Applying Advanced Technologies in Support of Landscape Restoration and Climate Change Adaptation • KP8 (1): User Manual: Sangker River Basin Decision Support System • KP8 (2): User Manual: Mindanao/Manupali River Basin Decision Support System • KP8 (3): Admin Manual: Sangker River Basin Decision Support System • KP8 (4): Admin Manual: Mindanao/Manupali River Basin Decision Support System • KP9 (1): Restoration plans for demonstration areas in Cambodia and the Philippines • KP9 (2): Gender and Social Inclusion in the Mindanao River Basin, the Philippines, and the Sangker River Basin, Cambodia • KP9 (3): Cost-Benefit Analysis of Landscape Restoration Measures in Sangker River Basin, Cambodia • KP9 (4): Cost-Benefit Analysis of Landscape Restoration Measures in Manupali Watershed, Philippines • KP 10: Integrating the principles of ecological agriculture into upland farming systems of Manupali Watershed, the Philippines
Project Team	**ICEM-ICRAF** Jeremy Carew-Reid, Caroline Duque-Pinon, Enrique Lucas Tolentino, Jr., Khun Bunnath, Heng Bauran, Jago Penrose, Lay Chanthy, Michael Waters, Mark Hopkins, Nguyen Phuong Thao, Orlando Fernando Balderama, Paulo Pasicolan, Porny You, Rachmat Mulia, Richard Cooper, Zarrel Gel M. Noza
Photo credit	Front page: **Landuse in Takes Meanchey, Cambodia**. Aerial view of agricultural farms in Takhes Meanchey (photo by ICEM). Back page: **Aerial view of agricultural farms in Dontret, Cambodia**. Drone photo taken in the 3rd field mission to pilot farms (photo by ICEM).

Abbreviations

ADB	Asian Development Bank
ICEM	International Centre for Environmental Management
ICRAF	World Agroforestry
TA	Technical assistance
KP	Knowledge product
DSS	Decision support system
MoE	Ministry of Environment
ODC	Open Development Cambodia
RGC	Royal Government of Cambodia
SDI	Spatial data infrastructure
OGC	Open geospatial consortium
IDE	Integrated development environment
WMS	Web map services
RAM	Random-access memory
CPU	Central processing unit
URL	Uniform resource locator

Contents

1. Introduction to Technical Assistance 6539

This ADB Knowledge and Support Technical Assistance (TA) project contributes to the implementation, integration, and future scaling of landscape restoration measures in the region by developing, demonstrating, and documenting learnings from an innovative set of project activities in target watersheds in Cambodia and the Philippines. This TA aims to explore, assess, and promote innovative forest restoration and agroecology interventions for climate change adaptation.

The TA takes an integrated approach to achieve the following project objectives:

- To develop, evaluate, and promote innovative approaches to climate change adaptation through agroecological landscape restoration;
- To strengthen the capacity of communities to restore and manage their climate-resilience landscapes for food and nutrition security;
- To support the member countries with analytical studies at sub-national and community level to understand the degradation and climate change vulnerability context of agriculture-forest lands supporting rural livelihoods;
- To assess and recommend appropriate agroecological, actionable, and cost-effective climate change adaptation interventions via landscape restoration and sustainable landscape management; and
- To contribute to the design of specific investments for climate change adaptation.

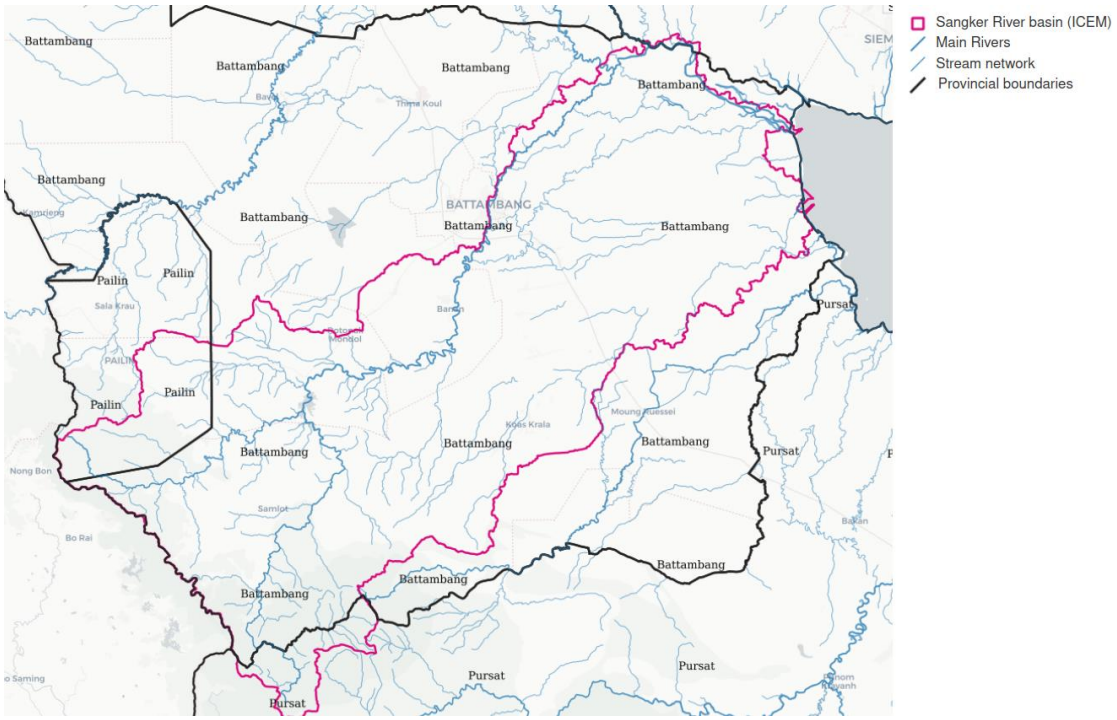
Under Phase III - *Climate adaptation options with forest and agroecological restoration* - a key output is developing a decision support system (KP8) for the Sangker River basin. The purpose of the decision support system (DSS)[1] is to inform agroecological landscape restoration planning for the Sangker River basin. The DSS supports prioritizing restoration sites and adaptation options within the watersheds and additionally provides local and provincial governments with a comprehensive spatial knowledge base for informing sustainable development planning.

In Cambodia the TA-6539 is jointly implemented by the Ministry of Environment (MoE) with support from ICEM, ICRAF and MJP. Natural resources protection, biodiversity conservation, and sustainable resources management fall within the mandate of MoE. The MoE is tasked with identifying and defining protected areas (RGC 2008, article 14).[2] Nine categories of protected areas have been designated within Cambodia through the Protected Areas Law of 2008 and the subsequent declaration of Biodiversity Conservation Corridors in 2017. These include national parks, marine parks, Ramsar sites, wildlife sanctuaries, biosphere reserves, protected landscapes, multi-purpose-use management areas, natural heritage sites, and biodiversity conservation corridors in collaboration with other sectors such as the Ministry of Agriculture, Forests and Fisheries. [3] 69 protected areas cover almost 40% of Cambodia's total geographical area.

1.1. TA-6539 Decision Support System

[1] A *Decision-Support System (DSS)* refers to an information management tool or system that brings together data and information to assist with analysis and decision-making. It integrates and visualizes various datasets, spatial and non-spatial, and presents information through an intuitive platform with interactive functionalities.
[2] RGC (2008) Protected Area Law. January 2018. Royal Government of Cambodia. https://portal.mrcmekong.org/assets/v1/documents/Cambodian-Law/-Protected-Areas-Law-(2008).pdf. Last accessed 23 Nov 2023.
[3] ODC (2016) Protected areas. Open Development Cambodia. https://opendevelopmentcambodia.net/topics/protected-areas/. Last accessed 23 Nov 2023.

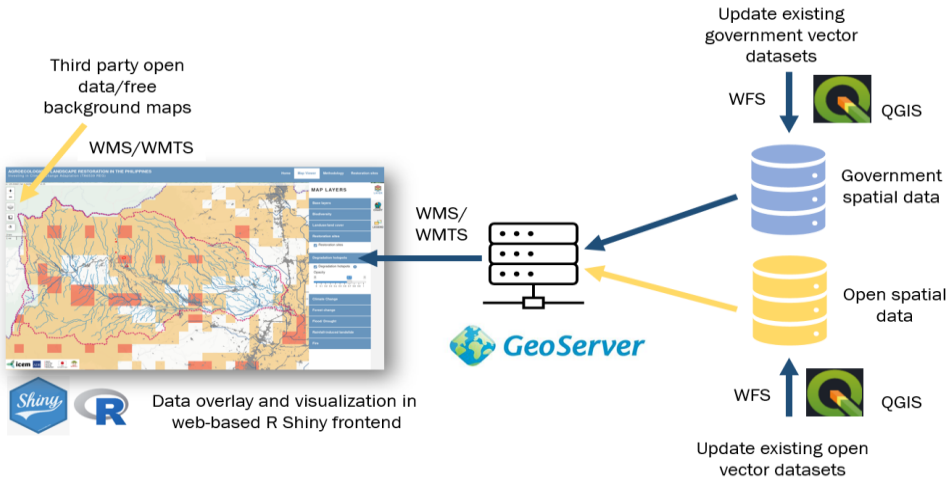
Figure 1: Sangker River Basin, Cambodia



Through discussions with the MoE during project implementation, it was agreed to develop a Decision Support System (DSS) that focuses on the identification of restoration sites and serves to enhance data visualization and access to datasets relevant to the Sangker River basin (Figure 1). The DSS comprises two main components: a GeoServer[4] backend, which connects to a database of geospatial files derived from open online sources and the RGC, and a frontend map viewer developed using the R Shiny web-based framework[5] (Figure 2).

The DSS developed under this project complements earlier DSS developed for the MoE (e.g., LISA[6], CAM-MeDiA[7], Cambodia Climate Change Toolbox[8]) and aims to develop spatial data infrastructure (SDI) to facilitate the identification of restoration sites and to improve access to data and information for supporting the sustainable management of water resources at the river basin scale.

Figure 2: Schematic diagram of decision support system software components and data flow



[4] http://geoserver.org
[5] https://shiny.rstudio.com
[6] https://lisa.icem.com.au
[7] https://camatlas.icem.com.au
[8] https://dss.icem.com.au/cambodiadss

This DSS facilitates the consolidation of key data resources of the Sangker River Basin and facilitates their use for identifying sites for restoration and supporting river basin management. The platform makes use of the well-known GeoServer application as a database backend, and a frontend interface has been developed using the R Shiny web-based framework. Both software components are open source and have strong user and developer communities. The GeoServer application provides functionality for sharing spatial data services compliant with international Open Geospatial Consortium (OGC) protocols (such as WMS, WMTS, and WFS). In addition to the DSS developed in this project that will enable users to explore river basin data, the MoE and other stakeholders can leverage these spatial data services to access geospatial data through desktop GIS applications or for inclusion in other web-based platforms.

In summary, the objectives of the DSS are as follows:

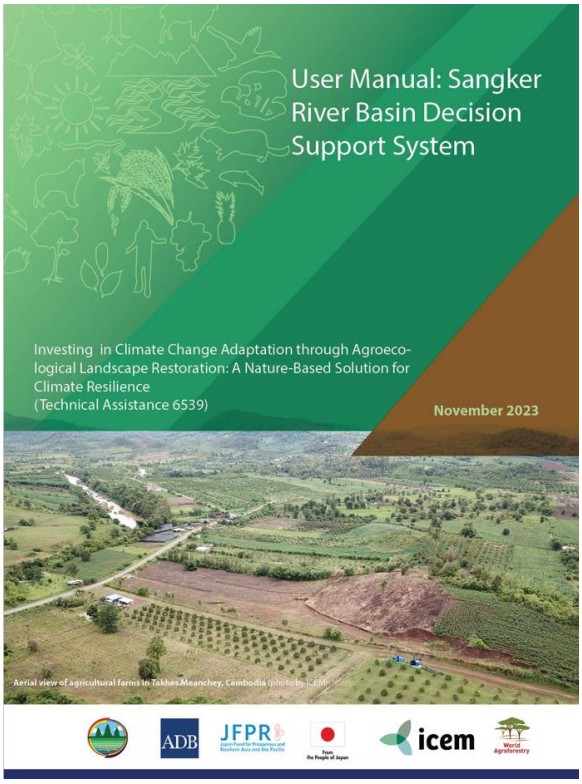
- Identify areas in need of restoration
- Support Sangker River basin planning
- Define spatial distribution of hazards
- Provide a single point of access for all data for use by development planners
- Facilitate multicriteria analyses
- To demonstrate a tool that has the potential to be scaled to support the planning and management of all river basins in Cambodia.

The URL of the official version of the DSS will be finalized after its deployment to a government server. However, the final version of the DSS can also be accessed on ICEM's web server at: **https://agrocam.icem.com.au**

1.2. User Manual

A *User Manual* (2023) (Figure 3) was prepared and submitted separately. The purpose of the *User Manual* is to provide guidance to DSS users, including MoE and other stakeholders, in identifying restoration sites and supporting planning in the Sangker River basin.

Figure 3. TA-6539 User Manual: Sangker River Basin Decision Support System



2. System Administration Manual

The purpose of this **System Administration Manual**, referred to otherwise as the **Manual,** is to provide guidance to IT and data management specialists in managing the data and software components of the system. This document provides guidelines for deploying the DSS web-based platform, including all its software components and associated data sets and subsequent maintenance.

Note that it was not possible to deploy the final app to a government server nor provide training during project implementation. However, training materials were prepared (see Appendices I to IV), and the software, application and data shared online (see below), which should provide adequate guidance for a trained system administrator to deploy and manage the application after project closure. Following deployment, this Manual should be updated accordingly with the app's configuration details (i.e., sections 2.1, and appendices V and VI).

All training materials related to deployment are presented in the Appendices. Section 2.1 summarizes the web addresses of installed DSS components on the ICEM web server,[9] section 2.2 gives an overview of installed software components, and Section 2.3 provides an account of subsequent tasks for maintaining an operational platform.

2.1. Summary of Web Addresses of Installed Decision Support System components

The DSS application components are currently accessible[10] from the ICEM server as follows:

Shiny app (production mode): https://agrocam.icem.com.au
GeoServer: https://geoagrocam.icem.com.au/geoserver

2.2. Overview of Software Components

The DSS application frontend is built on the R software environment using the Shiny R package.[11] 'R' is a software environment built for statistical analysis and visualization of data.[12] R is widely used worldwide and beneficially its functionality can be substantially extended via the installation of numerous software 'packages', of which 'Shiny' is the Web Application Framework package for R. R and Shiny are both open source, meaning there are no licensing restrictions on their use or sharing, no purchasing costs, and the source code can be viewed and edited.[13] There are many active users and an online community of support. The publication of peer-reviewed papers frequently supports the release of R packages. Shiny uses the 'Leaflet' JavaScript library for displaying interactive maps, which is widely used elsewhere, including by OpenStreetMap.[14]

In addition to the R and Shiny components, GeoServer and RStudio are used, the latter for providing an integrated development environment (IDE) for coding (Table 1). GeoServer is an open-source web server that allows the display, sharing, and editing of geospatial data. It uses open standards set by the OGC and is widely used and supported software. The RAMS uses GeoServer's web map services (WMS) to share images of selected raster and vector data. RStudio Server is a coding platform for the application's frontend, which uses the R Shiny web application framework.

[9] This section should be updated when the DSS is deployed to the MoE server
[10] Check with MoE on the final URL addresses when deployed to a government server
[11] https://shiny.posit.co/
[12] https://www.r-project.org
[13] Shiny as a whole is released under GNU General Public License (GPL), version 3, but also includes other open-source licences for its component parts. URL: https://github.com/rstudio/shiny/blob/master/LICENSE
[14] http://www.openstreetmap.org

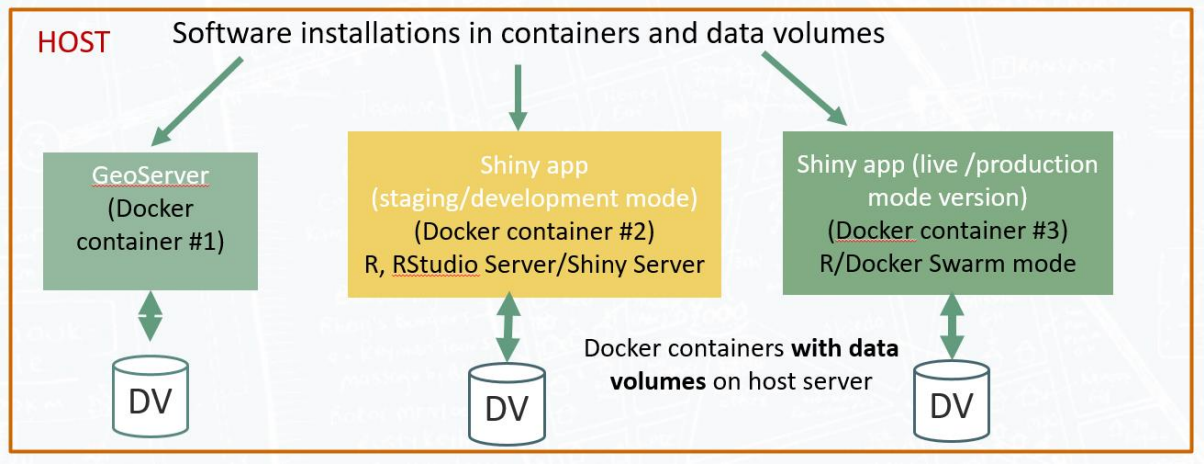
Table 1. Summary of software components and their functions in the decision support system

Software	Description	Function in the DSS
Shiny	Open-source package to build interactive web apps using the R coding language https://shiny.posit.co/	Frontend visualization of data accessible through a web browser
GeoServer	Open-source server for sharing geospatial data, compliant with international OCG protocols[15] https://geoserver.org	Database backend, generating spatial data services of the assets that can be shared online
RStudio Server	RStudio Server is an open-source platform for coding, which is accessible through a web browser. https://posit.co/products/open-source/rstudio-server/	This is the software platform used to code the Shiny frontend.
Docker Engine	Docker Engine is open-source containerization technology https://www.docker.com/community/open-source/	Docker containers contain the key DSS software components and are used to simplify deployment. They are also needed to set up the Shiny frontend's production mode.
QGIS	QGIS software is a popular and widely used GIS application that can be used to create, edit, visualise, analyze and publish geospatial information https://www.qgis.org/	Used to connect to the GeoServer for the purpose of accessing and updating data.

The DSS application makes use of Docker containerization (using the Ubuntu operating system - Ubuntu 22.04). Docker software enables software to be packaged within a virtual container: GeoServer, R, RStudio Server, and Shiny Server are installed within containers. Docker facilitates the migration of the application to other machines and is used to scale up the Shiny web application in production mode. Docker containers are lighter on computer resources than virtual machines such as VMWare and VirtualBox and can be run on Linux and Windows machines.

The DSS application makes use of three Docker containers (Figure 4). Associated datasets and databases (called data volumes in Docker terminology) are hosted directly on the server and linked to the Docker containers (GeoServer datasets and the R Shiny application).

Figure 4: Deployment of the web-based DSS using Docker virtual containers



[15] http://ogc.org/about

2.3. Decision Support System Deployment

The following section summarizes the key steps for deploying the DSS web-based application.

2.3.1 Download Decision Support System Software and Data

All software and data were made accessible to the MoE for download from the following link:

Link: https://1drv.ms/f/s!AoHzL3uXbH31hKtXMAXxdFPUz8CUGQ?e=aWGXZW

Password: workshop_sangker

2.3.2 Key Deployment Steps

Detailed steps for deployment are provided in the training materials as included in appendices I to IV, with an overview provided below:

1. Installation of Docker software (Annex I)

   This presentation gives an overview of the overall software framework and associated data inputs and presents in detail the installation of Docker software on Windows and Linux machines.

2. Deployment of GeoServer (Annex II)

   This presentation gives an overview of the key features of GeoServer and its deployment (Figure 5). There are two components to migrate: the Docker image containing GeoServer and the geospatial database that is linked to GeoServer. Detailed configuration steps are provided, as well as how to access and update geospatial data linked to GeoServer using the QGIS desktop application (Figure 6).

Figure 5. Layer preview of data integrated into the GeoServer application

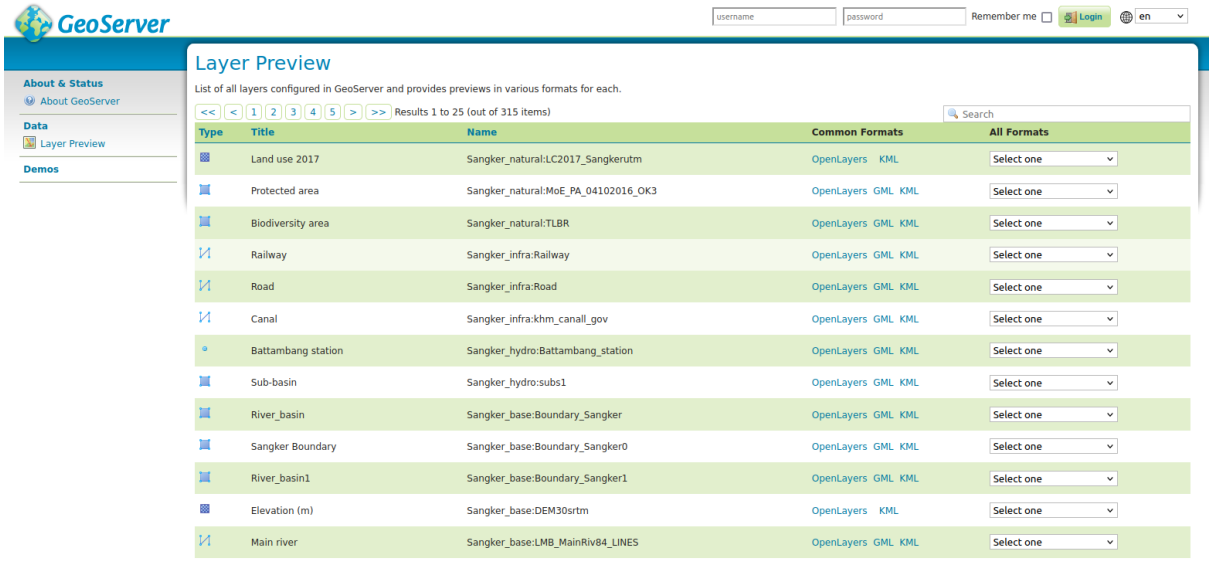
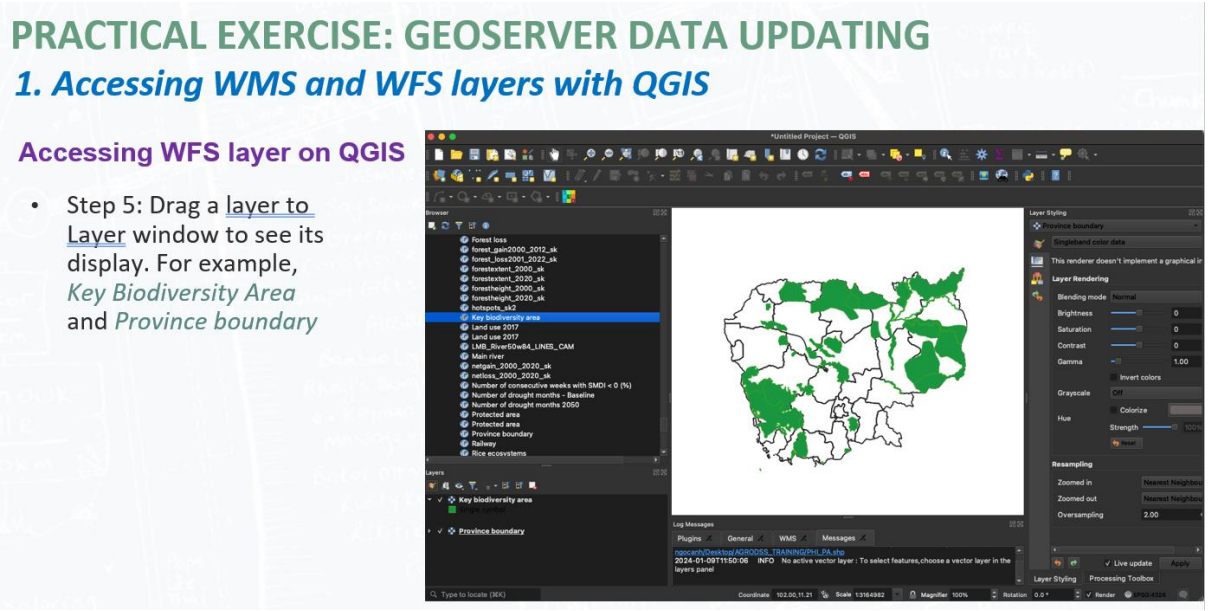


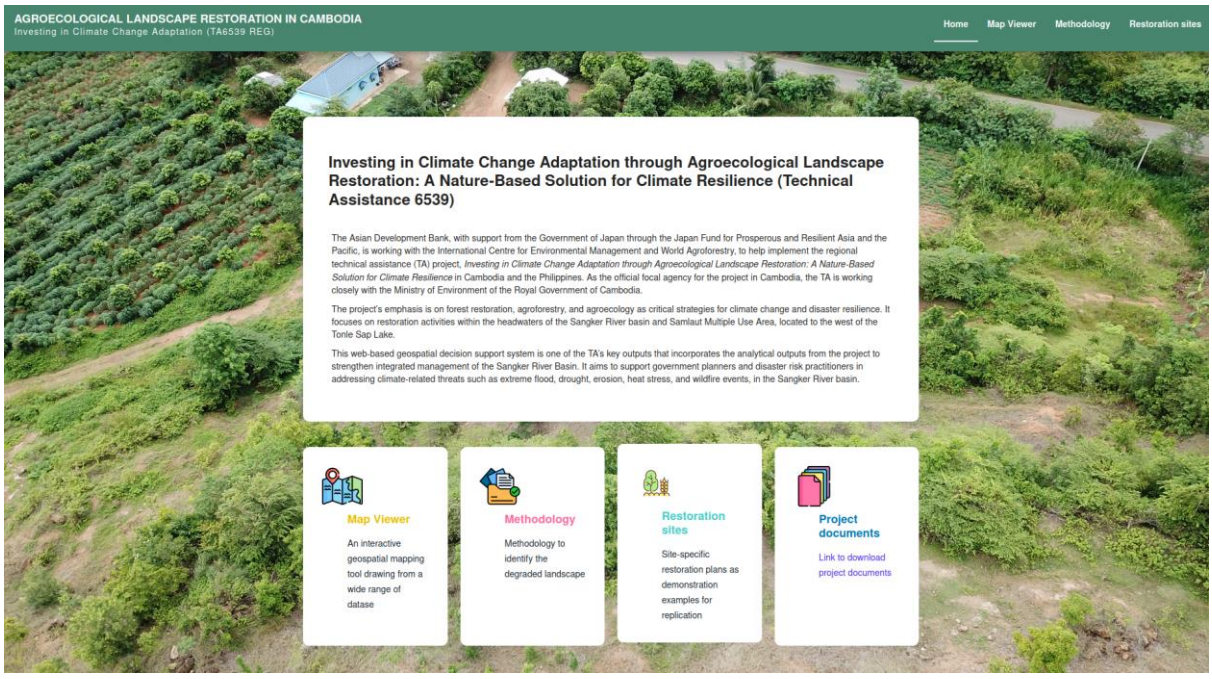
Figure 6. Accessing and updating of DSS geospatial data using QGIS



3. Deployment of production mode of Shiny application (Annex III)

   The deployment of the Shiny application in production mode uses Docker swarm functionality, which importantly enables the Shiny app to be scaled up for more concurrent users. The presentation can be found in Annex III. Figure 7 shows the DSS' home page.

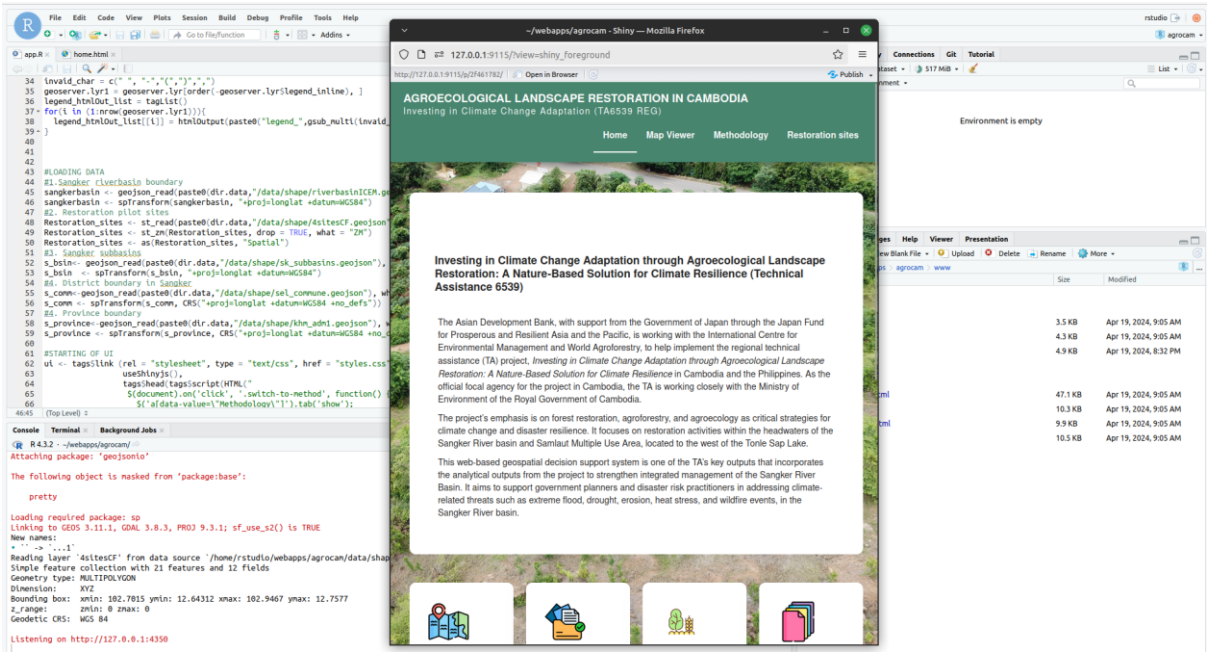
Figure 7: DSS home page



4. Deployment of the development mode of the Shiny application (RStudio Server and Shiny Server) (Annex IV)

The presentation focuses on setting up the Shiny application in development mode using RStudio Server (and Shiny Server, which can be used to display the app online) (Figure 8). Step-by-step guidance is provided on deploying the Docker container. Given that only one version of R can be installed on a given machine, the use of Docker enables the operation of multiple versions of R on a single machine.

Figure 8. Main window of RStudio Server with running DSS



5. Final configuration tasks, including for the reverse proxy (using Traefik), can be added later when the app is deployed (Annex V and VI).

The final configuration steps (to be later added in appendices V and VI) will provide details of the reverse proxy that uses Traefik.[16] Traefik has been designed to work with microservices as provided by Docker and Docker swarm.

It is recommended that deployment steps 1 to 5 are followed sequentially. While step 4 is considered optional – RStudio Server is required for future development of the DSS application - it is recommended as it allows system administrators to better understand the nature of the R Shiny frontend application. It is recommended that future development of the Shiny application be initially conducted using RStudio Server installed on a laptop. Then the release version migrated to the RStudio Server online for sharing and review.

2.4. Decision Support System Maintenance

The following sections detail potential solutions for addressing s issues encountered in running the DSS, how to start the DSS software components, security, system monitoring and backups, and future development.

2.4.1 Potential Problems and Responses

Typically, all components of the DSS application should reliably run continuously without trouble, but if changes are made to the server, issues may arise and need resolving. The following lists some issues that could potentially arise while running the DSS application and responses to address them.

Problem: The Shiny application is becoming slow to respond.

Response:
(i) Check the amount of RAM and CPU resources being used by the Docker containers (and contained apps):

```
$ docker stats --no-stream
```

[16] https://github.com/traefik/traefik

(ii) This may also occur if more users are accessing the application. Modify the number of replicas in the **docker-compose-shinyapp.yml** and consider adding more RAM/CPU resources as more replicas will use more computer resources.

For example, to deploy 10 replicas, modify the docker-compose yml file:

```
deploy:
  replicas: 10
```

Then redeploy the Shiny app service (docker service rm … / docker stack deploy …)

Problem: The Shiny application becomes inaccessible or unresponsive

Response:

(i) Redeploy the Docker Shiny app service

- Check Docker service is running:
  $ docker service ls
- Stop both Traefik and Shiny app services:
  $ docker service rm [NAME OF SERVICE]
- Redeploy Traefik and Shiny app services:
  Change to Shiny app directory containing docker-compose yml files:
  agrophp $ docker stack deploy –c docker-compose.traefik.yml traefik
  agrophp $ docker stack deploy –c docker-compose.shinyapp.yml agrocam
- Check services are running:
  $ docker service ls
- Check containers are running:
  $ docker ps

If problem persists:

Response:

- Check service logs:
  $ docker service logs [SERVICE NAME]
- Restart the Docker service (**note: this will stop all containers**)
  Windows server: open PowerShell as admin: restart-service *docker*
  Linux: $ sudo service docker restart or sudo systemctl restart docker

Remember to restart other previously running containers as a Docker service restart will stop all running containers (e.g., add to Linux cronjob/Windows restart for automatic restarts)

NOTE: If no changes have been made to the Docker images and docker-compose yml files, then check for changes to the general server configuration.

The above responses can also be tried regarding the GeoServer application.

Problem: Maps become slow to display in the map viewer and GeoServer application

Response:

(i) Check the amount of RAM and CPU resources being used by the Docker container (containing GeoServer):

$ docker stats --no-stream

(ii) This may also occur if more users are accessing the application. Modify the number of replicas (also called containers) in **docker-compose-geoserver.yml** and consider adding more RAM/CPU resources to the server as more replicas will use more resources.

For example, to deploy 2 replicas, modify the docker-compose yml file:

```
deploy:
  replicas: 2
```

Then redeploy the GeoServer service (docker service rm … / docker stack deploy …)

Problem: If GeoServer becomes inaccessible or unresponsive (maps not showing in map viewer)

Response:

(i) Redeploy the Docker GeoServer service

- Check Docker service is running:
  $ docker service ls
- Stop both Traefik and GeoServer services:
  $ docker service rm [NAME OF SERVICE]
- Redeploy Traefik and GeoServer services:
  Change to Shiny app directory containing docker-compose yml files:
  $ docker stack deploy -c docker-compose.traefik.yml agroecolcam
  $ docker stack deploy -c docker-compose.geoserver.yml geo
- Check services are running: $ docker service ls
  Check containers are running: $ docker container ls

Problem: The Shiny application is running but the map layers are not displaying

Response:

(i) Check that the GeoServer Docker container is running by visiting the application's online home page (see link above) and checking that layers are accessible (Click on LayerPreview > OpenLayers link).

Restart the docker container if required (e.g., docker service rm … / docker stack deploy …)

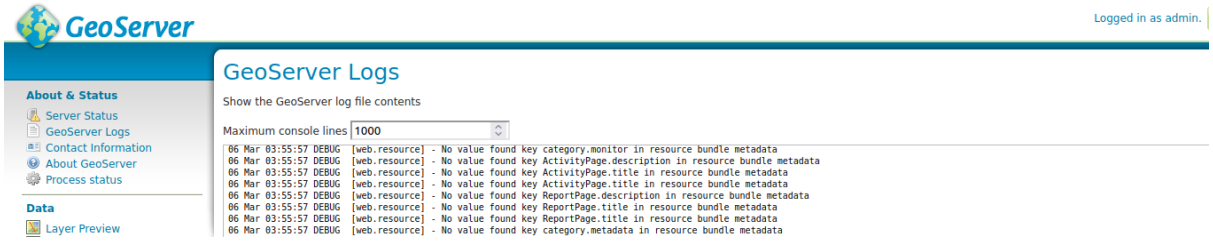
(ii) If the GeoServer application has been migrated to another machine, check that the GeoServer link is correctly indicated in the Shiny app.R file and the application has been mapped to port 443.

If a problem persists:

Response:

- Check service logs:
  $ docker service logs [SERVICE NAME]
- Check logs in GeoServer application (log in as admin) (e.g., Figure 9)

Figure 9. GeoServer logs



- Restart the Docker service (**note: this will stop all containers**)
  Windows server: open PowerShell as admin: restart-service *docker*
  Linux: $ sudo service docker restart or sudo systemctl restart docker

Remember to restart other previously running containers as a Docker service restart will stop all running containers (e.g., add to Linux cronjob/Windows restart for automatic restarts)

NOTE: If no changes have been made to the Docker images and docker-compose yml files, then check for changes to the general server configuration.

Other information resources

There are various online resources, including active communities of users and developers. Here are some of the official information resources:

- **Docker**: https://docs.docker.com
- **GeoServer**: https://docs.geoserver.org
- **R**: https://cran.r-project.org/manuals.html
- **RStudio**: https://posit.cloud/learn/primers
- **Shiny web application framework**: https://shiny.rstudio.com/tutorial
- **Shiny Server**: https://docs.rstudio.com/shiny-server
- **Traefik**: https://github.com/traefik/traefik

2.4.2 Security

It is important to maintain security updates for the operating system of the government host server.

Ideally, the Ubuntu operating system (version 22.04) used in the Docker containers and the container RAMS software components should also be updated regularly (e.g., every 12 months). It is advisable to **back up the Docker containers before any update** in case of any issue arising that prevents an application from subsequently running. If upgrading GeoServer, R, RStudio Server, or Shiny Server, always check the release notes of the new software versions for any additional modifications that may be required, e.g., a major upgrade for GeoServer may (or may not) require the database configuration to be modified.

Note that upgrading R will likely require subsequent updating of the R packages in the DSS application; however, this can be done readily by following the guidelines in the workshop training materials.

It is recommended that software in the Docker containers be updated by using the dockerfiles (as shared in the workshop). After creating a new Docker image, edit the name of the Docker image in the docker-compose yml file.

It is recommended that the IT administrator monitor security updates for each of the key DSS software components and update the application accordingly if a significant security risk is reported. Details on software security can currently be found at the following URLs:

> GeoServer: https://osgeo-org.atlassian.net/projects/GEOS/summary
> e.g., https://osgeo-org.atlassian.net/jira/software/c/projects/GEOS/issues/?jql=project%20%3D%20%22GEOS%22%20ORDER%20BY%20priority%20DESC%2C%20updated%20DESC
>
> Docker: https://docs.docker.com/security/
> R: https://bugs.r-project.org/buglist.cgi?quicksearch=security
> RStudio and Shiny: https://community.rstudio.com/tag/security

2.4.3 System monitoring and backups

As with any web-based system, the DSS application should be continuously monitored to minimize downtime (e.g., by using a tool such as montastic.com). Key operational tasks also require that data are backed up regularly (depending on the frequency of data updates) and that any security patches are applied to the underlying operating system and the DSS software components as required.

Key datasets that require backing up include the GeoServer database. Backups can be made using a utility such as 'rsync' (with the process automated, such as with a cron job scheduler in Linux).

2.5. Future development

It is recommended that future development of the Shiny app, using RStudio Server, be done initially on a laptop computer, and prototypes then deployed online (using RStudio Server and Shiny Server) for sharing and feedback from stakeholders. Note that Shiny Server is only useful for development as it cannot be scaled up to enable access to increasing visitors. Scaling up is achieved using Docker swarm mode, where multiple replicas of Docker can be deployed.

Annexes

- Annex I: Overview of Sangker River basin DSS and its management
- Annex II: Deployment and maintenance of GeoServer (DSS backend)
- Annex III: Deployment and maintenance of R Shiny application (production mode)
- Annex IV: Deployment of R Shiny application (development mode)
- Annex V: Traefik Configuration (TBD)
- Annex VI: Post Deployment Configuration (TBD)

Annex I: Overview of Sangker River Basin Decision Support System and Its Management

Time	Agenda	Presenter/Moderator
Session 1: Overview of the DSS and Docker containerization		
13:30 – 14:00	Workshop and participant introductions	Richard Cooper
14:00 – 14:30	Overview of DSS, software components and data inputs	Richard Cooper
14:30 – 15:00	Overview of Docker engine/containerization	Richard Cooper
15:00 – 15:15	Tea/coffee break	
15:15 – 17:00	Practical exercise: Installation of Docker engine	ICEM team and participants
17:00-17:30	Discussion/questions and wrap-up	ICEM team and participants

SESSION 1: EXPECTED OUTPUTS

- Participants to gain an appreciation of the key software and data components of the application
- Participants to learn about Docker containerization as used to facilitate app migration between machines and setting up the apps in production mode

4

SESSION 1: OUTLINE

1. Download application and presentation files
2. Install WSL and Docker
3. Overview of DSS software framework and data inputs
4. Software installation: Docker containerization
5. Practical exercise: installation of Docker software

5

1. DOWNLOAD APPLICATION FILES AND PRESENTATIONS

Download from cloud
Link: https://1drv.ms/f/s!AoHzL3uXbH31hKtXMAXxdFPUz8CUGQ?e=aWGXZW
Password: workshop_sangker

6

2. INSTALL WSL AND DOCKER ENGINE

7

2. INSTALL WSL AND DOCKER

Two steps:

- Install WSL2
- Install Docker Engine

8

2. INSTALL WSL AND DOCKER

Check Windows version: In PowerShell/command prompt, enter winver

1. Install Windows Subsystem for Linux (WSL2)

(i) For Windows Desktop 10 64-bit (2004 Build 19041 and higher)/
11 64-bit (version 21H2 or higher):
Reference: https://learn.microsoft.com/en-us/windows/wsl/install

Prerequisites for WSL 2:

- 64-bit processor with Second Level Address Translation (SLAT) (next slide).
- 4GB system RAM
- BIOS-level hardware virtualization support must be enabled in the BIOS settings (next slide).
  https://docs.docker.com/desktop/troubleshoot/topics/#virtualization

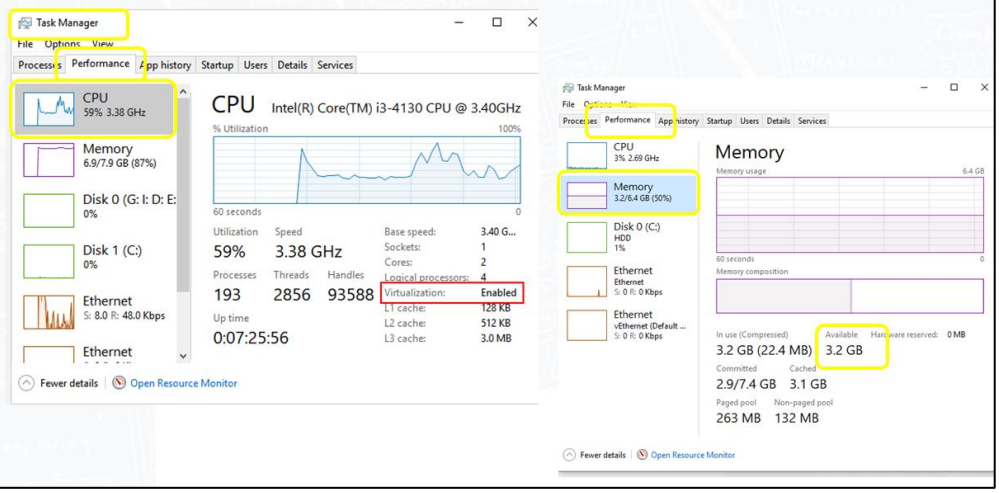
Run this single command (if WSL2 not installed before):
Start menu > PowerShell > right-click > run as Administrator

```
wsl --install
```

Then restart machine

9

WINDOWS 10/11: CHECK VIRTUALIZATION ENABLED



WINDOWS 10: ENABLE VIRTUALIZATION

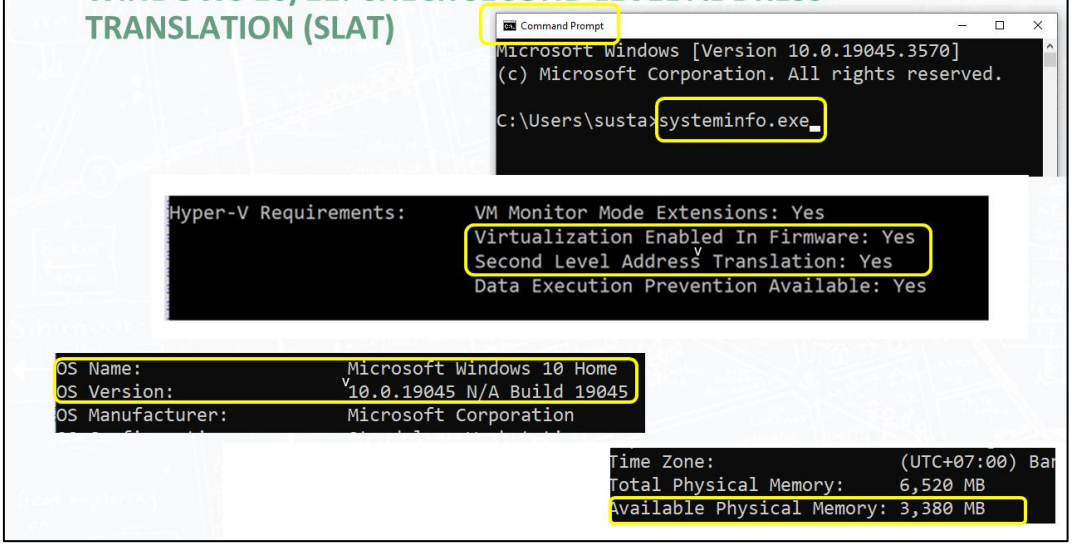
- **Windows 10:**
  - ➢ Enable virtualization in BIOS:
    - – Power on computer.
    - – Press the specific hotkey to enter BIOS. Depending on the make of the machine hotkeys may vary, e.g., Esc, F2.
    - – Navigate to the Advanced tab, press Enter to continue.
    - – Select Virtualization, enable and save.
    - – Reboot computer
    - – Recheck virtualization is enabled

- **Windows 11**:

  ➢https://support.microsoft.com/en-us/windows/enable-virtualization-on-windows-11-pcs-c5578302-6e43-4b4b-a449-8ced115f58e1

11

WINDOWS 10/11: CHECK SECOND LEVEL ADDRESS TRANSLATION (SLAT)



2. INSTALL WSL AND DOCKER

(ii) For older versions of Windows Desktop 10

Reference: https://learn.microsoft.com/en-us/windows/wsl/install-manual

Follow steps 1 to 5 in above reference.

Prerequisites for WSL2:

- For x64 systems: Version 1903 or later, with Build 18362.1049 or later
- 64-bit processor with Second Level Address Translation (SLAT)
- 4GB system RAM
- BIOS-level hardware virtualization support must be enabled in the BIOS settings.
- https://docs.docker.com/desktop/troubleshoot/topics/#virtualization

13

2. INSTALL WSL AND DOCKER

(iii) For Windows Server

https://learn.microsoft.com/en-us/windows/wsl/install-on-server

Start menu > PowerShell > right-click > Run as Administrator

Windows Server 2022 and later:

Run this single command: wsl --install

Then restart machine

Older versions (Windows Server 2019 (version 1709+)):

Run this single command:

Enable-WindowsOptionalFeature -Online -FeatureName Microsoft-Windows-Subsystem-Linux

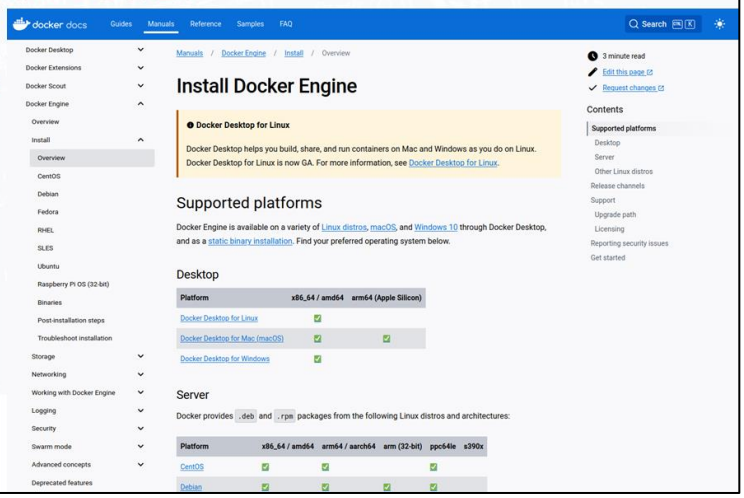
Then Restart machine

14

2. INSTALL WSL AND DOCKER

Supported platforms:
https://docs.docker.com/engine/install/

Windows 10, 11
Windows Server
macOS
Linux distros

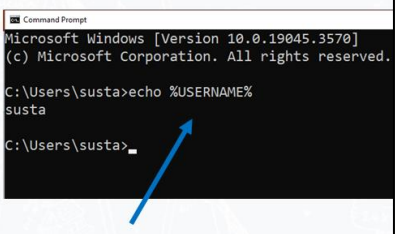


2. INSTALL WSL AND DOCKER

2. Install Docker Engine

(i) For Windows

(a) Download and install Docker Engine (~600 MB):
https://docs.docker.com/desktop/install/windows-install/



Docker Desktop for Windows Download is ~600 MB

(b) Use WSL 2 instead of Hyper-V option on the Configuration page
(c) Add user to docker-users group: $ net localgroup docker-users <user> /add

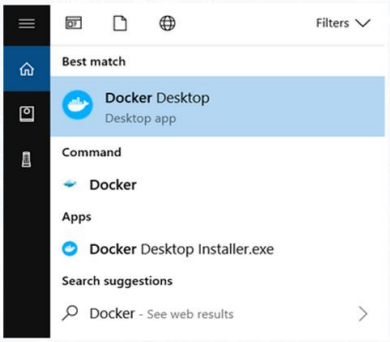
(ii) For macOS and Linux:
https://docs.docker.com/desktop/install/linux-install/

Linux: Add user to sudo group to enable docker without sudo user:
$ sudo groupadd docker
$ sudo usermod -aG docker $USER
$ sudo newgrp docker

16

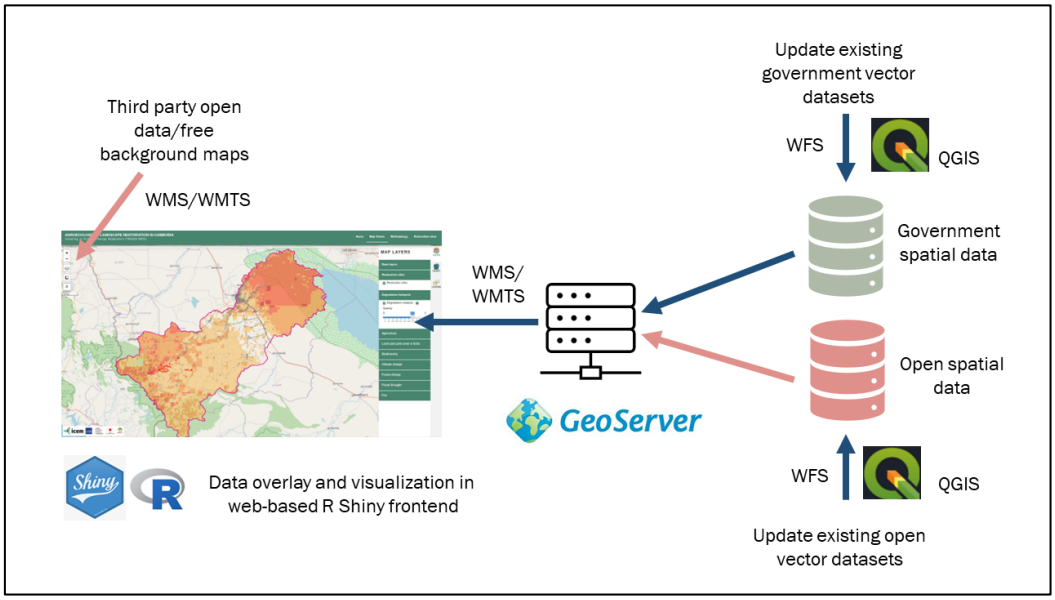
2. INSTALL WSL AND DOCKER

- To Start Docker Desktop, search for 'Docker' and select "Docker Desktop" in search results
- The whale in the status bar stays steady, Docker Desktop is up and running

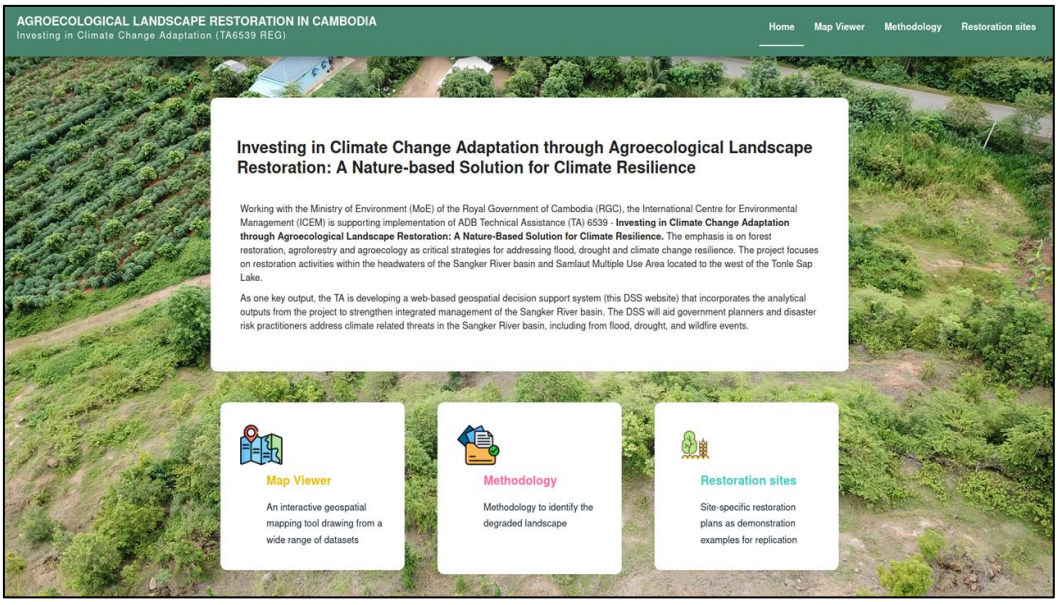


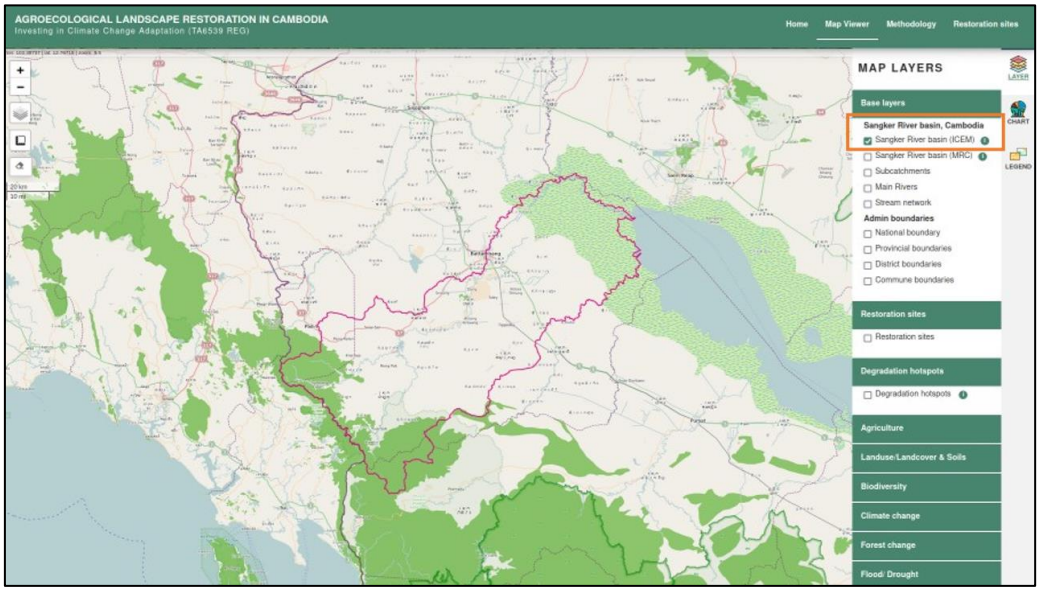
17

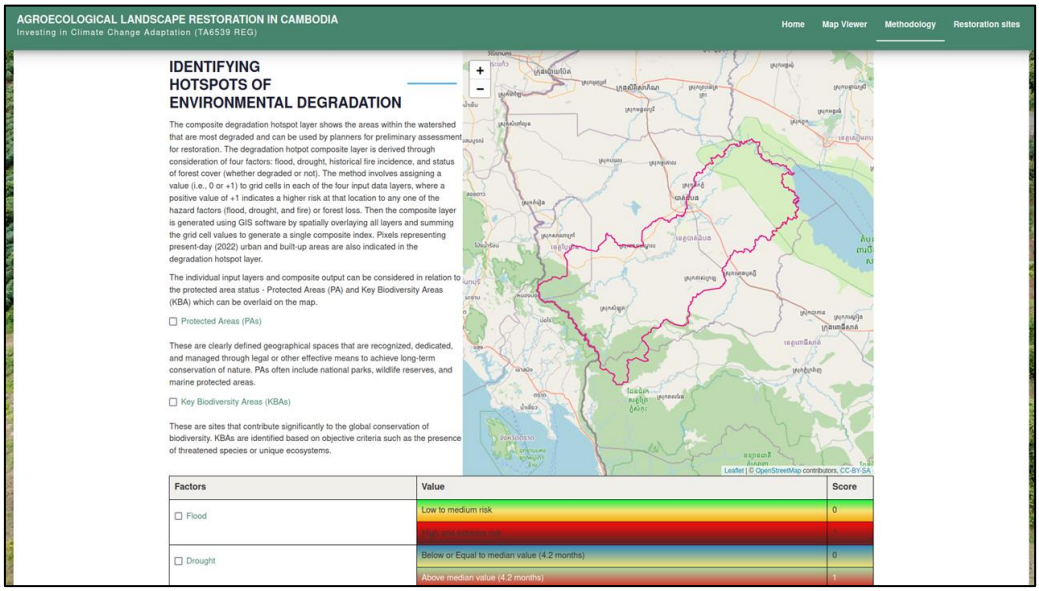
3. OVERVIEW OF DSS SOFTWARE FRAMEWORK AND DATA INPUTS

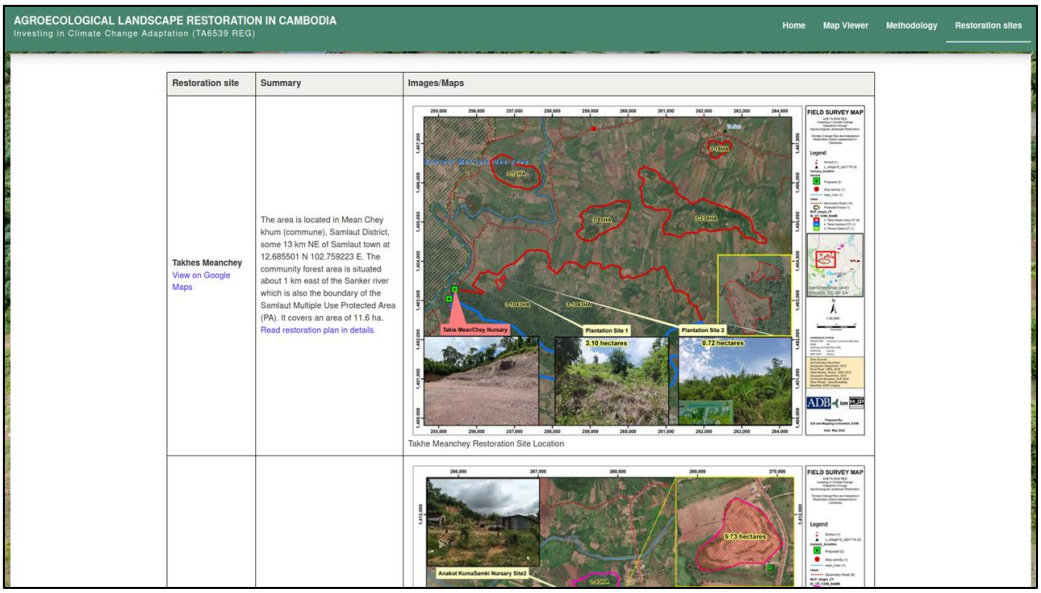

Update existing government vector datasets
WFS
QGIS
Third party open data/free background maps
WMS/WMTS
Government spatial data
WMS/ WMTS
Open spatial data
GeoServer
WFS
QGIS
Data overlay and visualization in web-based R Shiny frontend
Update existing open vector datasets


Development version
On ICEM server at
https://shinydev.icem.com.au/agrocam
User Manual provides detailed guidance on using the DSS for preliminary assessment of potential restoration sites and basin planning
20
20


AGROECOLOGICAL LANDSCAPE RESTORATION IN CAMBODIA
Investing in Climate Change Adaptation (TA6539 REG)
Home
Map Viewer
Methodology
Restoration sites
Investing in Climate Change Adaptation through Agroecological Landscape Restoration: A Nature-based Solution for Climate Resilience
Working with the Ministry of Environment (MoE) of the Royal Government of Cambodia (RGC), the International Centre for Environmental Management (ICEM) is supporting implementation of ADB Technical Assistance (TA) 6539 - Investing in Climate Change Adaptation through Agroecological Landscape Restoration: A Nature-Based Solution for Climate Resilience. The emphasis is on forest restoration, agroforestry and agroecology as critical strategies for addressing flood, drought and climate change resilience. The project focuses on restoration activities within the headwaters of the Sangker River basin and Samlaut Multiple Use Area located to the west of the Tonle Sap Lake.
As one key output, the TA is developing a web-based geospatial decision support system (this DSS website) that incorporates the analytical outputs from the project to strengthen integrated management of the Sangker River basin. The DSS will aid government planners and disaster risk practitioners address climate related threats in the Sangker River basin, including from flood, drought, and wildfire events.
Map Viewer
An interactive geospatial mapping tool drawing from a wide range of datasets
Methodology
Methodology to identify the degraded landscape
Restoration sites
Site-specific restoration plans as demonstration examples for replication


AGROECOLOGICAL LANDSCAPE RESTORATION IN CAMBODIA
Investing in Climate Change Adaptation (TA6539 REG)
Home
Map Viewer
Methodology
Restoration sites
MAP LAYERS
Base layers
Sangker River basin, Cambodia
Sangker River basin (ICEM)
Sangker River basin (MRC)
Subcatchments
Main Rivers
Stream network
Admin boundaries
National boundary
Provincial boundaries
District boundaries
Commune boundaries
Restoration sites
Restoration sites
Degradation hotspots
Degradation hotspots
Agriculture
Landuse/Landcover & Soils
Biodiversity
Climate change
Forest change
Flood/ Drought


AGROECOLOGICAL LANDSCAPE RESTORATION IN CAMBODIA
Investing in Climate Change Adaptation (TA6539 REG)
Home
Map Viewer
Methodology
Restoration sites
IDENTIFYING HOTSPOTS OF ENVIRONMENTAL DEGRADATION
The composite degradation hotspot layer shows the areas within the watershed that are most degraded and can be used by planners for preliminary assessment for restoration. The degradation hotpot composite layer is derived through consideration of four factors: flood, drought, historical fire incidence, and status of forest cover (whether degraded or not). The method involves assigning a value (i.e., 0 or +1) to grid cells in each of the four input data layers, where a positive value of +1 indicates a higher risk at that location to any one of the hazard factors (flood, drought, and fire) or forest loss. Then the composite layer is generated using GIS software by spatially overlaying all layers and summing the grid cell values to generate a single composite index. Pixels representing present-day (2022) urban and built-up areas are also indicated in the degradation hotspot layer.
The individual input layers and composite output can be considered in relation to the protected area status - Protected Areas (PA) and Key Biodiversity Areas (KBA) which can be overlaid on the map.
Protected Areas (PAs)
These are clearly defined geographical spaces that are recognized, dedicated, and managed through legal or other effective means to achieve long-term conservation of nature. PAs often include national parks, wildlife reserves, and marine protected areas.
Key Biodiversity Areas (KBAs)
These are sites that contribute significantly to the global conservation of biodiversity. KBAs are identified based on objective criteria such as the presence of threatened species or unique ecosystems.
Leaflet | © OpenStreetMap contributors, CC-BY-SA
Factors
Value
Score
Flood
Low to medium risk
0
Drought
Below or Equal to median value (4.2 months)
0
Above median value (4.2 months)
1


AGROECOLOGICAL LANDSCAPE RESTORATION IN CAMBODIA
Investing in Climate Change Adaptation (TA6539 REG)
Home
Map Viewer
Methodology
Restoration sites
Restoration site
Summary
Images/Maps
Takhes Meanchey
View on Google Maps
The area is located in Mean Chey khum (commune), Samlaut District, some 13 km NE of Samlaut town at 12.685501 N 102.759223 E. The community forest area is situated about 1 km east of the Sanker river which is also the boundary of the Samlaut Multiple Use Protected Area (PA). It covers an area of 11.6 ha.
Read restoration plan in details
Takhe Meanchey Nursery
Plantation Site 1
Plantation Site 2
3.10 hectares
0.72 hectares
FIELD SURVEY MAP
Takhe Meanchey Restoration Site Location
Arakot KumaSamki Nursary Site2
5.73 hectares

Scaling and upgrading

- Upgrading of existing government (and open) datasets
  - Replace data layer linked to GeoServer backend
- Adding new datasets for display in the map viewer will require further coding (R) of Shiny application
  - R coding is typically done using RStudio Server

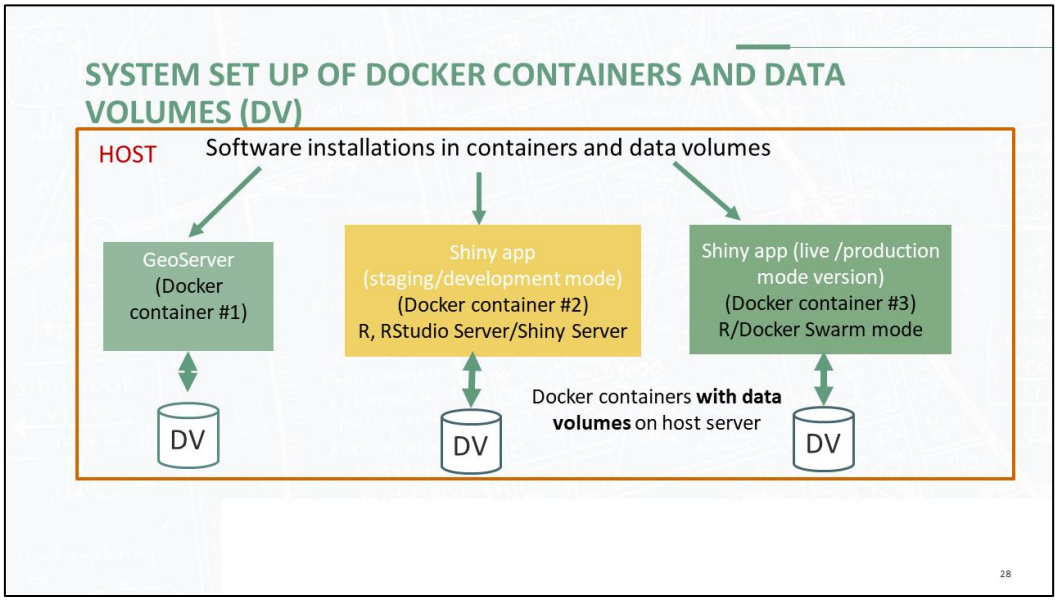
25

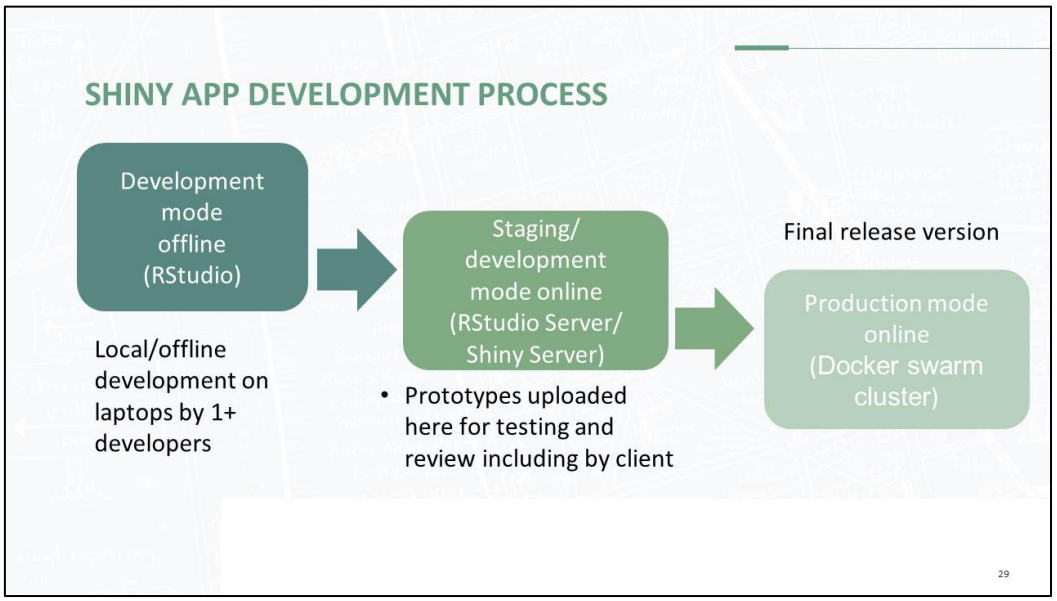
3. SOFTWARE INSTALLATION: DOCKER CONTAINERIZATION

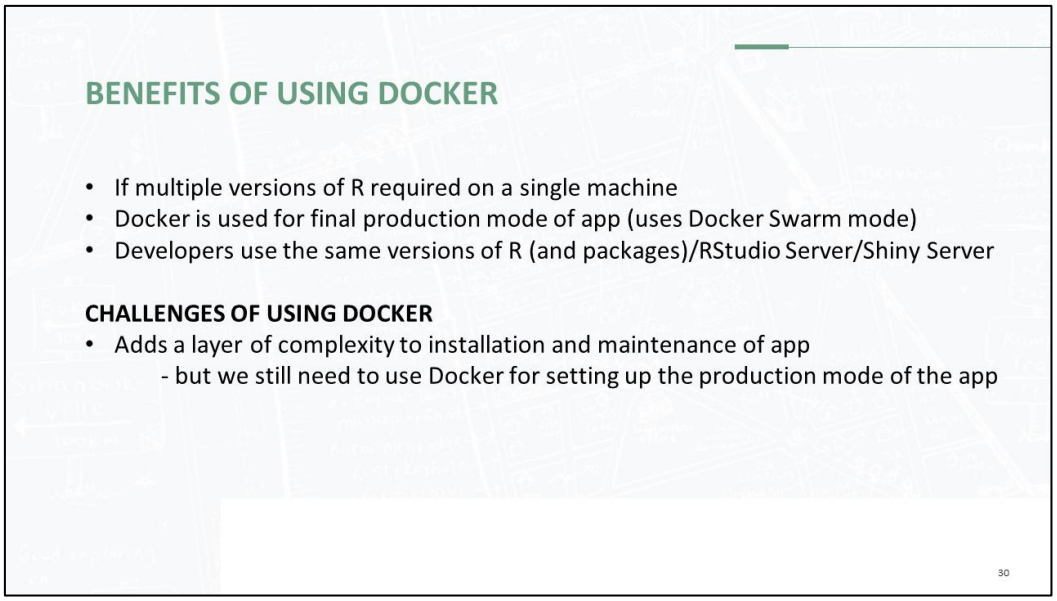
WHAT IS DOCKER?

- Docker software enables software to be packaged within a **virtual container.**
- Docker containers are lighter on computer resources than virtual machines such as VMWare, and VirtualBox
- Docker containerization facilitates the application's installation and migration (https://www.docker.com).
- The use of Docker facilitates migration of the app between servers, without the need to separately install multiple software components on potentially different operating systems.
- The Docker images used for the app contain the Ubuntu operating system, and these can be run on Linux or Windows machines.

27


SYSTEM SET UP OF DOCKER CONTAINERS AND DATA VOLUMES (DV)
HOST
Software installations in containers and data volumes
GeoServer (Docker container #1)
Shiny app (staging/development mode) (Docker container #2) R, RStudio Server/Shiny Server
Shiny app (live /production mode version) (Docker container #3) R/Docker Swarm mode
DV
DV
DV
Docker containers with data volumes on host server
28


SHINY APP DEVELOPMENT PROCESS
Development mode offline (RStudio)
Staging/ development mode online (RStudio Server/ Shiny Server)
Final release version
Production mode online (Docker swarm cluster)
Local/offline development on laptops by 1+ developers
• Prototypes uploaded here for testing and review including by client
29


BENEFITS OF USING DOCKER
• If multiple versions of R required on a single machine
• Docker is used for final production mode of app (uses Docker Swarm mode)
• Developers use the same versions of R (and packages)/RStudio Server/Shiny Server
CHALLENGES OF USING DOCKER
• Adds a layer of complexity to installation and maintenance of app
- but we still need to use Docker for setting up the production mode of the app
30

DOCKER INSTALLATION

- To show how to install Docker Engine <- participants to install Docker Engine on their computers
- Installation on:
  - Laptop (Windows/macOS/Linux) <- this workshop
  - Server <- adopt overall same approach as on laptop

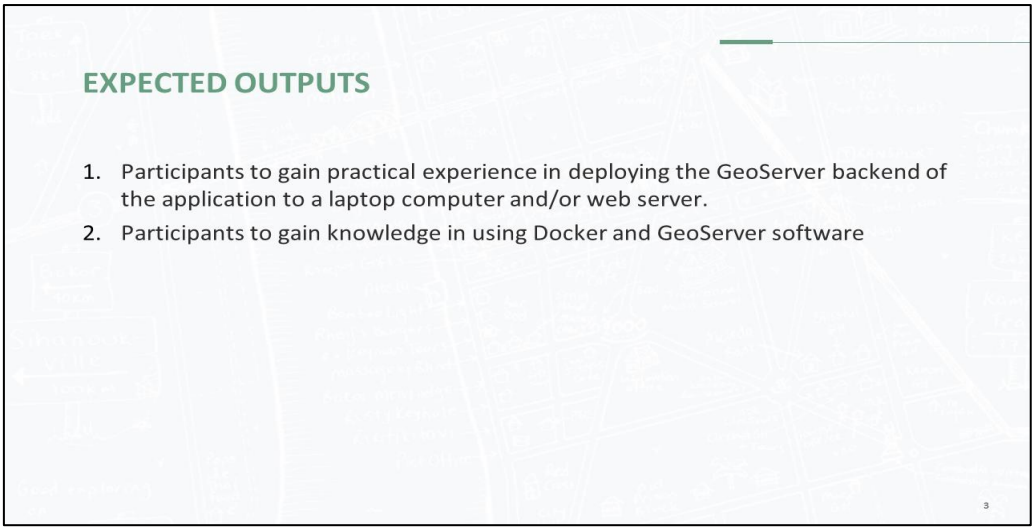
31

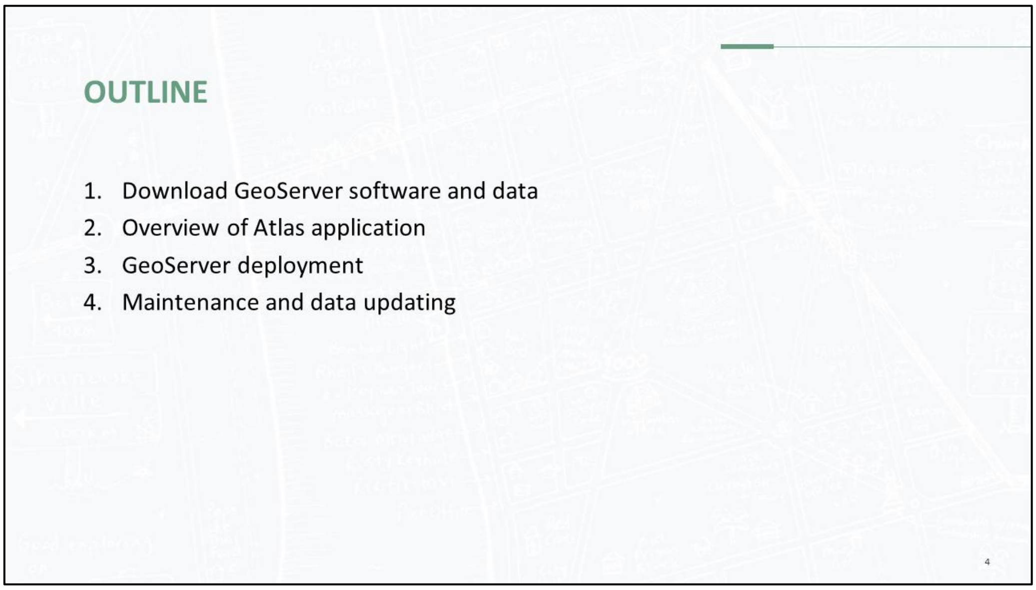
4. PRACTICAL EXERCISE:

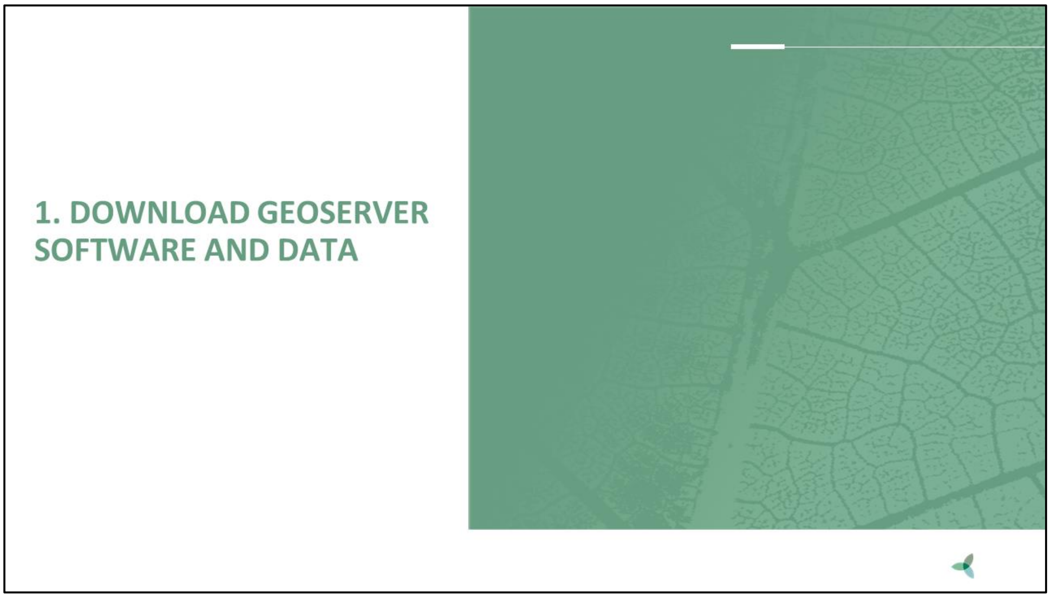
COMPLETE INSTALLATION OF DOCKER ENGINE

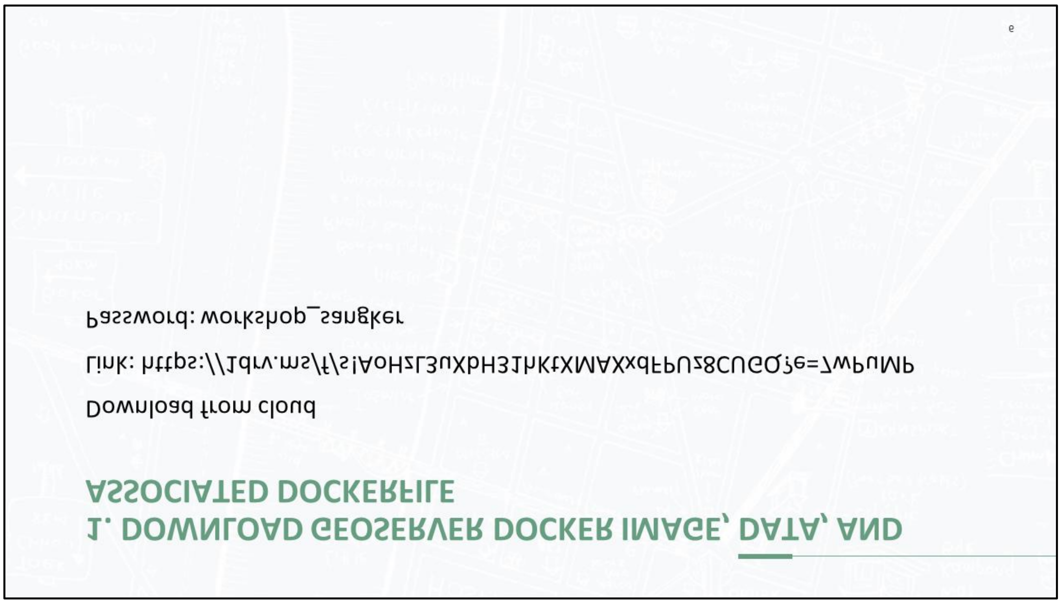
Annex II: Deployment and Maintenance of GeoSever (Decision Support System BackEnd)

Time	Agenda	Presenter/Moderator
Session 2: Deployment and maintenance of GeoServer/data updating		
13:30 – 14:00	Introduction to GeoServer deployment and maintenance	Richard Cooper
14:00 – 15:00	Practical exercise: GeoServer deployment	ICEM team and participants
15:00 – 15:15	Tea/coffee break	
15:15 – 16:00	Practical exercise: GeoServer deployment	ICEM team and participants
16:00 – 16:15	Overview of GeoServer data updating	Ngoc Anh
16:15 – 17:15	Practical exercise: GeoServer data updating	Ngoc Anh and participants
17:15 - 17:30	Discussion/questions and wrap-up	ICEM team and participants




OUTLINE
1. Download GeoServer software and data
2. Overview of Atlas application
3. GeoServer deployment
4. Maintenance and data updating
4


1. DOWNLOAD GEOSERVER SOFTWARE AND DATA

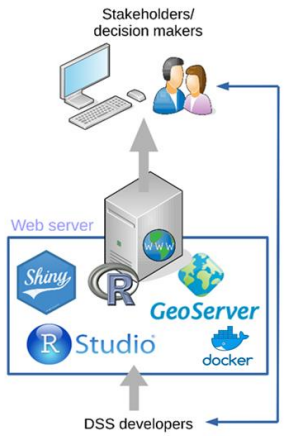

1. DOWNLOAD GEOSERVER DOCKER IMAGE, DATA, AND ASSOCIATED DOCKERFILE
Download from cloud
Link: https://1drv.ms/f/s!AoHzL3uXbH31hKtXMAXxdFPUz8CUGQ?e=7wPuMP
Password: workshop_sangker
6

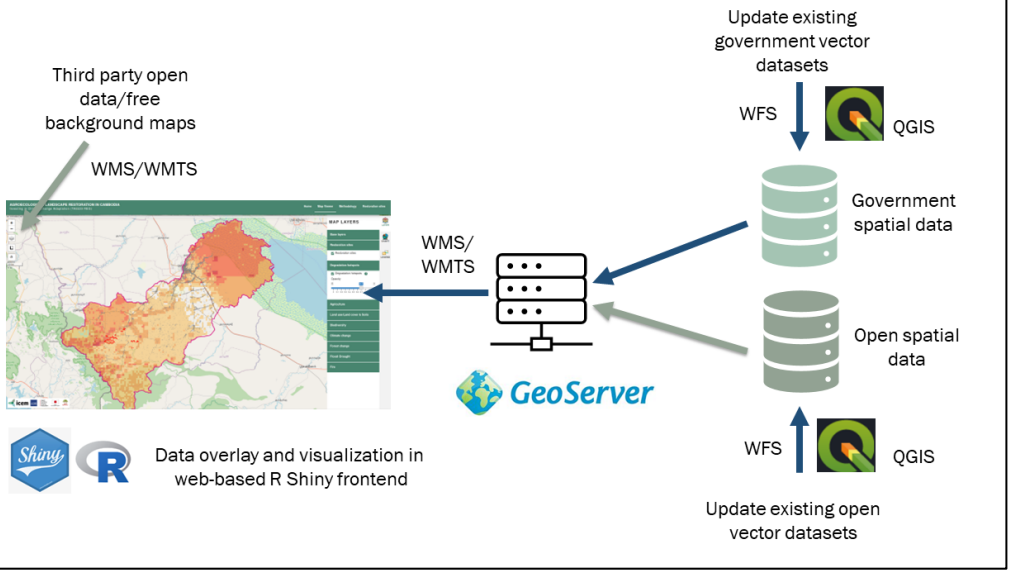
2. OVERVIEW OF ATLAS APPLICATION

- Software components, overall framework, and data flow
- Minimum hardware specifications
  - Memory
  - Disk space
  - GeoServer database
- Docker containers/software containerization
- Overview of GeoServer application

OVERVIEW OF ATLAS SOFTWARE COMPONENTS

- Open source software:
  - **R Shiny** <- R's web application framework
  - **RStudio Server** <- R coding environment (IDE)
  - **Shiny Server** <- enables sharing of development version of app online
  - **GeoServer** <- generate and share spatial data services
  - **Docker** <- facilitate installation/migration, and production mode set up for the app. Installation works cross-platform
- GeoServer (database backend, publishes spatial data services, e.g., WMS, WMTS, WFS)
  - International open standards for data sharing
- R Shiny front end (visualization of data/geospatial data)





DSS WEB HOSTING AND MANAGEMENT

Hardware	Docker + Geoserver + Database	Docker + RSTUDIO[1]	Docker + Shiny web app	Docker software	Other (swap, cache)	Total
CPU cores	2 (CPU MHz >= 2.0 GHz)					2
RAM memory (GB)	5	1	4[2]	4[3]	1	15
Hard drive (GB)	4	8	4	4	10	30

Recommended minimum specifications

[1] RStudio only required for development purposes

[2] estimated for 5 running replicas/instances (more replicas can be added to scale to more users if required by amending docker-compose.yml)

[3] Recommend minimum of 8 GB if running on Windows instead of Linux

Backups require additional space on another machine/external HDD

10

DSS WEB HOSTING AND MANAGEMENT

Set up two subdomains for server deployment

- one for the Shiny app: e.g., https://**sangker**.moe.gov.kh
- one for the GeoServer backend e.g., https://**geosangker**.moe.gov.kh

In the future, another subdomain might be needed for RStudio Server (e.g., **rstudio**.moe.gov.kh)

However, in this training we will show deployment to laptops, as this is important for app development. RStudio Server online deployment can be used to share future development versions with colleagues, before deploying to its final production mode for public access.

11

MINIMUM SPECIFICATIONS: MEMORY

- Memory usage of Geoserver (1 x Docker swarm container):
- $ docker stats --no-stream

CPU usage %

Memory usage

Memory usage %

```
rcooper@richard-XPS:/home/rstudio/webapps/agrophp$ docker stats --no-stream
CONTAINER ID   NAME                                                CPU %     MEM USAGE / LIMIT     MEM %
b60a436e20ed   geoagrophp_agrophpservice.1.qpilo0p7y3y5sxh6nm4hqqr2c   0.26%     2.988GiB / 31.01GiB   9.64%
a7a7e7c1132f   traefik traefik.1.eks4cmup8qs13b7gneogibu8j          0.00%     31.04MiB / 31.01GiB   0.10%
```

12

MINIMUM SPECIFICATIONS: DISK SPACE

- **To view disk space usage** of **Geoserver Docker images** and **containers**:
  $ docker ps --size <- *virtual* refers to total space used by Docker image and container (i.e., read-only and writable layers). The other value refers to container (i.e., writable layer) only.

```
rcooper@richard-XPS:/home/rstudio/webapps/agrophp$ docker ps --size
CONTAINER ID   IMAGE                              COMMAND                  CREATED             STATUS              PORTS
                                                          NAMES                                                    SIZE
b60a436e20ed   geoserver_ub2204_tomcat8.96:v2.22.4   "/bin/sh -c '/opt/ap…"   About an hour ago   Up About an hour
                                                          geoagrophp_agrophpservice.1.qpi1o0p7y3y5sxh6nm4hqqr2c   80.3kB (virtual 1.73GB)
a7a7e7c1132f   traefik:v2.10.6                    "/entrypoint.sh --lo…"   About an hour ago   Up About an hour   80/tcp
                                                          traefik_traefik.1.eks4cmup8qs13b7gneogibu8j              0B (virtual 153MB)
```

$ docker images

```
rcooper@richard-XPS:/home/rstudio/webapps/agrophp$ docker images
REPOSITORY                    TAG       IMAGE ID       CREATED        SIZE
geoserver_ub2204_tomcat8.96   v2.22.4   0215c7a3d17c   3 hours ago    1.73GB
traefik                       v2.10.6   7ec32645961a   3 days ago     153MB
```

13

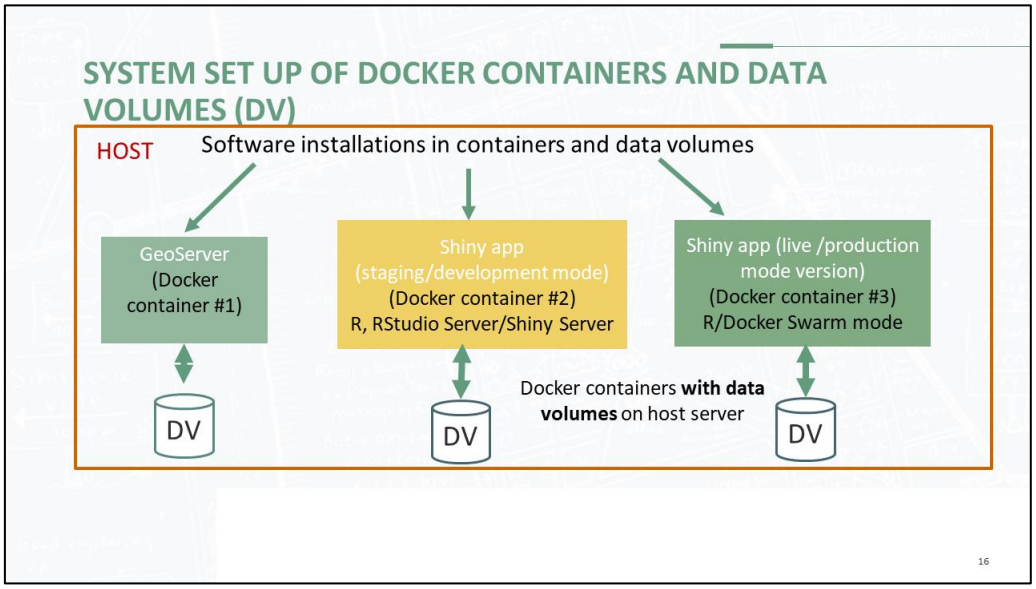
MINIMUM SPECIFICATIONS: GEOSERVER DATABASE SIZE

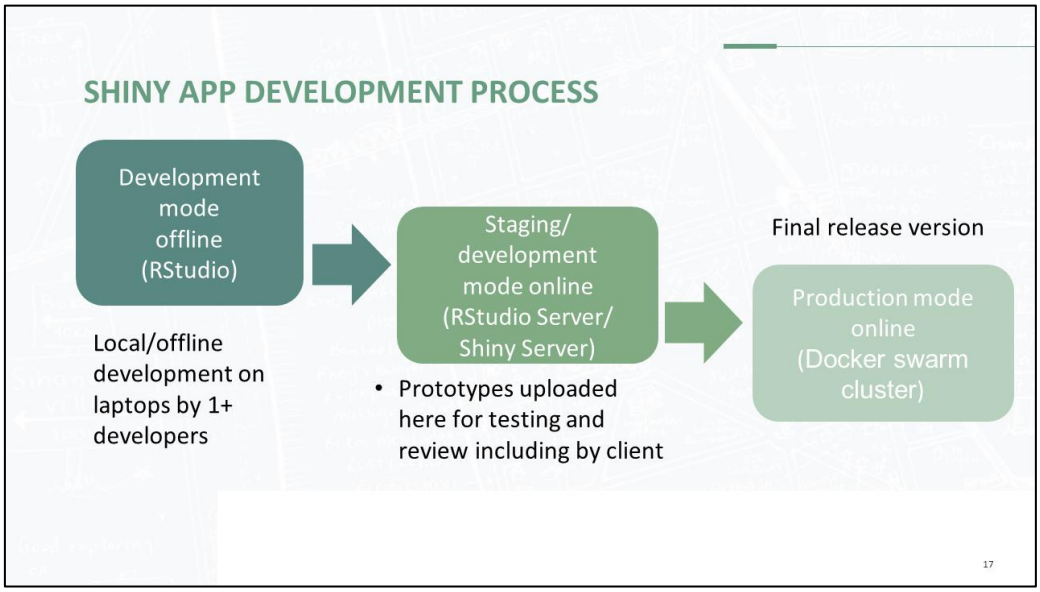
Path to GeoServer database (geoserver_data) (this may vary depending on your installation):

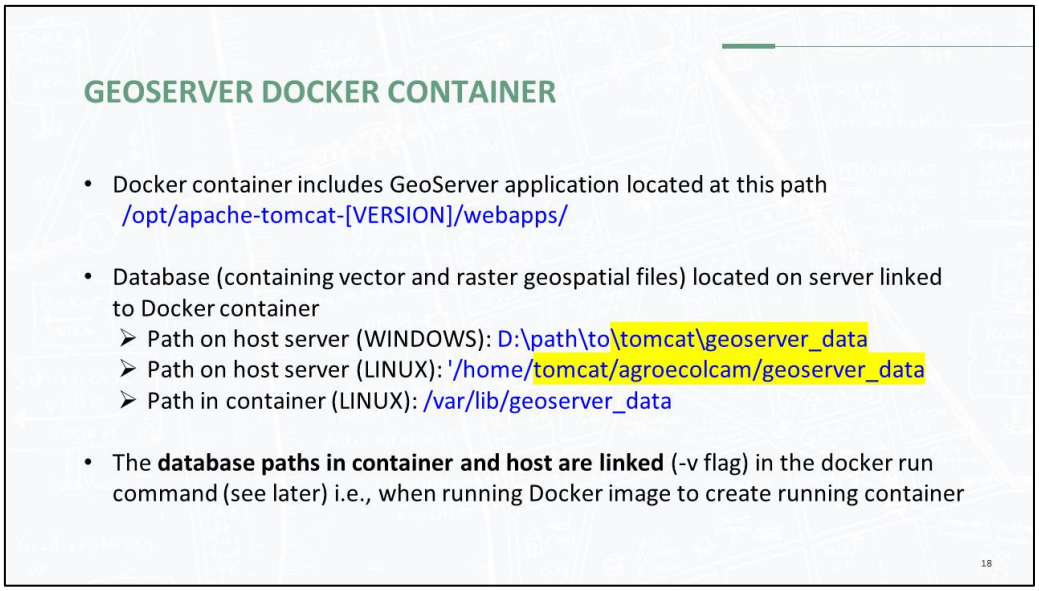
/home/tomcat/agroecolcam$ ls
geoserver_data
/home/tomcat/agroecolcam$ sudo du -sh geoserver_data/
2.6 GB

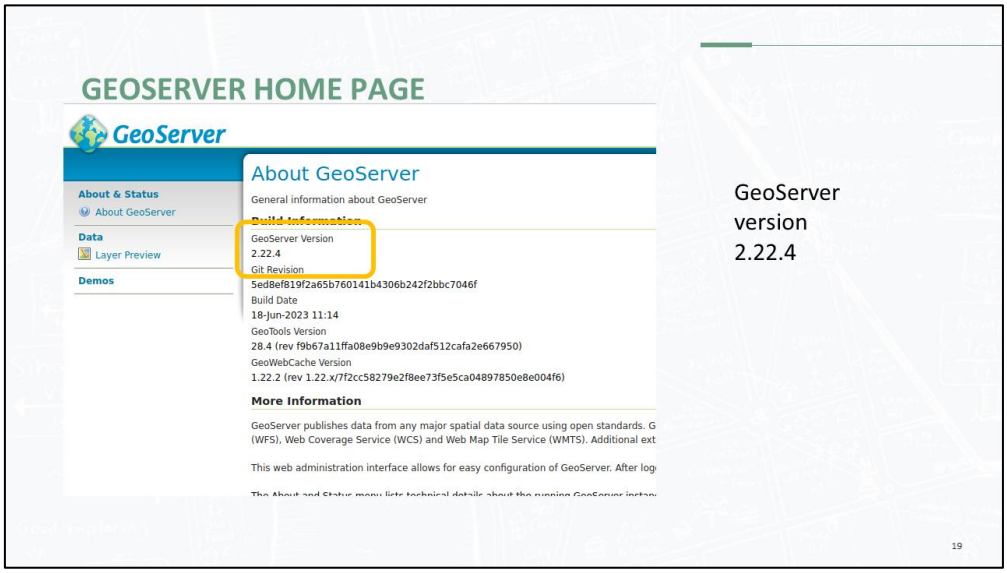
14

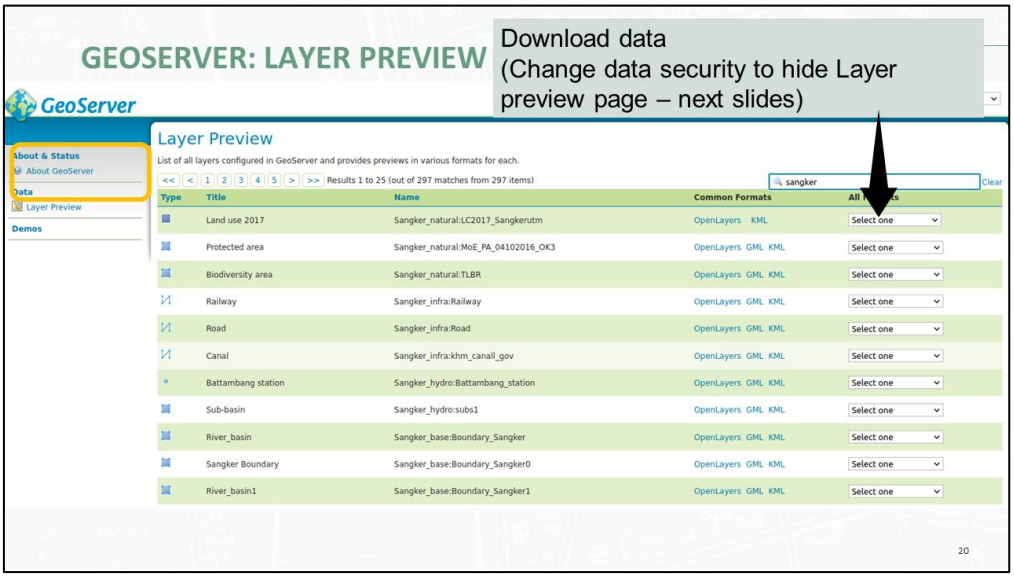
3. GEOSERVER DEPLOYMENT

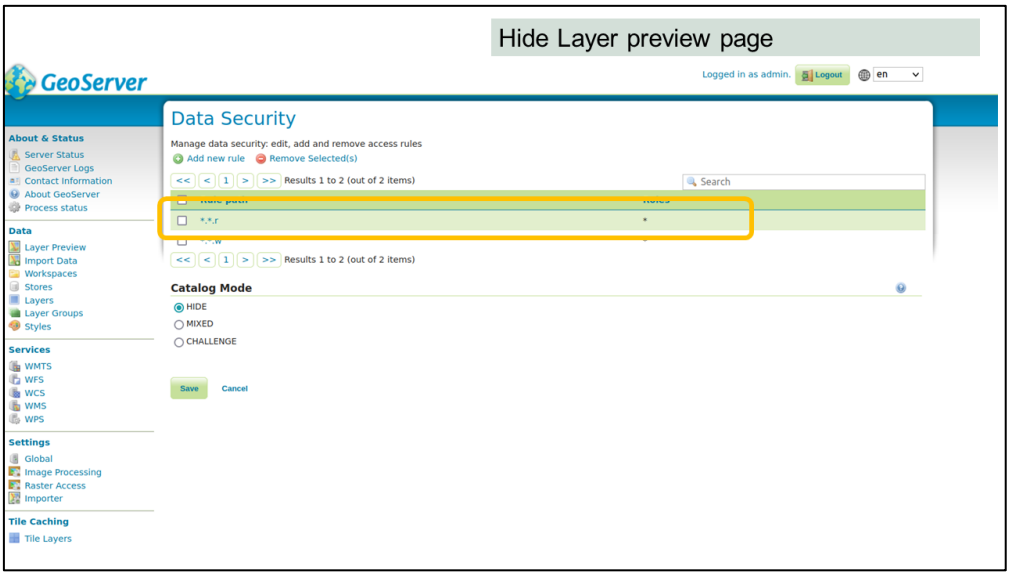

SYSTEM SET UP OF DOCKER CONTAINERS AND DATA VOLUMES (DV)
HOST
Software installations in containers and data volumes
GeoServer (Docker container #1)
Shiny app (staging/development mode) (Docker container #2) R, RStudio Server/Shiny Server
Shiny app (live /production mode version) (Docker container #3) R/Docker Swarm mode
DV
DV
DV
Docker containers with data volumes on host server
16

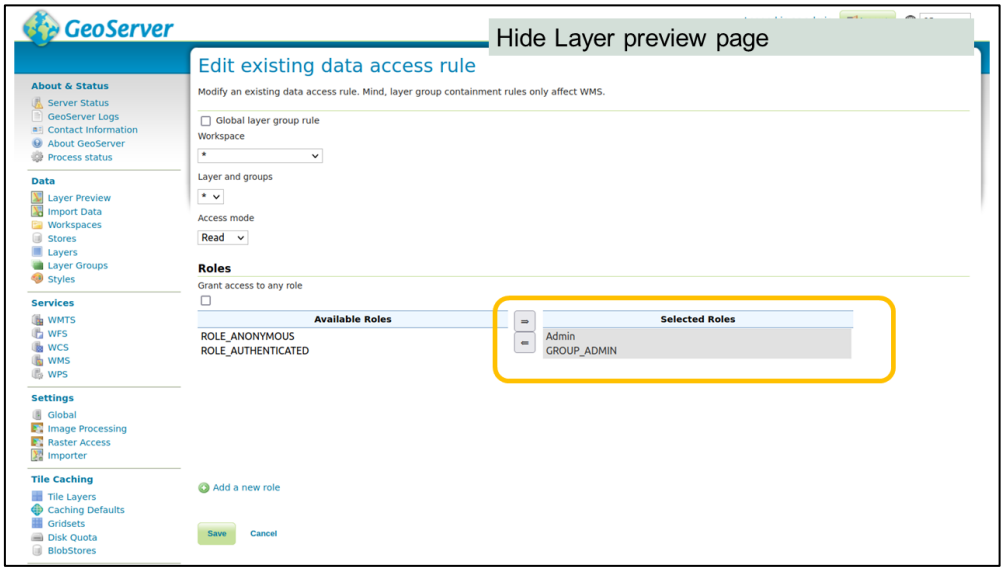

SHINY APP DEVELOPMENT PROCESS
Development mode offline (RStudio)
Staging/ development mode online (RStudio Server/ Shiny Server)
Final release version
Production mode online (Docker swarm cluster)
Local/offline development on laptops by 1+ developers
• Prototypes uploaded here for testing and review including by client
17

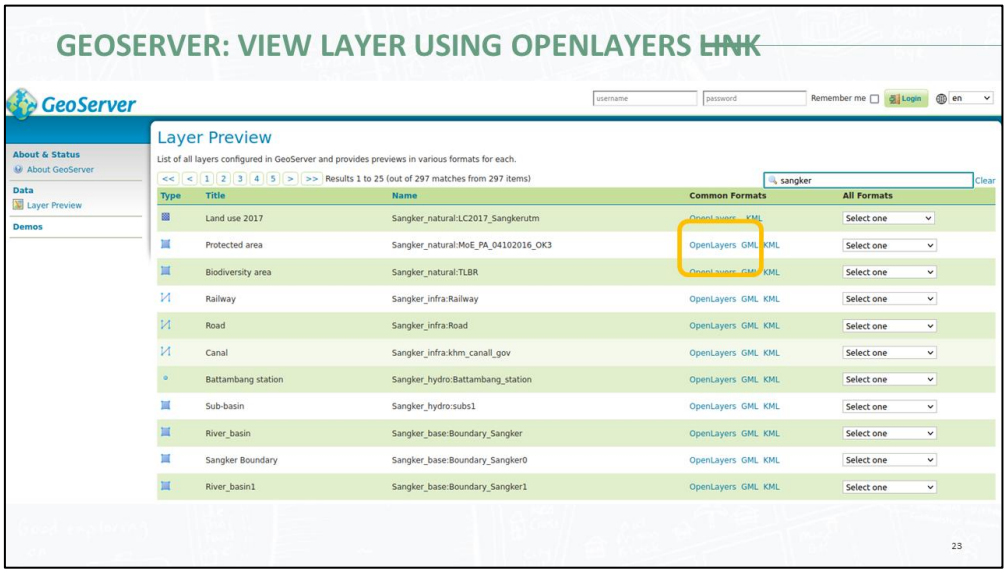

GEOSERVER DOCKER CONTAINER
• Docker container includes GeoServer application located at this path /opt/apache-tomcat-[VERSION]/webapps/
• Database (containing vector and raster geospatial files) located on server linked to Docker container
➢ Path on host server (WINDOWS): D:\path\to\tomcat\geoserver_data
➢ Path on host server (LINUX): '/home/tomcat/agroecolcam/geoserver_data
➢ Path in container (LINUX): /var/lib/geoserver_data
• The database paths in container and host are linked (-v flag) in the docker run command (see later) i.e., when running Docker image to create running container
18


GEOSERVER HOME PAGE
GeoServer
About & Status
About GeoServer
Data
Layer Preview
Demos
About GeoServer
General information about GeoServer
GeoServer Version
2.22.4
Git Revision
5ed8ef819f2a65b760141b4306b242f2bbc7046f
Build Date
18-Jun-2023 11:14
GeoTools Version
28.4 (rev f9b67a11ffa08e9b9e9302daf512cafa2e667950)
GeoWebCache Version
1.22.2 (rev 1.22.x/7f2cc58279e2f8ee73f5e5ca04897850e8e004f6)
More Information
GeoServer publishes data from any major spatial data source using open standards.
(WFS), Web Coverage Service (WCS) and Web Map Tile Service (WMTS). Additional ext
This web administration interface allows for easy configuration of GeoServer. After log
GeoServer version 2.22.4
19


GEOSERVER: LAYER PREVIEW
Download data
(Change data security to hide Layer preview page – next slides)
GeoServer
Layer Preview
List of all layers configured in GeoServer and provides previews in various formats for each.
Results 1 to 25 (out of 297 matches from 297 items)
sangker
Clear
Type
Title
Name
Common Formats
Land use 2017
Sangker_natural:LC2017_Sangkerutm
Protected area
Sangker_natural:MoE_PA_04102016_OK3
Biodiversity area
Sangker_natural:TLBR
Railway
Sangker_infra:Railway
Road
Sangker_infra:Road
Canal
Sangker_infra:khm_canall_gov
Battambang station
Sangker_hydro:Battambang_station
Sub-basin
Sangker_hydro:subs1
River_basin
Sangker_base:Boundary_Sangker
Sangker Boundary
Sangker_base:Boundary_Sangker0
River_basin1
Sangker_base:Boundary_Sangker1
OpenLayers GML KML
Select one
20


Hide Layer preview page
GeoServer
Logged in as admin.
Logout
en
Data Security
Manage data security: edit, add and remove access rules
Add new rule
Remove Selected(s)
Results 1 to 2 (out of 2 items)
Search
Catalog Mode
HIDE
MIXED
CHALLENGE
Save
Cancel
About & Status
Server Status
GeoServer Logs
Contact Information
About GeoServer
Process status
Data
Layer Preview
Import Data
Workspaces
Stores
Layers
Layer Groups
Styles
Services
WMTS
WFS
WCS
WMS
WPS
Settings
Global
Image Processing
Raster Access
Importer
Tile Caching
Tile Layers


GeoServer
Hide Layer preview page
Edit existing data access rule
Modify an existing data access rule. Mind, layer group containment rules only affect WMS.
About & Status
Server Status
GeoServer Logs
Contact Information
About GeoServer
Process status
Data
Layer Preview
Import Data
Workspaces
Stores
Layers
Layer Groups
Styles
Services
WMTS
WFS
WCS
WMS
WPS
Settings
Global
Image Processing
Raster Access
Importer
Tile Caching
Tile Layers
Caching Defaults
Gridsets
Disk Quota
BlobStores
Global layer group rule
Workspace
Layer and groups
Access mode
Read
Roles
Grant access to any role
Available Roles
ROLE_ANONYMOUS
ROLE_AUTHENTICATED
Selected Roles
Admin
GROUP_ADMIN
Add a new role
Save
Cancel


GEOSERVER: VIEW LAYER USING OPENLAYERS ~~LINK~~
GeoServer
Layer Preview
List of all layers configured in GeoServer and provides previews in various formats for each.
Results 1 to 25 (out of 297 matches from 297 items)
sangker
Type
Title
Name
Common Formats
All Formats
Land use 2017
Sangker_natural:LC2017_Sangkerutm
Protected area
Sangker_natural:MoE_PA_04102016_OK3
Biodiversity area
Sangker_natural:TLBR
Railway
Sangker_infra:Railway
Road
Sangker_infra:Road
Canal
Sangker_infra:khm_canall_gov
Battambang station
Sangker_hydro:Battambang_station
Sub-basin
Sangker_hydro:subs1
River_basin
Sangker_base:Boundary_Sangker
Sangker Boundary
Sangker_base:Boundary_Sangker0
River_basin1
Sangker_base:Boundary_Sangker1
OpenLayers GML KML
Select one
23


GEOSERVER
WMS version: 1.1.1
Tiling: Single tile
Antialias: Full
Format: PNG 24bit
Styles: Default
Width/Height: Auto
Scale = 1 : 4M
MoE_PA_04102016_OK3
fid
OBJECTID
NAME
CODE
PA_Type
PA_Code
Decree_ha
Name_E
Print_Code
Co
MoE_PA_04102016_OK3.8
9
Kulen Promtep
9
Wildlife Sanctuary
16
402500
Kulen Promtep
17
6
24

3. GEOSERVER DEPLOYMENT

- Two components to migrate:
  - **Database:** Geospatial database linked to GeoServer
  - **Software:** Docker/GeoServer software image
- Download GeoServer data and Docker image (see above link)
- GeoServer configuration
  - A. Add tomcat user, and match UID and GID (LINUX ONLY)
  - B. Extract GeoServer data (*tar.gz) to host machine
  - C. Load and run Docker image
  - D. Access GeoServer

25

BACKGROUND NOTES: MOVING A DOCKER IMAGE: SAVING A DOCKER IMAGE AS A TAR.GZ FILE

Participants don't need to do this, but details included here for reference on how to save a Docker image as a tar.gz archive for migrating to another machine

```
$ docker save -o IMAGE_NAME.tar IMAGE_NAME      (save image as *tar file)
$ gzip < IMAGE_NAME.tar > IMAGE_NAME.tar.gz     (compress image tar as *tar.gz)
```

Example:

```
$ docker save -o img_agroecolcam_geoserver_5dec2023.tar geoserver_ub2204_tomcat8596:v2.22.4
$ gzip img_agroecolcam_geoserver_05dec2023.tar > img_agroecolcam_geoserver_05dec2023.tar.gz
```

Use the 'scp' command line utility to move the (large) tar.gz file to another server

NOTE: The Docker tar.gz file can be downloaded from the above link

26

BACKGROUND NOTES: MOVING A DOCKER CONTAINER: CREATING A DOCKER IMAGE FROM A RUNNING CONTAINER AND SAVING AS A TAR.GZ FILE

Participants don't need to do this, but details included here for reference on how to convert a container to an image and tar.gz archive for migrating to another machine

- Use three commands sequentially: docker commit, docker save and gzip

```
$ docker commit -p CONTAINER_NAME IMAGE_NAME  (create image from a container ('-p' is is pause a running container)
$ docker save -o IMAGE_NAME.tar IMAGE_NAME    (save image as *tar file)
$ gzip < IMAGE_NAME.tar > IMAGE_NAME.tar.gz   (compress image tar as *tar.gz)
```

} See previous slide

(A) ADD TOMCAT USER, AND MATCH UID AND GID (LINUX ONLY)

1. If tomcat user and group do not exist on host:
On host, add tomcat user and modify user and group identifiers to match UID/GIDs in Docker container:

```
$ group useradd -g 1002 tomcat <- add tomcat user with UID of 1002
$ sudo groupmod -g 1002 tomcat <- modify GID of tomcat group to 1002 (optional, check GID)
$ sudo groupadd -g 5020 shiny-app <- add shiny-app group (group name used for files uploaded
to ../geoserver_data/raster_vectors folder)
$ id user_name <- check UID of user

Check UID
$ awk -F: '{printf "%s:%s\n",$1,$3}' /etc/passwd   # use to list UIDs on HOST/CONTAINER
    $ cat /etc/passwd <-  show all users
```

28

(A) ADD TOMCAT USER, AND MATCH UID AND GID (LINUX ONLY)

Or, if tomcat user and group already exist on host,
2A: Access running container (**see 'load' and 'run' commands in later slides**) from host and **change UIDs/GID, and remember to append 'bash' (without quotes) to the 'docker exec' command:**

```
$ docker exec -u root -it [CONTAINER NAME] bash
```

Modify the tomcat UID and GID IN THE CONTAINER to match those on the host. E.g., to modify UID and GID of tomcat user and group to 5001:

```
# groupmod -g 5001 tomcat
# usermod -u 5001 tomcat
```

Change file ownership

```
# find / -uid 1002 -exec chown -v -h 5001 '{}' \;
# find / -gid 1002 -exec chgrp -v -h 5001 '{}' \;
```

The modified container can then be converted to a new image using 'docker commit':

```
$ docker commit -p [CONTAINER NAME] [NEW IMAGE NAME]
```

See later slides on how to run the newly created image with changed UIDs/GID to create a new container.

29

(A) ADD TOMCAT USER, AND MATCH UID AND GID (LINUX ONLY)

Or, if tomcat user and group already exist on host,
2B: Rebuild Docker image using dockerfile (see guidelines in dockerfile)

30

(B) EXTRACT GEOSERVER DATA (*TAR.GZ) TO GEOSERVER_DATA DIRECTORY

(i) On host (Linux), create agroecolcam directory:
mkdir /home/tomcat/agroecolcam

(i) On host (Windows) create path, for example, as follows:
D:\path\to\tomcat\agroecolcam

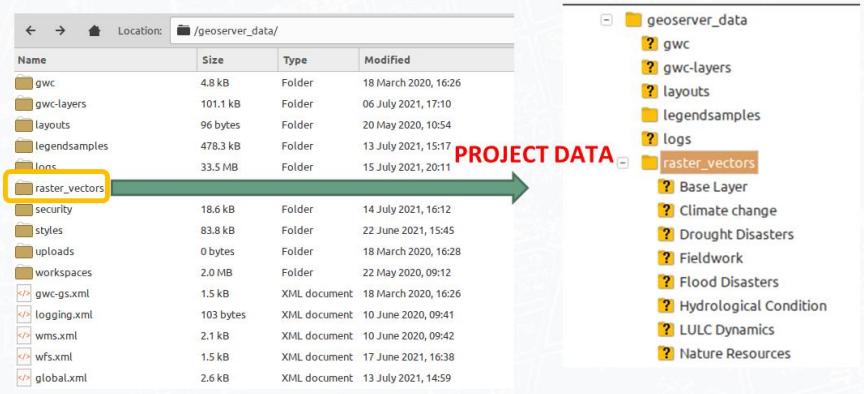
(ii) Download and extract data from agroecolcam_geoserver_data_05dec2023.tar.gz* into agroecolcam directory

Extract archive:
/home/tomcat/agroecolcam $ sudo tar -xzvf *tar.gz

(ii) On host Windows:
Use 7-zip or alternative to extract:
https://www.7-zip.org/download.html

NOTE: command used to create data archive: **sudo tar -czvf agroecolcam_geoserver_data_05dec2023.tar.gz --exclude "*tar.gz*".**
Move file to another server: **$ scp agroecolcam_geoserver_data_05dec2023.tar.gz user@xx.xx.xx.xxx:/home/user**

(B) ON HOST EXTRACT GEOSERVER DATA (*TAR.GZ) TO GEOSERVER_DATA DIRECTORY



32

(B) ON HOST EXTRACT GEOSERVER DATA (*TAR.GZ) TO GEOSERVER_DATA DIRECTORY (LINUX ONLY)

(iii) On a **Linux** host machine, modify ownership and permissions of extracted geoserver_data:
cd /home/tomcat/agroecolcam
sudo chown -R tomcat:tomcat .
sudo chmod -R ug+rwx .
sudo chmod -R g+s .

(iv) Data linked to GeoServer are stored in /home/tomcat/agroecolcam/geoserver_data/raster_vectors
cd /home/tomcat/agroecolcam/geoserver_data/raster_vectors
sudo chown -R tomcat:shiny-app .
sudo chmod -R ug+rwx .
sudo chmod -R g+s .

NOTE: users who upload data to the raster_vectors directory need to be added to shiny-app group on host machine
e.g., $ sudo groupadd -g username shiny-app

(vi) Load and run Docker container (see section below).

33

(C) LOAD AND RUN DOCKER IMAGE

- For **laptop installation** (http) download this file to download folder:
  img_agroecolcam_geoserver_05dec2023.tar.gz
- For **server installation** (https)
  Run the above image and modify server.xml (see later slide for details)
  NOTE: an https version of the file has been prepared (see downloads)

34

(C) LOAD AND RUN DOCKER IMAGE (LINUX)

- Change to directory containing downloaded Docker tar.gz file and load Docker image:
  $ docker load < IMAGE_NAME.tar.gz

35

(C) LOAD AND RUN DOCKER IMAGE (WINDOWS)

Change to directory containing downloaded Docker image and load Docker image:

> docker load -i "D:\path\to\imagedownload\ IMAGE_NAME.tar.gz"

(C) LOAD AND RUN DOCKER IMAGE

View Docker image is loaded
$ docker images

Example:

```
rcooper@ubuntu-cloud-2:~/dockerimages$ docker images
REPOSITORY                    TAG              IMAGE ID       CREATED         SIZE
geoserver_ub2204_tomcat8596   v2.22.4          ca9fb99f29fa   4 hours ago     1.73GB
```

37

(C) LOAD AND RUN DOCKER IMAGE

Can run GeoServer Docker image in two ways:

C1. Using the docker run command to create a single Docker container (and test container is working properly)

C2. Using docker compose yml file in Docker swarm mode (as used in production mode on server). This approach can be scaled up in future.

38

(C1) LOAD AND RUN DOCKER IMAGE (LINUX)

1. Using docker run command:

host path (Linux): container path (Ubuntu/Linux)

- Run Docker image:
  $ docker run -d -it
  -v '/home/tomcat/agroecolcam/geoserver_data:/var/lib/geoserver_data'
  -p 8030:8080 --name geoserver_agroecolcam IMAGE_NAME:TAG

Check name of image to be loaded is correct using 'docker images' command

NOTES: (i) in above command, 8030 is the host port and 8080 the container port,
(ii) the container created on running the command is 'geoserver_agroecolcam'
(iii) -v links the paths of the host and container (host path: container path)
(iv) Example of IMAGE_NAME:TAG is geoserver_ub2204_tomcat8596:v2.22.4

```
rcooper@ubuntu-cloud-2:~/dockerimages$ docker images
REPOSITORY                    TAG              IMAGE ID       CREATED         SIZE
geoserver_ub2204_tomcat8596   v2.22.4          ca9fb99f29fa   4 hours ago     1.73GB
```

(C1) LOAD AND RUN DOCKER IMAGE (WINDOWS)

1. Check GeoServer works in running Docker container:

- Run Docker image:

host path (Windows): container path (Ubuntu/Linux)

```
docker run -d -it
-v "D:\path\to\tomcat\agroecolcam\geoserver_data:/var/lib/geoserver_data"
--name geoserver_agroecolcam
-p 8030:8080 IMAGE_NAME:TAG
```

Check name of image to be loaded is correct using 'docker images' command

NOTES: (i) in above command, 8030 is the host port and 8080 the container port,
(ii) the container created on running the command is 'geoserver_agroecolcam'
(iii) -v links the paths of the host and container (host path: container path)
(iv) Example of IMAGE_NAME:TAG is geoserver_ub2204_tomcat8596:v2.22.4

40

(C1) LOAD AND RUN DOCKER IMAGE

- Check for running Docker container
  $ docker ps

Example:

```
rcooper@richard-XPS:~/Downloads/AGROECOLOGY_FINAL/CAMBODIA/GEOSERVER$ docker ps
CONTAINER ID   IMAGE                                      COMMAND                  CREATED              STATUS              PORTS
                                                                  NAMES
9de5f03f2399   geoserver_ub2204_tomcat8596:v2.22.4   "/bin/sh -c '/opt/ap…"   6 seconds ago        Up 5 seconds        0.0.0.0:8030->8080/tcp,
:::8030->8080/tcp                                                 geoserver_agroecolcam
```

GeoServer will start automatically on running the 'docker run' command above

- In browser go to http://127.0.0.1:8030/geoserver/web/
- Stop container
  $ docker stop CONTAINER_NAME

41

(C2) RUN DOCKER IMAGE (PRODUCTION MODE)

Create path with directories to Shiny app and copy docker compose file to agroecol directory

- Set up Docker swarm (ON HOST MACHINE):
  - On host (Windows) go to for example, D:\path\to\shiny\webapps\agrocam

```
$ cd /home/shiny/webapps/agrocam
$ docker swarm init
$ docker info
```

- Create Docker overlay (to connect all containers):
  $ docker network create -d overlay --attachable net2

Check config matches in docker-compose files

- Check all containers access network:
  $ docker network inspect net2
- Traefik is used as reverse proxy and load balancer for docker swarm set up:
  https://github.com/traefik/traefik
  - Download Traefik Docker image on host machine:
    $ docker pull traefik:v2.10.6
  - Alternative to Traefik is Nginx but potentially more complicated to configure

NOTE: Traefik only needed n server set up

42

(C2) RUN DOCKER IMAGE (PRODUCTION MODE)

- copy docker-compose-geoserver yml file to Shiny app directory (/home/shiny/webapps/agrocam)
- copy docker-compose-traefik yml file to Shiny app directory (/home/shiny/webapps/agrocam)
- change to directory containing above docker-compose yml files

43

docker-compose-geoserver-laptop.yml:

```
version: "3.7"
 services:
   agrocamservice:
     image: geoserver_ub2204_tomcat8.96:v2.22.4   <- CHECK THIS
     deploy:
       replicas: 1   <- Only 1 needed
       labels:
         - "traefik.enable=true"
         - "traefik.http.routers. agrocamservice.entrypoints=web"
         - "traefik.docker.network=net2"
         - "traefik.http.routers.agrocamservice.rule= Host(`127.0.0.1`) && (PathPrefix(`/agroecolcam/`))"
         - "traefik.http.middlewares.agrocamservice.stripprefixregex.regex=/agroecolcam/[a-z0-9]+/[0-9]+/"
```

44

docker-compose-geoserver-laptop.yml:

```
...
         - "traefik.http.routers.agrocamservice.middlewares=agrocamservice@docker"
         - "traefik.http.services.agrocamservice.loadbalancer.server.port=8080"
         - "traefik.http.services.agrocamservice.loadbalancer.sticky=true"
         - "traefik.http.services.agrocamservice.loadbalancer.sticky.cookie.name=stickycookie"
       restart_policy:
         condition: on-failure
       update_config:
         delay: 2s
     volumes:
       - /home/tomcat/agroecolcam/geoserver_data:/var/lib/geoserver_data      <- adjust as required
     networks:
       - net2
 networks:
   test_net2:
     driver: overlay
     external: true
```

45

docker-compose-traefik-laptop.yml

```
version: "3.7"
services:
  traefik:
    image: traefik:v2.10.6
    deploy:
      restart_policy:
        condition: any
      placement:
        constraints:
          - node.role == manager
      labels:
       - "traefik.http.services.traefik.loadbalancer.server.port=80"
       #- "traefik.http.services.traefik.loadbalancer.server.port=443"
```

46

docker-compose-traefik-laptop.yml

```
command:
    - "--log.level=DEBUG"
    - "--api.insecure=true"
    - "--providers.docker=true"
    - "--providers.docker.swarmMode=true"
    - "--providers.docker.exposedbydefault=true"
    - "--entrypoints.web.address=:80"
  ports:
    #- "8098:8098"
    - "9080:80"
    - "9020:8080"
    #- "443:443"
  volumes:
    - "/var/run/docker.sock:/var/run/docker.sock:ro"
  networks:
    - net2
networks:
  net2:
  driver: overlay
  external: true
```

47

To deploy/start GeoServer app

$ cd /home/shiny/webapps/agrocam

- On host (Windows) go to for example, D:\path\to\shiny\webapps\agrocam

$ docker stack deploy -c docker-compose-traefik-laptop.yml agrocam
$ docker stack deploy -c docker-compose-geoserver-laptop.yml geo

Access Shiny app in browser **on laptop**:
http://127.0.0.1:9080

Visit Traefik app at port 9020 (e.g. on laptop: 127.0.0.1:9020)
Check Traefik logs: $ docker service logs agrocam_traefik
Check Shiny app logs: $ docker service logs geo_agrocamservice

48

(C2) RUN DOCKER IMAGE (PRODUCTION MODE)

- Deploy docker-compose-geoserver.yml file
  $ docker stack deploy -c docker-compose-geoserver-laptop.yml geoagrocam

  $ docker stack deploy -c docker-compose-traefik-laptop.yml geoagrocam

  NOTE: Traefik used here to map ports. Used for 'https' access on server set up

- Show Docker service is running (shows no. of containers/replicas): $ docker service ls

```
rcooper@richard-XPS:~/Downloads/AGROECOLOGY_FINAL/CAMBODIA/GEOSERVER$ docker service ls
ID             NAME                MODE         REPLICAS   IMAGE                                PORTS
q8qniq8kkvbs   geo_agrocamservice  replicated   1/1        geoserver_ub2204_tomcat8.96:v2.22.4
i98v7to10rjy   traefik_traefik     replicated   1/1        traefik:v2.10.6                      *:9020->8080/tcp, *:9080->80/tcp
```

- Show GeoServer container is running (shows individual containers): $ docker container ls

```
rcooper@richard-XPS:~/Downloads/AGROECOLOGY_FINAL/CAMBODIA/GEOSERVER$ docker container ls
CONTAINER ID   IMAGE                                COMMAND                  CREATED             STATUS             PORTS
                                                                             NAMES
1d0cd7de3e7a   geoserver_ub2204_tomcat8.96:v2.22.4   "/bin/sh -c '/opt/ap…"   45 seconds ago      Up 44 seconds
                                                                             geo_agrocamservice.1.v2iq7t7bpoosp3zg3uaxtss3h
f95368f8c8b6   traefik:v2.10.6                      "/entrypoint.sh --lo…"   About an hour ago   Up About an hour   80/tcp
                                                                             traefik_traefik.1.ppl54mgkfrbvzc075s9to2ky8
```

49

4A. MAINTENANCE

MAINTENANCE

Problem: The GeoServer application becomes slow to respond.

Response:

(i) Check the amount of RAM and CPU resources being used by the Docker containers (and contained apps):

$ docker stats --no-stream

(ii) This may also occur if more users are accessing the application. Modify the number of replicas (also called containers) in **docker-compose-geoserver.yml** and consider adding more RAM/CPU resources to the server as more replicas will use more resources.

For example, to deploy 2 replicas:

```
deploy:
  replicas: 2
```

Then redeploy the GeoServer service (docker service rm … / docker stack deploy …)

51

MAINTENANCE

Problem: If GeoServer becomes inaccessible or unresponsive (maps not showing in map viewer)

Response:

(i) Redeploy the Docker GeoServer service

Obtain service name from $docker service ls

- Check Docker service is running: $ docker service ls
- Stop both Traefik and GeoServer services: $ docker service rm [NAME OF SERVICE]
- Redeploy Traefik and GeoServer services:
  Change to Shiny app directory containing docker-compose yml files:
  $ docker stack deploy –c docker-compose.traefik.yml agroecolcam
  $ docker stack deploy –c docker-compose.geoserver.yml geo
- Check service is running: $ docker service ls
- Check containers are running: $ docker container ls

52

MAINTENANCE

If problem persists:

- Check service logs:
  $ docker service logs [SERVICE NAME]

- Restart the Docker service (**note: this will stop all containers**)
  Windows server: open PowerShell as admin: restart-service *docker*
  Linux: $ sudo service docker restart or sudo systemctl restart docker
  Remember to restart other previously running containers as Docker service restart will stop all running containers (e.g., add to Linux cronjob / Windows restart for automatic restarts)

NOTE: If no changes have been made to the Docker images and docker-compose yml files, then check for changes to general server configuration.

53

UPGRADING GEOSERVER

Two approaches:

1. Modify Docker file and build new Docker image
   See contents of Docker file (shared on download link) for more details
   This is the preferred approach as the Docker image can then be readily duplicated programmatically, rather than manually upgrading the Docker container and creating a new Docker image using the docker commit command (see 2 below).

2. Run docker container ('docker run …'), access the container using 'docker exec …', make the required modifications, then create a new Docker image from the container ('docker commit ..'). This approach may not always work depending on the changes required.

After 1 or 2 above, address these tasks:
(i) Modify image link in the docker-compose-geoserver.yml
(ii) Finally redeploy the Docker GeoServer service as detailed earlier ('docker service rm … /docker stack deploy … ')

54

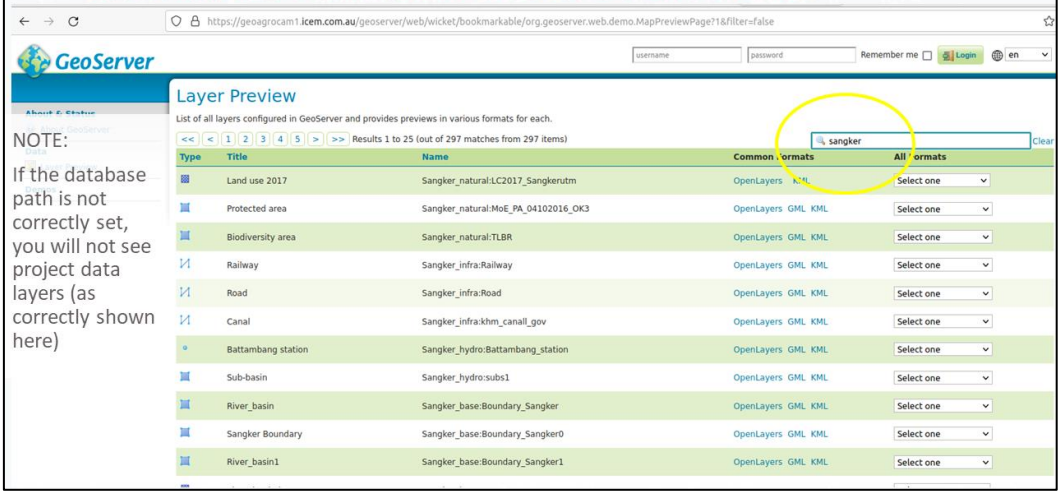
POTENTIAL FUTURE ISSUES

Modifying web.xml and server.xml files (see following slides):

- The following process is recommended for modifying configuration files within the GeoServer container:
  (i) Create a new Docker container from the GeoServer Docker image ('docker run …')
  (ii) Access the container and edit the required file(s) ('docker exec …')
  (iii) Create a new Docker image from the modified container ('docker commit…)
  (iv) Update the name of the Docker image in the docker-compose-geoserver.yml file
  (v) Re-run the Docker service:
  i. cd to folder containing docker-compose-geoserver.yml file and
  ii. run 'docker service rm [geoserver_service_name] / docker stack deploy -c docker-compose-geoserver.yml geo')
  iii. Check Geoserver Docker service is running (docker service ls)

55

DATA LAYERS NOT VISIBLE UNDER LAYER PREVIEW



NOTE:

If the database path is not correctly set, you will not see project data layers (as correctly shown here)

DATA LAYERS NOT VISIBLE UNDER LAYER PREVIEW

(i) Edit web.xml

If the database at /home/tomcat/agroecolcam/geoserver_data is not accessible after running the Docker container (see next slide), ensure that the **web.xml** at /opt/apache-tomcat-[version]/webapps/geoserver/WEB-INF/ contains the following text at the end of the file:

```
<context-param>
<param-name>GEOSERVER_DATA_DIR</param-name>
<param-value>/var/lib/geoserver_data</param-value>
</context-param>
</web-app>
```

57

DATA LAYERS NOT VISIBLE UNDER LAYER PREVIEW

(i) continued:
To edit web.xml
Access running container:
$ docker exec -u root -it [CONTAINER NAME] bash

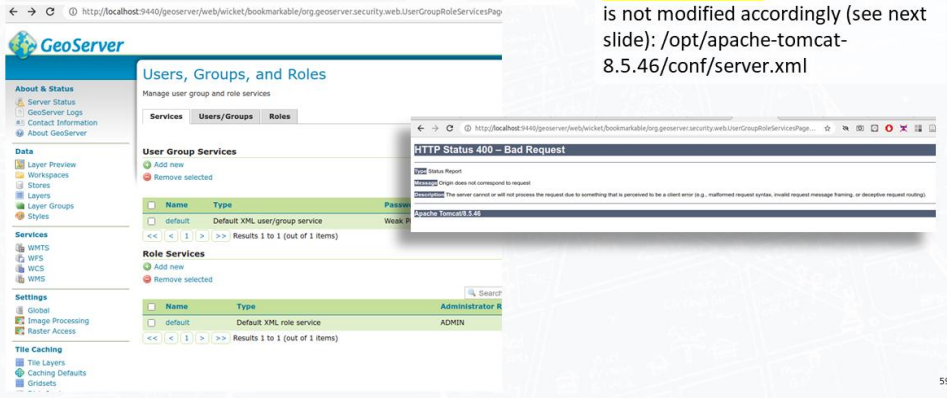
Edit web.xml with nano editor:
nano web.xml

Nano editor commands:
To save editing: CTRL+o
Then press return
Then exit nano editor with CTRL+x

58

USERS/GROUPS GIVE STATUS 400

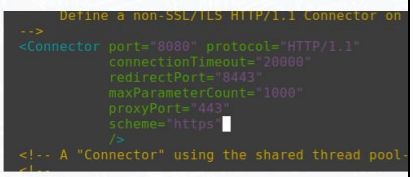
(ii) Edit server.xml for http/https access
Log into GeoServer using 'admin' account and see if the following error occurs



An error may show when clicking on Users/Groups tab if the following file is not modified accordingly (see next slide): /opt/apache-tomcat-8.5.46/conf/server.xml

59

USERS/GROUPS GIVE STATUS 400



(ii continued) To enable https in tomcat Geoserver
In file /opt/apache-tomcat-[version]/conf/server.xml
add these two lines:
proxyPort="443"
scheme="https"
After editing, create a new image from the modified Docker container ($ docker commit -p ….) and edit the image name in the docker-compose-geoserver yml file

NOTE: remove these two lines when running as http (e.g., as localhost on laptop) or an error on trying to view images and Users/Groups will occur. Two GeoServer images are provided for http and https.

Example:
$ docker commit -p geo_agrocamservice.1.v2iq7t7bpoosp3zg3uaxtss3h geoserver_ub2204_tomcat8596:v2.22.4_https
$ docker save -o img_agroecolcam_geoserver_https_06nov2023.tar geoserver_ub2204_tomcat8596:v2.22.4_https
$ gzip < img_agroecolcam_geoserver_https_06nov2023.tar > img_agroecolcam_geoserver_https_06nov2023.tar.gz
Load https image into server:
$ docker load < img_agroecolcam_geoserver_https_06nov2023.tar.gz

60

CHECKING LINUX OS INSTALLED IN CONTAINER

```
$docker exec -it -u root geo_agrocamservice.1.v2iq7t7bpoosp3zg3uaxtss3h bash
root@1d0cd7de3e7a:/home/tomcat# cat /etc/os-release
    PRETTY_NAME="Ubuntu 22.04.3 LTS"
    NAME="Ubuntu"
    VERSION_ID="22.04"
    VERSION="22.04.3 LTS (Jammy Jellyfish)"
    VERSION_CODENAME=jammy
    ID=ubuntu
    ID_LIKE=debian
    HOME_URL="https://www.ubuntu.com/"
    SUPPORT_URL="https://help.ubuntu.com/"
    BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
    PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
    UBUNTU_CODENAME=jammy
    root@1d0cd7de3e7a:/home/tomcat#
```

Upgraded to ubuntu 22.04.3 LTS

61

4B. DATA UPDATING

- Overview of GeoServer
- Overview of GeoServer data updating
- Practical exercise: GeoServer data updating

OVERVIEW OF GEOSERVER

Geoserver is an open source server that allows users to share, process and edit geospatial data. It is built on GeoTools, an open source Java GIS toolkit.

Geoserver is a type of software that provides services to web applications. Making it easier for web developers to implement communication of input/output of large geospatial database.

63

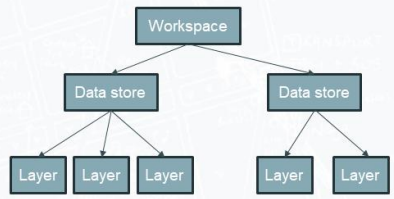
OVERVIEW OF GEOSERVER – DATA ORGANIZATION

Workspaces
- Enables grouping of similar datasets together

Data Store
- Connects/links to a spatial data source such as file (raster, vector), database or server.

Layer
- A set of geographic features derived from a spatial data source (e.g., may correspond to a GIS file).

Layer Group
- A container that allows organizing layers and other layer groups in a hierarchical structure, and can be referred to by a single name in WMS requests



Layer Groups Edit page

64

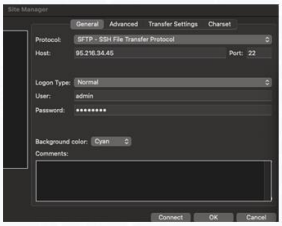
DATA STORAGE – FILE MANAGER

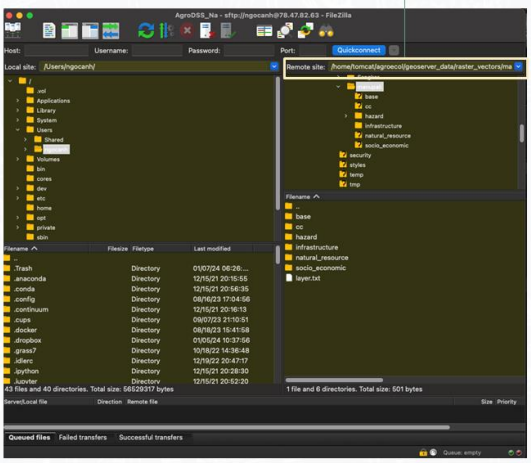
/home/tomcat/agroecol/geoserver_data/raster_vectors/

Assess geoserver data directory via FileZilla
- Download at: https://filezilla-project.org

Connect to the remote site with the following url and credentials: GeoServer login: *admin*: *eeR!oh7d*

NOTE: change this password when deployed online to restrict access to limited number of managers





65

OVERVIEW OF GEOSERVER DATA UPDATING

1. Accessing WMS and WFS layer on QGIS
2. Edit data on QGIS
3. Republishing data
4. Updating styles
5. Viewing updated data on Geoserver or web application

66

PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

1. Accessing WMS and WFS layers with QGIS

QGIS overview

- Quantum GIS (QGIS) is a user-friendly open source geographic information system (GIS).
- QGIS runs on Linux, Unix, Mac OSX, and Windows and supports numerous vector, raster, and database formats and functionalities.
- The current stable version of QGIS is available for download from https://www.qgis.org/en/site/forusers/download.html

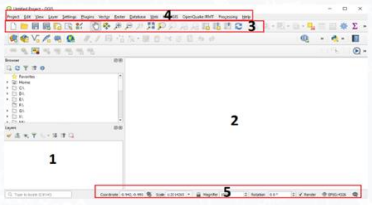
67

PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

1. Accessing WMS and WFS layers with QGIS

QGIS overview / Interface

- 1 - **Map Legend**: Lists all the data layers in the project. The style and labelling of the layer can be modified.
- 2 - **Map View**: The map displayed in this window will depend on the vector and raster layers selected.
- 3 - **Toolbar**: Provides access to most functions and tools for interacting with the displayed layer.
- 4 - **Menu Bar**: Provides access to various QGIS features using a standard hierarchical menu.
- 5 - **Status Bar**: The status bar shows current position in map coordinates of the mouse pointer.



Toolbars

68

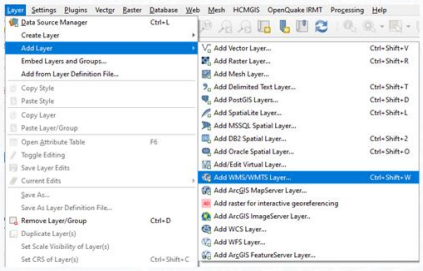
PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

1. Accessing WMS and WFS layers with QGIS

Viewing WMS layer on QGIS

A WMS (Web Map Service) publishes an image of a data set that cannot be edited. To add WMS layers to QGIS follow the steps below:

- Step 1: To add a WMS layer from the menu, choose ***Layer → Add Layer → Add WMS/WMTS Layer***



69

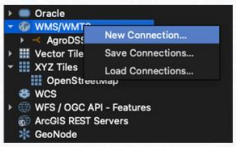
PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

1. Accessing WMS and WFS layers with QGIS

Viewing WMS layer on QGIS

- Step 2: A **Data Source Manager** dialog will appear, select **New**

- Alternatively, right click on the **WMS/WMST** button on the *Browser* window and select **New Connection…**



70

PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

1. Accessing WMS and WFS layers with QGIS

Viewing WMS layer on QGIS

- Step 3: In the **Create a new WMS Connection** pop-up:
  - add a **Name** for your service, such as *AGROCAM DATA* and
  - add the service **URL**: https://geoagrocam1.icem.com.au/geoserver/wms
  - and then click **OK**

71

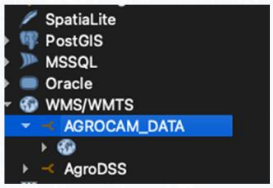
PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

1. Accessing WMS and WFS layers with QGIS

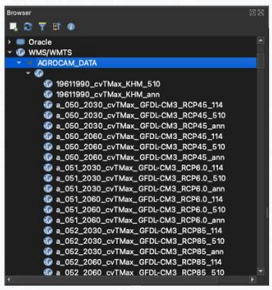
Viewing WMS layer on QGIS

Step 4:

- Now you can see new WMS connection on the *Browser* window.



- Click on the connection, a series of layers will be listed as below



- Note: WMS image are viewable by all users, whether logged in or not, but no actual data is accessible

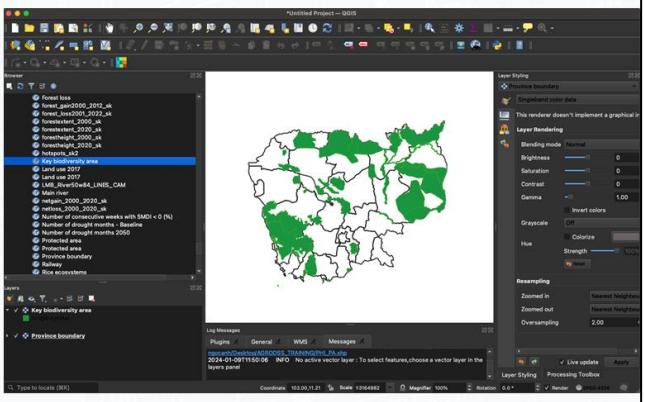
72

PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

1. Accessing WMS and WFS layers with QGIS

Accessing WFS layer on QGIS

- Step 5: Drag a layer to Layer window to see its display. For example, *Key Biodiversity Area* and *Province boundary*



73

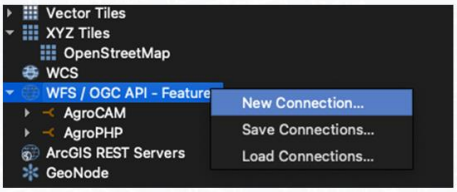
PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

1. Accessing WMS and WFS layers with QGIS

Accessing WFS layer on QGIS

A Web Feature Service (WFS) publishes and allows editing of the vector source data (e.g., shapefiles) that can be loaded directly into QGIS

- Step 1: To add a WFS layer from the menu, right click on the **WFS/OGC API - Features** button on the *Browser* window and select **New Connection...**



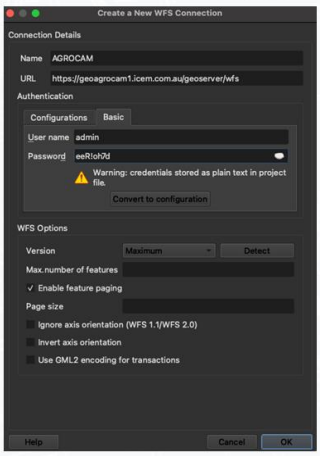
74

PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

1. Accessing WMS and WFS layers with QGIS

Accessing WFS layer on QGIS

- Step 2: In the **Create a New WFS Connection** dialog
  - Add a name for your service, such as *AGROPHP DATA*
  - Enter the service URL: https://geoagrophp1.icem.com.au/geoserver/wfs
  - On Authentication, select tab Basic and then enter your username and password. Use same *admin* with pdw: *eeR!oh7d* as login in GeoServer
  - Then click on OK
  - A box messages will appear, click **OK**



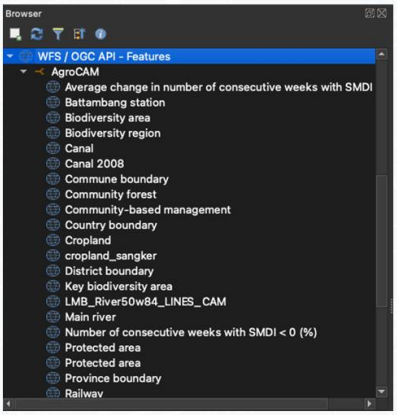
75

PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

1. Accessing WMS and WFS layers with QGIS

Accessing WFS layer on QGIS

Step 3: Expanding the list of layers from the Browser window



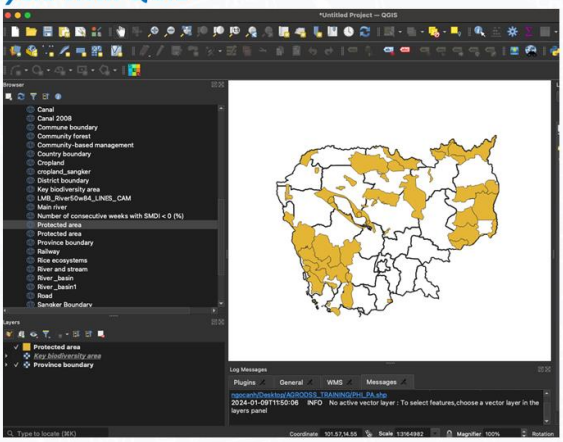
76

PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

1. Accessing WMS and WFS layers with QGIS

Accessing WFS layer in QGIS

- Step 4: After connecting to GeoServer, a series of layers will be listed. Drag a layer to the **Layer** window to display. For example, Protected area as shown.

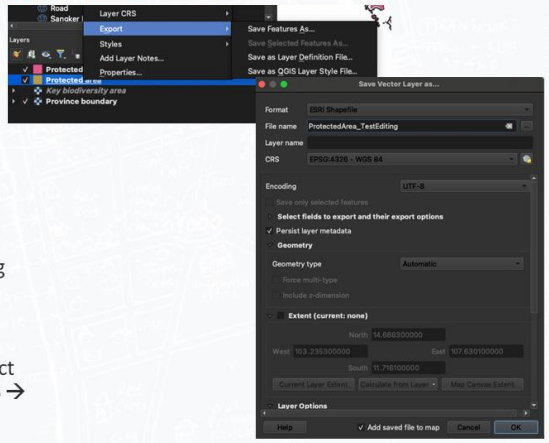


PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

1. Accessing WMS and WFS layers with QGIS

Accessing WFS layer on QGIS

- Step 5: A WFS service enables downloading a copy of the data for offline use:
  - Right click on the layer and select **Export → Save Features As…**
  - In the **Save Vector Layer as…** dialog, there are some exporting formats including **ESRI Shapefile** and **GeoJSON**.
  - Navigate to a folder to save the layer, specify file name and select suitable CRS such as EPGS: 4326 → Click **OK**



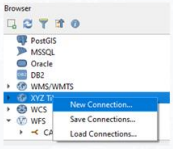
78

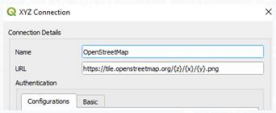
PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

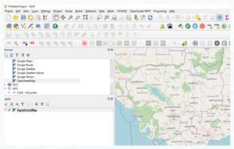
1. Accessing WMS and WFS layers with QGIS

Accessing WFS layer in QGIS

- After adding a data layer to QGIS via WFS, it can be edited in the same way as an offline layer.
- To support the editing process, you can add a base map such as OSM or Google Map to QGIS by the following steps:
  - Go to XYZ Tile in Browser window, right click and select New connection
  - Enter Name and URL of the base map
    - OSM: https://tile.openstreetmap.org/{z}/{x}/{y}.png
    - Google Map: http://mt1.google.com/vt/lyrs=m&x={x}&y={y}&z={z}
    - Google Satellite: http://mt1.google.com/vt/lyrs=y&x={x}&y={y}&z={z}
  - Click on XYZ Tile → Select a base map → Drag base map to **Layer** Window







79

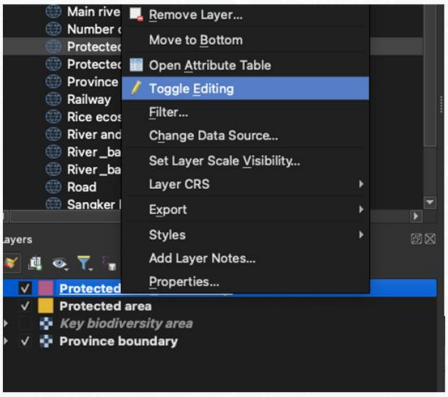
PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

1. Accessing WMS and WFS layers with QGIS

Accessing WFS layer on QGIS

This following steps will guide a user how to edit a road layer.

- Drag Protected Area_TestEditing layer from **WFS** to **Layer** window
- Start editing by right-clicking on layer → select **Toggle Editing**



80

PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

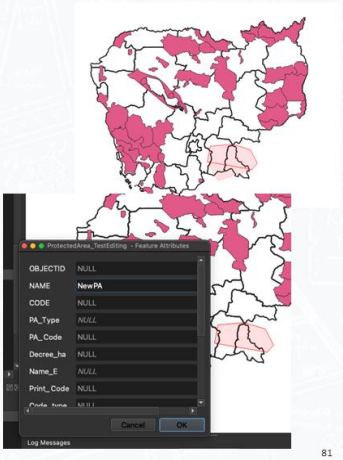
1. Accessing WMS and WFS layers with QGIS

Accessing WFS layer on QGIS

Adding features

- Step 1: The editing tool panel will appear at the top left of the screen. Click on the icon to add a new polygon or editing existing polygon of PA

- Step 2: To add a polygon, keep on left-clicking for each additional point you wish to capture, and press right mouse button to finish.
  - then a **Features Attribute** box will appear, enter information of PA attribute such as Name, PA_Code… → **OK**



81

PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

1. Accessing WMS and WFS layers with QGIS

Accessing WFS layer on QGIS

- Step 3**:** After finishing, click **Save** icon as below to save all created information. The online layer will be automatically updated.

- Step 4**:** Click on **Toggle** icon on toolbar to finish the polyline update. You can go to the GeoServer site (Layer Preview) to see the update

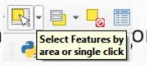
82

PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

1. Accessing WMS and WFS layers with QGIS

Accessing WFS layer on QGIS

Updating of features

- Click on **Select Features** button 

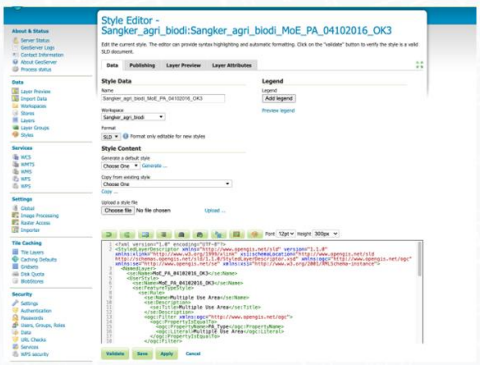
on toolbar to select the feature(s) on the source layer
- Now you can see all options of Editor toolbar with features to move, cut, and paste.

Edit attribute Cut Copy Paste

83

PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

3. Updating styles on GeoServer

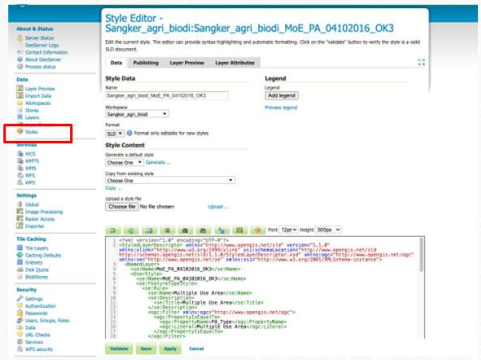


84

PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

3. Updating styles on Geoserver

- Step 1. From the GeoServer Welcome Page navigate to **Data** → **Styles** to see list of existing styles.

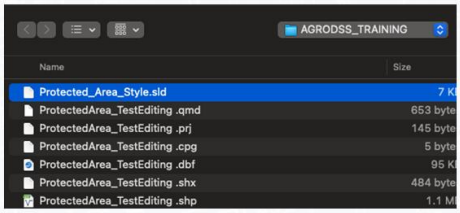


85

PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

3. Updating styles on Geoserver

- Step 2: click on the style on the **Style Name** column
- Step 3: On the **Edit Style** dialog, go to **upload a style** and click on **Choose File**
- Step 4: **Upload** a local file that contains the style. The SLD file can be generated on QGIS (see next section)



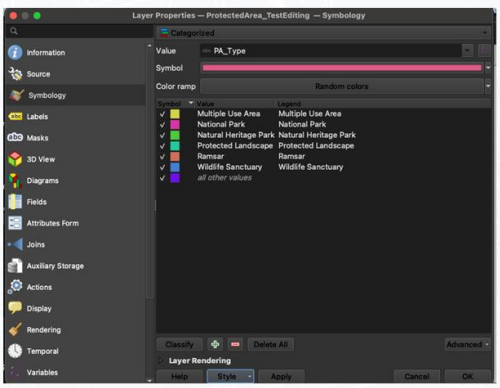
86

PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

3. Updating styles on Geoserver

Generating styles with QGIS

- Step 1: Upload layer to GeoServer and then open layer through WFS in **QGIS** (minimum version 3.0)



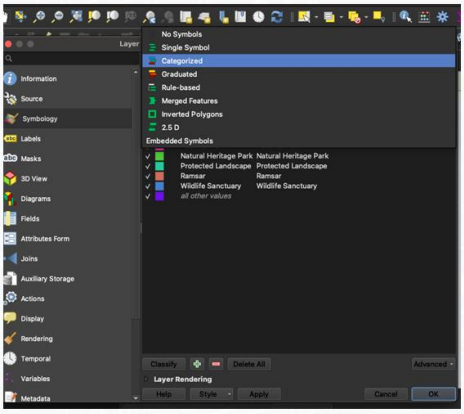
87

PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

3. Updating styles on Geoserver

Generating styles with QGIS

- Step 2: Double click the layer to open the **Properties** dialog and choose the **Symbology** section to change layer style



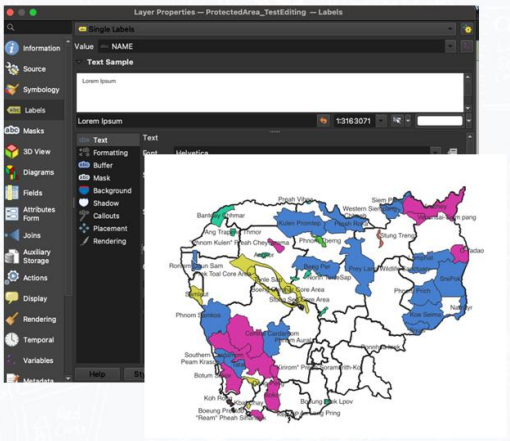
88

PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

3. Updating styles on Geoserver

Generating styles with QGIS

- Step 3: Go to **Labels** section, choose **Single labels**, label with the *NAME* attribute and choose your preferred text rendering options, as shown in figure
- The text, font and formatting of label can be modified.
- The style of layer can be seen in QGIS.



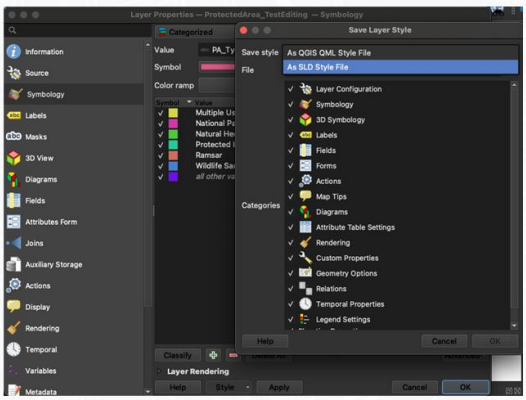
89

PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

3. Updating styles on Geoserver

Generating styles with QGIS

- Step 4: Go back the **Properties** dialog, from the bottom of the **Symbology/ Label** sections, choose **Style → Save Style**
- Step 5: Choose file export as SLD format → **OK**



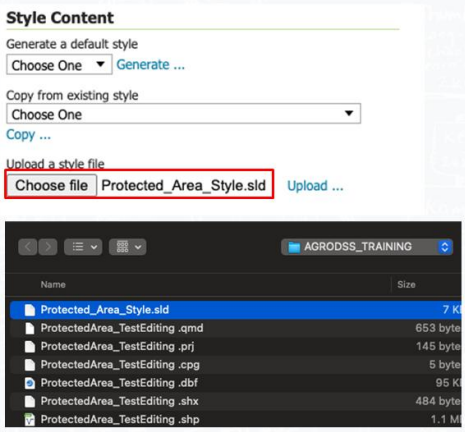
90

PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

3. Updating styles on Geoserver

Generating raster styles with QGIS

- Go back to Geoserver and go to **Style** → Click on an existing style to be updated
- On the **Edit Style** dialog, go to **Upload a style** and click on **Choose File**
- **Upload** a local file that contains the style. The SLD file can be generated on QGIS



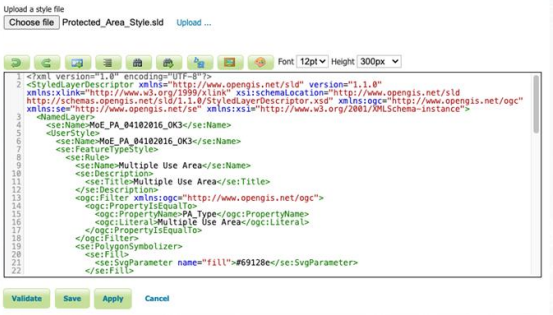
91

PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

3. Updating styles on Geoserver

Generating raster styles with QGIS

- When uploading an SLD style, a script will overwrite on the box at the bottom of Style Editor. A box will appear to confirm your overwrite→ Click **OK**
- This box will allow for direct editing.



92

PRACTICAL EXERCISE: GEOSERVER DATA UPDATING

3. Updating styles on Geoserver

Generating raster styles with QGIS

- During editing and especially after editing is complete, you will want to check validation of the syntax. This can be done by clicking the **Validate** button at the bottom. If no errors are found, you will see this message.

- Now click **Apply** to add new style

93

ADDITIONAL SUPPORT

There are various online resources, including active communities of users and developers. Here are some of the official information resources:

- **Docker**: https://docs.docker.com
- **GeoServer**: https://docs.geoserver.org
- **R**: https://cran.r-project.org/manuals.html
- **RStudio**: https://rstudio.cloud/learn/primers
- **Shiny web application framework**: https://shiny.rstudio.com/tutorial
- **Shiny Server**: https://docs.rstudio.com/shiny-server
- **Traefik**: https://github.com/traefik/traefik

94

Annex III: Deployment and Maintenance of R Shiny Application (Production Mode)

AGENDA

Session 3: Deployment and maintenance of R Shiny frontend (production mode)		
13:30 – 14:00	Introduction to GeoServer deployment and maintenance	Richard Cooper
14:00 – 15:00	Practical exercise: Shiny app deployment	Richard Cooper and participants
15:00 – 15:15	Tea/coffee break	
15:15 – 17:00	Practical exercise: Shiny app deployment	Richard Cooper and participants
17:00 – 17:30	Discussion/questions and wrap-up	ICEM team

EXPECTED OUTPUTS

1. Participants to gain practical experience in setting up the Shiny application in production mode
2. Participants to gain knowledge in using Docker, R and the Shiny web application framework software

OUTLINE

1. Overview of the software application framework
2. Shiny app deployment and management

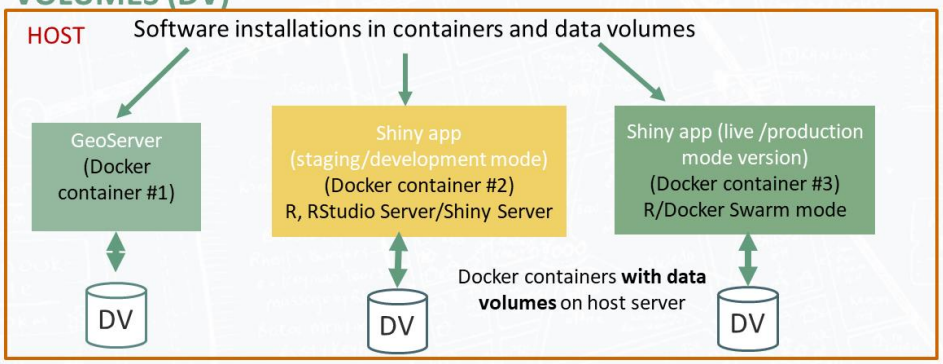
4

1. OVERVIEW OF ATLAS APPLICATION FRAMEWORK

- Shiny app development process
- System set up of Docker containers and data volumes
- Scaling Shiny app frontend

5

SYSTEM SET UP OF DOCKER CONTAINERS AND DATA VOLUMES (DV)



6

SHINY APP DEVELOPMENT PROCESS

Development mode offline (RStudio)

Local/offline development on laptops by 1+ developers

Staging/ development mode online (RStudio Server/ Shiny Server)

- Prototypes uploaded here for testing and review including by client

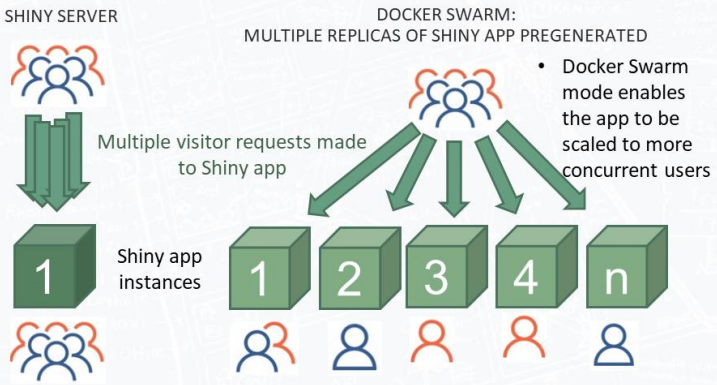
Final release version

Production mode online (Docker swarm cluster)

7

SCALING SHINY APP FRONTEND

- R is single threaded process: multiple requests to the Shiny app are processed sequentially.
- App responsiveness may be impacted by multiple concurrent users
- App in this set up (as used in Shiny Server Docker container for development) cannot be scaled effectively to more users



- Docker Swarm mode enables the app to be scaled to more concurrent users

8

MINIMUM SPECIFICATIONS: MEMORY

- Memory usage of Shiny apps (1 x Docker swarm container):
- $ docker stats --no-stream

```
rcooper@ubuntu-cloud-2:~$ docker stats --no-stream
CONTAINER ID   NAME                                                                CPU %     MEM USAGE / LIMIT     MEM %     NET I/O
        BLOCK I/O         PIDS
8decbb18ebf7   agrophp-geo_agrophp_geo_service.1.vz37swokybf3oohp5w677fi1z   0.74%     1.68GiB / 30.6GiB     5.49%     16.9kB /
1.23kB    9.6MB / 171MB     68
7206982ab4f   agrophp-web_dataservice_agrophp.1.remchx58d0sms55q7s7b0k2xd    0.17%     572.5MiB / 30.6GiB    1.83%     380kB / 5
0.7MB     4.44MB / 1.11MB   12
c8c98044a4a2   agrocam-web_dataservice_agrocam.1.4kukz2dfvw0tzuldx2duijge9   0.22%     499MiB / 30.6GiB      1.59%     548kB / 4
2.3MB     12.3kB / 53.2kB   12
29237d32269f   agrocam-geo_agrocam_geo_service.1.fyhqauj9yoktex0xsqhhdc4uz   0.35%     2.747GiB / 30.6GiB    8.98%     170kB / 1
.13MB     1.24MB / 457MB    68
```

CPU usage %

Memory usage

Memory usage %

9

MINIMUM SPECIFICATIONS: DISK SPACE

- **To view disk space usage** of **Shiny app Docker image** and **container**:
  $ docker ps --size <- *virtual* refers to total space used by Docker image and container (i.e., read-only and writable layers). The other value refers to container (i.e., writable layer) only.

```
rcooper@ubuntu-cloud-2:~$ docker ps --size
CONTAINER ID   IMAGE                                          COMMAND                  CREATED        STATUS        PORTS
     NAMES                                                                                SIZE
8decbb18ebf7   docker.osgeo.org/geoserver:2.23.1               "/bin/sh -c /opt/sta…"   47 hours ago   Up 47 hours
     agrophp-geo_agrophp_geo_service.1.vz37swokybf3oohp5w677fi1z   27.5MB (virtual 1.04GB)
72066982ab4f   shiny:agrophp_r432_ub2204                      "R -e shiny::runApp(…"   2 days ago     Up 2 days     3838/tcp
     agrophp-web_dataservice_agrophp.1.remchx58d0sms55q7s7b0k2xd   33.1kB (virtual 1.94GB)
c8c98044a4a3   shiny:agrocam_r432_ub2204                      "R -e shiny::runApp(…"   2 days ago     Up 2 days     3838/tcp
     agrocam-web_dataservice_agrocam.1.4kukz2dfvw0tzuldx2duijge9   33.1kB (virtual 1.94GB)
29237d522697   geoserver_ub2204_tomcat8500:v2.22.4_https      "/bin/sh -c /opt/sta…"   2 days ago     Up 2 days
     agrocam-geo_agrocam_geo_service.1.fyhqauj9yoktex0xsqhhdc4uz   96.5kB (virtual 1.73GB)
```

$ docker images

```
rcooper@ubuntu-cloud-2:~$ docker images
REPOSITORY                 TAG                 IMAGE ID       CREATED        SIZE
postgis/postgis            <none>              aa90f7108c98   6 days ago     594MB
shiny                      agrophp_r432_ub2204 26c604100c5d   10 days ago    1.94GB
shiny                      agrocam_r432_ub2204 a9870b183349   10 days ago    1.94GB
```

10

MINIMUM SPECIFICATIONS: SHINY APP SIZE

Path to Shiny app, which may vary depending on your installation:

$ cd /home/shiny/webapps/agrocam

$ sudo du -sh agrocam

~1GB (incl. renv directory containing R package libraries)

11

2. SHINY APP DEPLOYMENT AND MAINTENANCE

- 1. Download Shiny app Docker image, Shiny app, and Shiny app Dockerfile
- 2. On host set up Shiny user / Shiny-app group (LINUX ONLY)
- 3. Migrate Shiny app to host machine
- 4. Load and run Docker image
- 5. Install app in R from **within** Docker container
- 6. Enable Docker Swarm mode to run the Shiny application in production
- 7. Deploy Docker Swarm from host
- 8. Maintenance and upgrading

12

(1) DOWNLOAD SHINY APP DOCKER IMAGE, SHINY APP (AND ASSOCIATED DOCKERFILE)

Download from cloud
Link: https://1drv.ms/f/s!AoHzL3uXbH31hKtU2sWCjGu-U94QTQ?e=W7FUoe
Password: workshop_manupali

13

(2) ON HOST SET UP SHINY USER / SHINY-APP GROUP (LINUX ONLY)

If not already existing on host machine, add shiny user and shiny-app group with IDs as follows:

Group: shiny-app (GID = 5020)
User: shiny (UID = 990)

```
$ sudo useradd -m -u 5001 shiny
$ sudo groupadd -g 5020 shiny-app
$ usermod -aG shiny-app shiny <- add shiny to shiny-app group
```

OR, alternatively, if host already has Shiny user or shiny-app group with different UID/GIDs,

(A) Modify UID/GIDs in the container to match those on host and create a new Docker image (**see next slide**)

(B) Build a new Docker image and use 'docker run' to create new container (**see details in downloaded Dockerfile**)

14

(2) ON HOST SET UP SHINY USER / SHINY-APP GROUP (LINUX ONLY)

(A) If host already has Shiny user or Shiny-app group with different UID/GIDs, modify UID/GIDs in current container and create a new image using 'docker commit'

(a) Run container

```
$ docker run --name [CONTAINER_NAME] -it -d -v '/home/shiny/webapps:/home/shiny/webapps' -p 9020:3838 [IMAGE_NAME] bash
```

(b) Access running container from host to change UIDs/GID:

```
$ docker exec -it -u root [CONTAINER_NAME] bash
```

(c) Change shiny user UID and shiny-app group GID

```
# usermod -u 1001 shiny
# groupmod -g 1002 shiny-app
```

(d) Change file/folder ownership to new UID/GID

```
# find /home/shiny/webapps/[proj name] -uid 5001 -exec chown -v -h 1001 '{}' \;
# find / -not -path "/home/shiny/webapps/*" -uid 5001 -exec chown -v -h 1001 '{}' \;

# find /home/shiny/webapps/[proj name] -gid 5020 -exec chgrp -v -h 1002 '{}' \;
# find / -not -path "/home/shiny/webapps/*" -gid 5020 -exec chgrp -v -h 1002 '{}' \;
```

Enter new UID/GID to match host

NOTE: above commands avoid changing any content in linked volume on host

15

(2) ON HOST SET UP SHINY USER / SHINY-APP GR0UP (LINUX ONLY)

(e) The modified container can then be converted to a new image. Need to also modify the 'bash' command to 'R -e shiny:runApp…' using 'docker commit':

```
$ docker commit -p [CONTAINER_NAME] [NEW_IMAGE_NAME]
e.g.
docker commit -p --change='Cmd [ "R", "-e",
"shiny::runApp('\''/home/shiny/webapps/agrocam'\'')" ]' shiny_agrocam img_shiny_agrocam
```

In this example:
[CONTAINER_NAME] = shiny_agrocam
[NEW_IMAGE_NAME] = img_shiny_agrocam

The image name of 'img_shiny_agrocam' should be in the docker-compose.yml for running Docker swarm

16

(3) MIGRATE SHINY APP TO HOST MACHINE

Saving the Shiny app as tar.gz:
Participants don't need to do the following steps as **the *tar.gz (or *zip) file can be downloaded from cloud link** above (but is included for future reference):

- In RStudio console or in open terminal with R running:
  $ cd to project app directory (e.g., ../[proj_name]
  $ R
  > renv::snapshot() <- to create package list of in lock file (renv.lock)
- Create archive of app on CURRENT HOST machine:
  In RStudio Server terminal or server's command line terminal:
  $ cd ../[proj_name]
  Example:
  $ sudo tar -czvf img_shiny_PROJNAME__DDMMYYYY.tar.gz . --exclude "*.tar.gz" --exclude "*backup*" --exclude "renv"
- Check contents of archive:
  $ sudo tar -tvf img_shiny_PROJNAME_DDMMYYYY.tar.gz

17

(3) MIGRATE SHINY APP TO HOST MACHINE

Extract the Shiny app:

- On host (Linux) create folder path /home/shiny/webapps/agrocam
- On host (Windows) create path, for example, D:\path\to\shiny\webapps\agrocam
- On host extract tar.gz to project app **directory**
  $ cd /home/shiny/webapps/agrocam
  $ sudo tar -xzvf ~ /NAME_OF_BACKUP.tar.gz
  or $ NAME_OF_BACKUP.zip

(ii) On host Windows: Use 7-zip or alternative: https://www.7-zip.org/download.html

Check contents are extracted into project directory:
e.g., /home/shiny/webapps/PROJNAME/www /home/shiny/webapps/PROJNAME/app.R etc
Also copy renv.lock to PROJNAME directory

- Modify permissions of extracted app (for LINUX ONLY):
  $ cd /home/shiny/webapps/agrocam
  agrocam $ sudo chown -R shiny:shiny-app . <- modify group ownership (don't miss '.' at end)
  agrocam $ sudo chmod -R g+rws . <- modify group permissions

18

(4) LOAD AND RUN DOCKER IMAGE

Saving the Docker image as a tar.gz archive:

- Participants don't need to do this, but details included here for reference on how to save a Docker image for migrating to another machine
- On ICEM's Hetzner cloud server the Docker image was saved as follows (and then moved using scp to the destination server):

Example:

```
$ docker save -o OUTPUT.tar [IMAGE_NAME]
$ gzip < OUTPUT.tar >  OUTPUT.tar.gz
```

Example:

```
docker save -o img_agrocam_shinyapp_07dec2023.tar shiny:agrocam_r432_ub2204
gzip < img_agrocam_shinyapp_07dec2023.tar > img_agrocam_shinyapp_07dec2023.tar.gz
```

(4) LOAD AND RUN DOCKER IMAGE (LINUX)

- Download Docker image to computer:
  for example:
  img_agrocam_shinyapp_07dec2023.tar.gz

  Change to directory containing Docker image and load Docker image:
  $ docker load -i [Docker image name].tar.gz

20

(4) LOAD AND RUN DOCKER IMAGE (LINUX)

Check image is loaded:

$ docker images

Image name: shiny:agrocam_r432_ub2204

```
rcooper@richard-XPS:/home/tomcat/agroecolcam/geoserver_data$ docker images | grep agrophp
shiny                        agrophp_r432_ub2204           26c684109e6d   3 days ago     1.94GB
rcooper@richard-XPS:/home/tomcat/agroecolcam/geoserver_data$
```

Run image (append **bash** to command):

$ docker run --name shiny_agrocam -it -d -v '/home/shiny/webapps:/home/shiny/webapps' -p 9447:3838 shiny:agrocam_r432_ub2204 **bash**

NOTE: IMAGE_NAME: shiny:agrocam_r432_ub2204
9447 is the host port, and
the container created on running the command is shiny_agrocam

Show container is created and running:

$ docker ps

```
rcooper@richard-XPS:/home/tomcat/agroecolcam/geoserver_data$ docker ps | grep shiny_agrophp
3dcfe8058783   shiny:agrophp_r432_ub2204          "bash"                   3 days ago          Up 52 minutes
  0.0.0.0:9101->3838/tcp, :::9101->3838/tcp                                                  shiny_agrophp
```

(4) LOAD AND RUN DOCKER IMAGE (WINDOWS)

- Copy Docker image to computer:
  shiny:agrocam_r432_ub2204.tar.gz
- Change to directory containing Docker image and load Docker image:
  ➢ docker load -i "D:\path\to\imagedownload\[IMAGE_NAME]"
- Check image is loaded: > docker images

```
rcooper@richard-XPS:/home/tomcat/agroecolcam/geoserver_data$ docker images | grep agrophp
shiny                                agrophp_r432_ub2204            26c684109e6d   3 days ago     1.94GB
rcooper@richard-XPS:/home/tomcat/agroecolcam/geoserver_data$
```

- Run image (with **bash** appended):

  docker run -ti -d -v "D:\path\to\shiny\webapps:/home/shiny/webapps" --name shiny_agrocam

  -p 9447:3838 [IMAGE_NAME] **bash**

  e.g. [IMAGE_NAME]: shiny:agrocam_r432_ub2204

  Note: 9447 is the host port, and the container created on running the command is shiny_agrocam
- Show container is created and running:
  > docker ps

```
rcooper@richard-XPS:/home/tomcat/agroecolcam/geoserver_data$ docker ps | grep shiny_agrophp
3dcfe8058783   shiny:agrophp_r432_ub2204            "bash"                   3 days ago          Up 52 minutes
   0.0.0.0:9101->3838/tcp, :::9101->3838/tcp                                                   shiny_agrophp
```

(5) INSTALL APP IN R FROM WITHIN DOCKER CONTAINER

- **ON HOST** access running container
  $ docker exec -it -u root shiny_agrocam

- **WITHIN CONTAINER** install R packages:
  # cd /home/shiny/webapps/agrocam
- Restore package libraries (versions included in renv.lock file) :
  $ R <- start R
  > install.packages("renv")
  > renv::settings$use.cache(FALSE) <- to ensure local renv repository is used for packages
  > renv::activate()

If cannot connect to above repository, chooseCRANmirror() and run install.packages("renv")

(5) INSTALL APP IN R FROM WITHIN DOCKER CONTAINER

- Check **library paths** are in renv path (i.e., packages are installed under renv directory)

> .libPaths() <- check path to library is in local **renv** folder, and if not set path as below
> .libPaths("/home/shiny/webapps/agrocam/renv/library/R-4.3/x86_64-pc-linux-gnu")
> renv::restore() <- note if this gives errors (see next slide for solution)

Run and test app works
> shiny::runApp('/home/shiny/webapps/agrocam')
> install.packages("codetools") <- may need to install this to address warning messages

Alternative command from bash prompt:
[# R -e "shiny::runApp('/home/shiny/webapps/agrocam')"]

(5) INSTALL APP IN R FROM WITHIN DOCKER CONTAINER

- Test app is running in browser: e.g., localhost:9447 <- specified port as included in Docker run command above.
- Install any missing packages (e.g. if renv.lock is not up-to-date) and until no errors observed after running renv::restore()
- May need to resolve installation as follows if renv::restore() does not work as expected:
  > install.packages("remotes")
  > install.packages('terra', repos='https://rspatial.r-universe.dev')
  > remotes::install_github('r-spatial/sf', configure.args= '--with-proj-lib=/lib/x86_64-linux-gnu')

  Now install packages separately (but not 'terra', 'sf') n<- see global.R file
  > install.packages("package_name")
  OR
  > shiny:runApp() <- to see if there are any missing packages

(6) ENABLE DOCKER SWARM MODE TO RUN THE SHINY APPLICATION IN PRODUCTION

Using Traefik reverse proxy:
Server:
docker-compose-shinyapp.yml and
docker-compose-traefik.yml

Laptop:
docker-compose-shinyapp-laptop.yml and
docker-compose-traefik-laptop.yml

- This set up ensures that a **sticky cookie** is enabled for the Shiny application, which can be important for ensuring that a user retains connection to a single replica, as would be required for connecting to a database for example.
- However, for laptop installations (this workshop) these two files are different from the server set up.

(6) ENABLE DOCKER SWARM MODE TO RUN THE SHINY APPLICATION IN PRODUCTION

- Set up Docker swarm (ON HOST MACHINE):
  $ cd /home/shiny/webapps/agrocam
  $ docker swarm init
  $ docker swarm join-token manager
  $ docker info
- On host (Windows) go to for example, D:\path\to\shiny\webapps\agrocam
- Check Docker networks: $ docker network ls
- Create Docker overlay (to connect all containers):
  $ docker network create -d overlay --attachable net2
- Check all containers access network:
  $ docker network inspect net2

Check config matches in docker-compose files

- Traefik is used as reverse proxy and load balancer for docker swarm set up:
  https://github.com/traefik/traefik
  - Download Traefik Docker image on host machine:
    $ docker pull traefik:v2.10.6
  - Alternative to Traefik is Nginx but potentially more complicated to configure

27

(6) ENABLE DOCKER SWARM MODE TO RUN THE SHINY APPLICATION IN PRODUCTION

For **laptop installation (for workshop)**

28

docker-compose-shinyapp-laptop.yml

```
version: "3.7"
services:
  dataservice:
    image: shiny:agrocam_r432_ub2204
    deploy:
      replicas: 2
      labels:
        - "traefik.enable=true"
        - "traefik.docker.network=net2"
        - "traefik.http.routers.dataservice.entrypoints=web"
        - "traefik.http.routers.dataservice.rule=Host(`127.0.0.1`)"
        - "traefik.http.middlewares.dataservice.stripprefix.prefixes=/"
        - "traefik.http.routers.dataservice.middlewares=dataservice@docker"
        - "traefik.http.services.dataservice.loadbalancer.server.port=3838"
        - "traefik.http.services.dataservice.loadbalancer.sticky=true"
        - "traefik.http.services.dataservice.loadbalancer.sticky.cookie.name=stickycookie"
```

29

docker-compose-shinyapp-laptop.yml

```
        - "traefik.http.services.dataservice.loadbalancer.sticky.cookie.secure=false"
        - "traefik.http.services.dataservice.loadbalancer.sticky.cookie.httpOnly=true"
      restart_policy:
        condition: on-failure
      update_config:
        delay: 2s
    volumes:
      - /home/shiny/webapps/agrocam:/home/shiny/webapps/agrocam
    #  - C:\windows-path\shiny\webapps\ agrocam :/home/shiny/webapps/agrocam
  networks:
      - net2
networks:
  net2:
    driver: overlay
    external: true
```

Make sure volumes path matches path to Shiny app and OS (Windows? Linux?)

30

docker-compose-traefik-laptop.yml

```
version: "3.7"
services:
  traefik:
    image: traefik:v2.10.6
    deploy:
      restart_policy:
        condition: any
      placement:
        constraints:
          - node.role == manager
      labels:
       - "traefik.http.services.traefik.loadbalancer.server.port=80"
       #- "traefik.http.services.traefik.loadbalancer.server.port=443"
```

31

docker-compose-traefik-laptop.yml

```
command:
    - "--log.level=DEBUG"
    - "--api.insecure=true"
    - "--providers.docker=true"
    - "--providers.docker.swarmMode=true"
    - "--providers.docker.exposedbydefault=true"
    - "--entrypoints.web.address=:80"
  ports:
    #- "8098:8098"
    - "9080:80"
    - "9020:8080"
    #- "443:443"
  volumes:
    - "/var/run/docker.sock:/var/run/docker.sock:ro"
  networks:
    - net2
networks:
  net2:
    driver: overlay
    external: true
```

32

To deploy/start Shiny app: run docker-compose files

$ cd /home/shiny/webapps/agrocam

- On host (Windows) go to for example, D:\path\to\shiny\webapps\ agrocam

$ docker stack deploy -c docker-compose-traefik-laptop.yml agrocam
$ docker stack deploy -c docker-compose-shinyapp-laptop.yml agrocam

Access Shiny app in browser **on laptop**:
http://127.0.0.1:9080

Visit Traefik app at port 9020 (e.g. on laptop: 127.0.0.1:9020)
Check Traefik logs: $ docker service logs traefik_traefik
Check Shiny app logs: $ docker service logs agrocam_dataservice

Ensure service names are as set up earlier

33

(6) ENABLE DOCKER SWARM MODE TO RUN THE SHINY APPLICATION IN PRODUCTION

For server installation

NOTE: some of the details of the yml files may need editing depending on the server configuration.

34

docker-compose-shinyapp.yml

```
version: "3.7"
services:
  dataservice:                                   <- Service name
    image: shiny:agrocam_r432_ub2204             <- Check
    deploy:
      replicas: 5                                <- Check
      labels:
        - "traefik.enable=true"
        - "traefik.docker.network=net2"
        - "traefik.http.routers.dataservice.rule= Host(`agrocam.icem.com.au.org`)"
        - "traefik.http.middlewares.opendata.stripprefix.forceSlash=false"
        - "traefik.http.routers.dataservice.middlewares=dataservice@docker"
        - "traefik.http.services.dataservice.loadbalancer.server.port=3838"
        - "traefik.http.services.dataservice.loadbalancer.sticky=true"
```

(change to appropriate subdomain.domain.top-level domain)

35

docker-compose-shinyapp.yml

```
        # - "traefik.http.services.dataservice.loadbalancer.sticky.cookie.secure=false"
        # - "traefik.http.services.dataservice.loadbalancer.sticky.cookie.httpOnly=true"
        # the following for https
        - "traefik.http.services.dataservice.loadbalancer.sticky.cookie.secure=true"
        - "traefik.http.routers.dataservice.entrypoints=websecure"
        - "traefik.http.routers.dataservice.tls=true"
        - "traefik.http.routers.dataservice.tls.certresolver=le"      <- Check
      restart_policy:
        condition: on-failure
      update_config:
        delay: 2s
    volumes:
      - D:\shiny\webapps\agrocam:/home/shiny/webapps/agrocam
#     - C:\windows-path\shiny\webapps\  agrocam :/home/shiny/webapps/agrocam
    networks:                                                         <- Check
      - net2
networks:                                                             <- Check
  net2:
    driver: overlay
    external: true
```

Make sure volumes path matches path to Shiny app and OS (Windows? Linux?)

36

docker-compose-traefik.yml

```
version: "3.7"
 services:
  traefik:
    image: traefik: v2.10.6
     restart_policy:
       condition: any
     placement:
       constraints:
         - node.role == manager
     labels:
     - "traefik.http.services.traefik.loadbalancer.server.port=80"
     # for https
     - "traefik.http.services.traefik.loadbalancer.server.port=443"
   command:
     - "--log.level=DEBUG"
     - "--api.insecure=true"
     #- "--api.insecure=false"
     - "--providers.docker=true"
     - "--providers.docker.swarmMode=true"
     - "--providers.docker.exposedbydefault=false"
```

Service name

Check

NOTE: download Traefik Docker image on host machine:
$ docker pull traefik:v2.10.6 (or latest version)

(if running on laptop comment out #)

37

docker-compose-traefik.yml

```
 - "--entrypoints.web.address=:80"
     - "--entrypoints.web.http.redirections.entrypoint.to=websecure"
     - "--entrypoints.web.http.redirections.entrypoint.scheme=https"
     - "--entrypoints.websecure.address=:443"
     # for SSL certificate
     - "--certificatesresolvers.le.acme.httpchallenge=true"
     - "--certificatesresolvers.le.acme.httpchallenge.entrypoint=web"
     - "--certificatesresolvers.le.acme.email=tho.ictu@gmail.com"
     - "--certificatesresolvers.le.acme.storage=acme.json"
     ##- --providers.docker.watch=true
   ports:
     - "80:80"
     - "9080:80"
     - "9020:8080"
     - "443:443"
   volumes:
     - "/var/run/docker.sock:/var/run/docker.sock:ro"
   networks:
     - net2
 networks:
   net2:
     driver: overlay
     external: true
```

(if running on laptop comment out # and change port setting)

Check ports

Check ports

Check ports

8080: Traefik console
9020: publish Traefik console externally on this port
443: https

38

(6) ENABLE DOCKER SWARM MODE TO RUN THE SHINY APPLICATION IN PRODUCTION

- To access the shiny app **from server**:
  Use these files as downloaded from ICEM server and modify accordingly (see above):
  docker-compose-shinyapp.yml
  docker-compose-traefik.yml

  Access app on server, e.g.:
  https://agrocam_rbco.denr.gov.ph

39

(7) DEPLOY DOCKER SWARM FROM HOST

Start Shiny app: run docker-compose files:
$ cd /home/shiny/webapps/agrocam

- On host (Windows) as created earlier, e.g., D:\path\to\shiny\webapps\agrocam

$ docker stack deploy -c docker-compose-traefik.yml agrocam
$ docker stack deploy -c docker-compose-shinyapp.yml agrocam

Show running Docker services:
$ docker service ls

```
rcooper@richard-XPS:/home/shiny/webapps/agrophp$ docker service ls
ID             NAME                      MODE         REPLICAS   IMAGE                          PORTS
zfwmsjbpl3ot   agrophp_dataservice       replicated   1/1        shiny:agrophp_r432_ub2204
i98v7to10rjy   traefik_traefik           replicated   1/1        traefik:v2.10.6                *:9020->8080/tcp, *:9080-
>80/tcp
```

Shiny app service is linked to 1 container (in this demo, but can scale to multiple replicas in yml file)
Traefik service linked to 1 container

40

(7) DEPLOY DOCKER SWARM FROM HOST

Show all running Docker containers:
$ docker container ls
Or
$docker ps

NOTE: in this example the Traefik service is called 'traefik', but it can be called anything, as specified in the traefik yml file. Shiny app service naming is specified similarly.

```
rcooper@richard-XPS:/home/shiny/webapps/agrophp$ docker container ls
CONTAINER ID   IMAGE                          COMMAND                  CREATED          STATUS          PORTS
                                  NAMES
cae0fe4685e3   shiny:agrophp_r432_ub2204      "R -e shiny::runApp(…"   14 minutes ago   Up 14 minutes   3838/tcp
                                  agrophp_dataservice.1.0su6d7jt5mgc6ffdzhbyvlzvx
afaa04241116   traefik:v2.10.6                "/entrypoint.sh --lo…"   7 hours ago      Up 7 hours      80/tcp
                                  traefik_traefik.1.wbiadiaqmx344e6ppb2hsp3ex
```

41

(7) DEPLOY DOCKER SWARM FROM HOST

Check logs for any errors:
$ docker service logs agrocam_dataservice <- use to check if Shiny app process

```
agrophp_dataservice.1.0su6d7jt5mgc@richard-XPS    | New names:
agrophp_dataservice.1.0su6d7jt5mgc@richard-XPS    | * `` -> `..1`
agrophp_dataservice.1.0su6d7jt5mgc@richard-XPS    | Warning: `includeHTML()` was provided a `path` that appears to be a complete HTML document.
agrophp_dataservice.1.0su6d7jt5mgc@richard-XPS    | x Path: www/home.html
agrophp_dataservice.1.0su6d7jt5mgc@richard-XPS    | i Use `tags$iframe()` to include an HTML document. You can either ensure `path` is accessible in y
our app or document (see e.g. `shiny::addResourcePath()`) and pass the relative path to the `src` argument. Or you can read the contents of `path` and
 pass the contents to `srcdoc`.
agrophp_dataservice.1.0su6d7jt5mgc@richard-XPS    |
agrophp_dataservice.1.0su6d7jt5mgc@richard-XPS    | Listening on http://0.0.0.0:3838
```

$ docker service logs proxy_traefik <- use to check Traefik process

42

(7) DEPLOY DOCKER SWARM FROM HOST

To remove service:

$ docker service rm agrocam_dataservice

$ docker service rm traefik_traefik

(to re-add the Docker swarm service use the docker stack deploy command above)

43

REMAINING SYSTEM ADMIN TASKS

NOTE: PORTS MY VARY DEPENDING ON FINAL SELECTED CONFIGURATIONS

Application ports:

- RStudio Server: 8787 (default) # check run command/docker compose yml file
- Shiny Server: 3838 (default) # only needed for online development version of Shiny app (check run command/docker compose yml file)
- GeoServer: 8080 (default) # check run command/docker compose yml file
- Shiny in Docker Swarm mode: 443 (as specified in docker-compose-traefik.yml/docker-compose.yml) <- production mode of Shiny app online
- Traefik: 9020: if using Traefik modify script (set --api-insecure=false) in docker-compose-traefik.yml
- Use Traefik to map ports appropriately
  - Note see Traefik scripts that ICEM uses on server
- To generate Docker images using a Dockerfile see downloaded files (also see online literature for more details)

- **Remember:**
  - Change admin password in GeoServer
  - Need to edit server.xml for https/http access by GeoServer?

44

REMAINING SYSTEM ADMIN TASKS

On server: renew SSL certificate:

- Schedule the following command every 2.5 months to renew SSL certificate

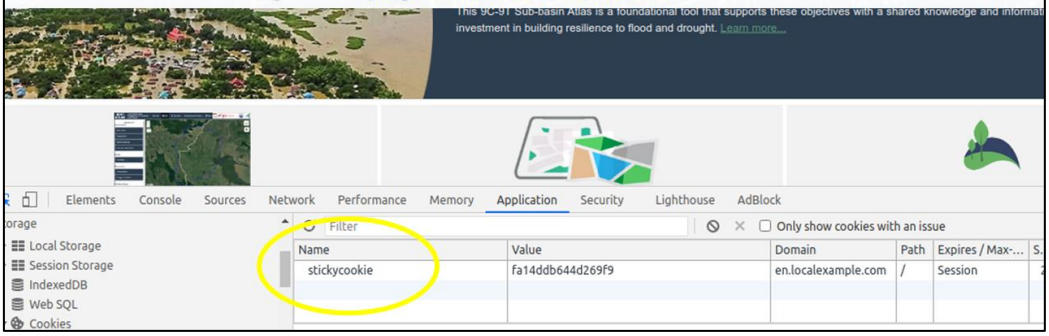
Go to directory containing docker-compose-traefik.yml file

docker stack deploy docker-compose-traefik.yml traefik

45

(6) ENABLE DOCKER SWARM MODE TO RUN THE SHINY APPLICATION IN PRODUCTION

- Traefik is used here as reverse proxy and is easier to configure than Nginx.
- It also sets a sticky cookie for each Docker replica to ensure that a visitor remains connected to a particular replica (instance of the Shiny app). The latter is not important for the current app, but is relevant, for example, if access to a database in future is required. Otherwise, the visitor may be directed to another replica container and will lose the database connection.
- To test if cookie is being set: Ref: https://github.com/traefik/traefik/issues/8546



MAINTENANCE

Problem: The Shiny application is becoming slow to respond.

- **Response:**

(i) Check the amount of RAM and CPU resources being used by the Docker containers (and contained apps):
$ docker stats --no-stream

(ii) This may also occur if more users are accessing the application. Modify the number of replicas in the **docker-compose-shinyapp.yml** and consider adding more RAM/CPU resources as more replicas will use more resources.

For example, to deploy 10 replicas:

```
deploy:
  replicas: 10
```

Then redeploy the Shiny app service (docker service rm … / docker stack deploy …)

47

MAINTENANCE

Problem: If the Shiny application becomes inaccessible or unresponsive

Response:

Obtain service name from $docker service ls

(i) Redeploy the Docker Shiny app service

- Check Docker service is running: $ docker service ls
- Stop both Traefik and Shiny app services: $ docker service rm [NAME OF SERVICE]
- Redeploy Traefik and Shiny app services:
  Change to Shiny app directory containing docker-compose yml files:
  agrocam $ docker stack deploy –c docker-compose.traefik.yml traefik
  agrocam $ docker stack deploy –c docker-compose.shinyapp.yml shiny_agrocam
- Check service is running: $ docker service ls
- Check containers are running: $ docker ps

48

MAINTENANCE

If problem persists:

- Check service logs:
  $ docker service logs [SERVICE NAME]

- Restart the Docker service (**note: this will stop all containers**)
  Windows server: open PowerShell as admin: restart-service *docker*
  Linux: $ sudo service docker restart or sudo systemctl restart docker

Remember to restart other previously running containers as a Docker service restart will stop all running containers (e.g., add to Linux cronjob/Windows restart for automatic restarts)

NOTE: If no changes have been made to the Docker images and docker-compose yml files, then check for changes to general server configuration.

49

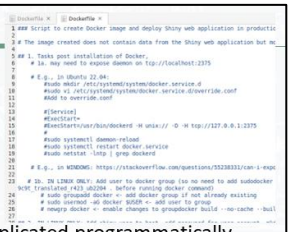
MAINTENANCE

Problem: Cannot access https/port 443

- **Response:**
- May occur if using free SSL certificate with 3-month validity
- Schedule the following command every 2.5 months to renew SSL certificate
- Go to directory containing docker-compose-traefik.yml file
- docker stack deploy docker-compose-traefik.yml traefik

50

FUTURE UPGRADING OF R AND PACKAGES



Two approaches:

1. Modify Docker file and build new Docker image
   See contents of Docker file (shared on download link) for more details
   This is the preferred approach as the Docker image can then be readily duplicated programmatically, rather than manually upgrading the Docker container and creating a new Docker image using the docker commit command (see 2 below).

2. Run docker container ('docker run …'), access the container using 'docker exec …', make the required modifications, then create a new Docker image from the container ('docker commit …'). This approach may not always work depending on the changes required.

After 1 or 2 above, address these tasks:
Modify image link in the docker-compose-shinyapp.yml
Finally redeploy the Docker Shiny app service as detailed earlier ('docker service rm … /docker stack deploy … ')

51

FUTURE DEVELOPMENT: NOTE ON RAM USAGE

- R Shiny apps are potentially heavy users of RAM memory
- When developing a Shiny app reduce RAM usage where possible (avoid loading unnecessary data, reduce image sizes, optimize code where possible)
- During development monitor app's memory usage using the R 'pryr' package
- Docker Swarm enables the Shiny app to be scaled to handle more concurrent users
  - ➢ specify number of replica containers in docker-compose-shinyapp.yml

52

USEFUL RESOURCES

- Docker: https://docs.docker.com
- GeoServer: https://docs.geoserver.org
- R: https://cran.r-project.org/manuals.html
- RStudio: https://rstudio.cloud/learn/primers
- Shiny: https://shiny.rstudio.com/tutorial
- Shiny Server: https://docs.rstudio.com/shiny-server
- Traefik: https://github.com/traefik/traefik

53

Annex IV: Deployment of R Shiny Application (Development Mode)

AGENDA

Session 4: Deployment of R Shiny frontend (development mode) and future app development		
13:30 – 14:00	Introduction to RStudio and app development	Richard Cooper
14:00 – 15:00	Practical exercise: RStudio deployment	Richard Cooper and participants
15:00 – 15:15	Tea/coffee break	
15:15 – 16:30	Practical exercise: RStudio deployment	Richard Cooper and participants
16:30 – 17:30	Discussion/questions and wrap-up Queries on server deployment	ICEM team

EXPECTED OUTPUTS

1. Participants to gain practical experience in deploying RStudio Server and the Shiny application (on a laptop and/or web server).
2. Participants to gain knowledge in managing RStudio Server and Docker software.
3. Participants to gain an insight into the R programming language, and RStudio software - an integrated development environment (IDE) for coding.

OUTLINE

1. Download files
2. Overview of software components
3. Overview of RStudio Server application
4. Overview of R and RStudio Server software sources
5. Deployment steps: RStudio Server and Shiny app
6. Deployment steps: Shiny Server
7. Q&A session

1. DOWNLOAD FILES

5

1. DOWNLOAD GEOSERVER DOCKER IMAGE, DATA, AND ASSOCIATED DOCKERFILE

Download from cloud

Link: https://1drv.ms/f/s!AoHzL3uXbH31hKtXMAXxdFPUz8CUGQ?e=7wPuMP

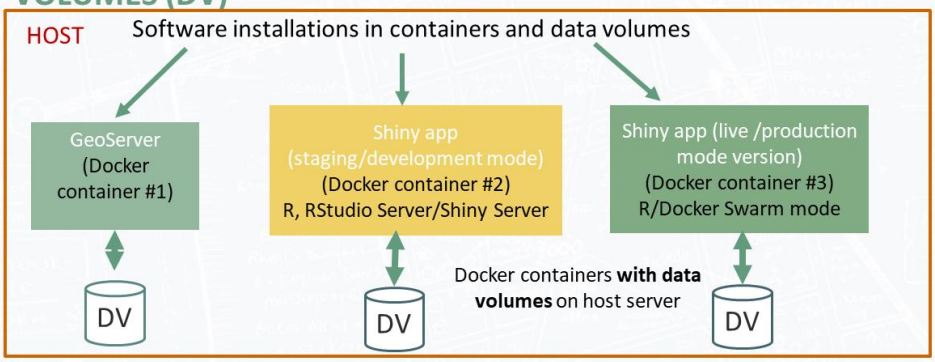
Password: workshop_sangker

6

2. OVERVIEW OF SOFTWARE COMPONENTS

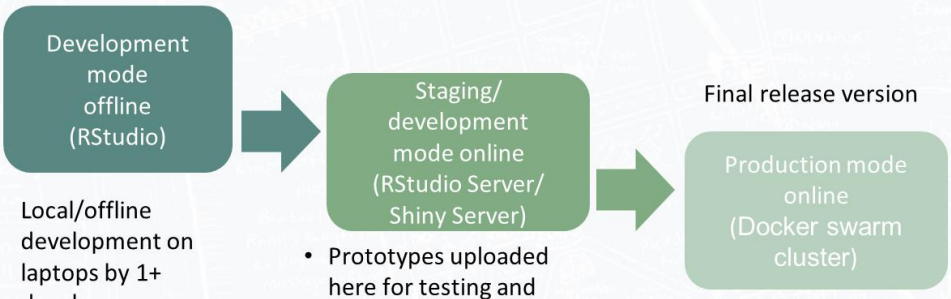
7

SYSTEM SET UP OF DOCKER CONTAINERS AND DATA VOLUMES (DV)



8

SHINY APP DEVELOPMENT PROCESS



Local/offline development on laptops by 1+ developers

- Prototypes uploaded here for testing and review including by client

9

3. OVERVIEW OF RSTUDIO SERVER

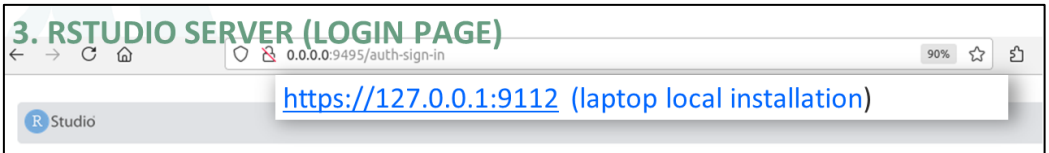
- USED FOR SHINY APPLICATION DEVELOPMENT

10

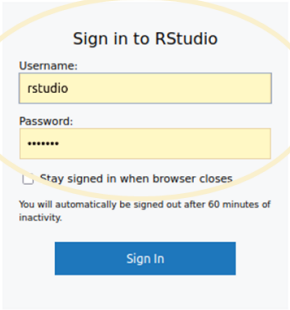
3. OVERVIEW OF RSTUDIO SERVER (USED FOR APPLICATION DEVELOPMENT)

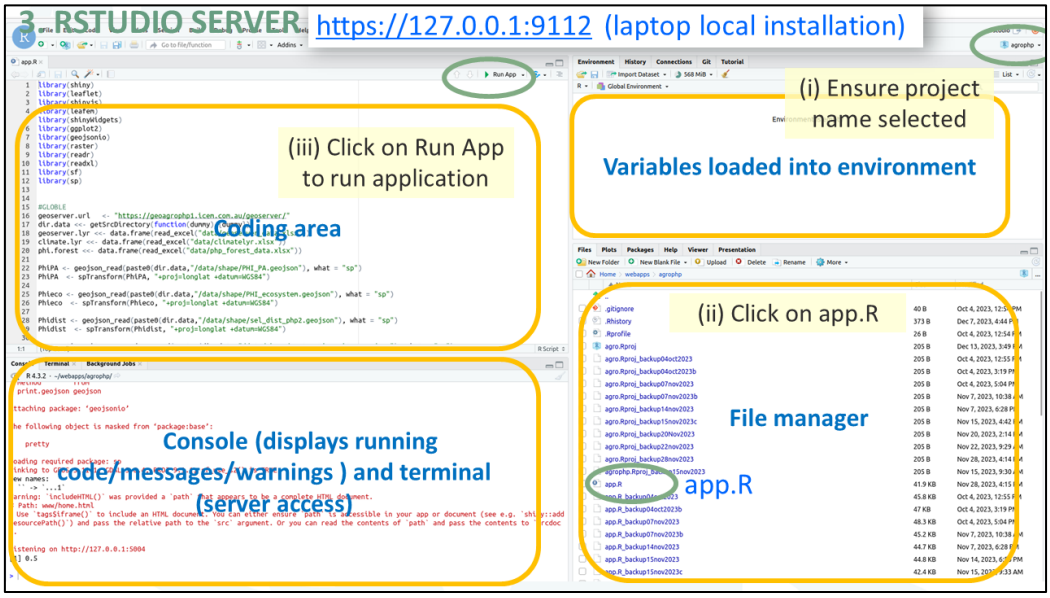
- Coding and app testing tool

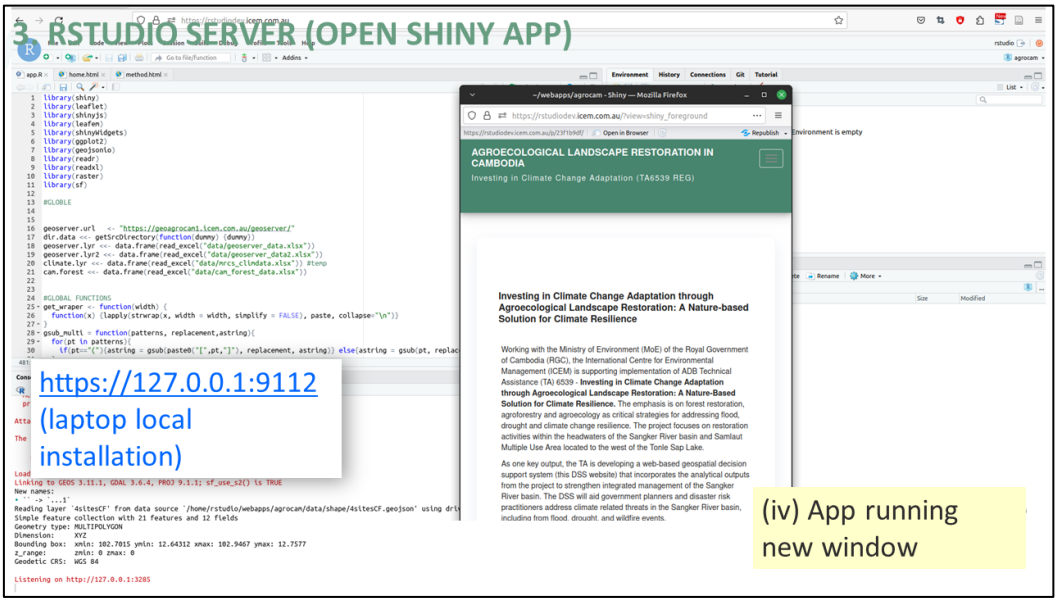
3. RSTUDIO SERVER (LOGIN PAGE)



User name: rstudio
Password: rstudio <- (this can be updated within the docker container)



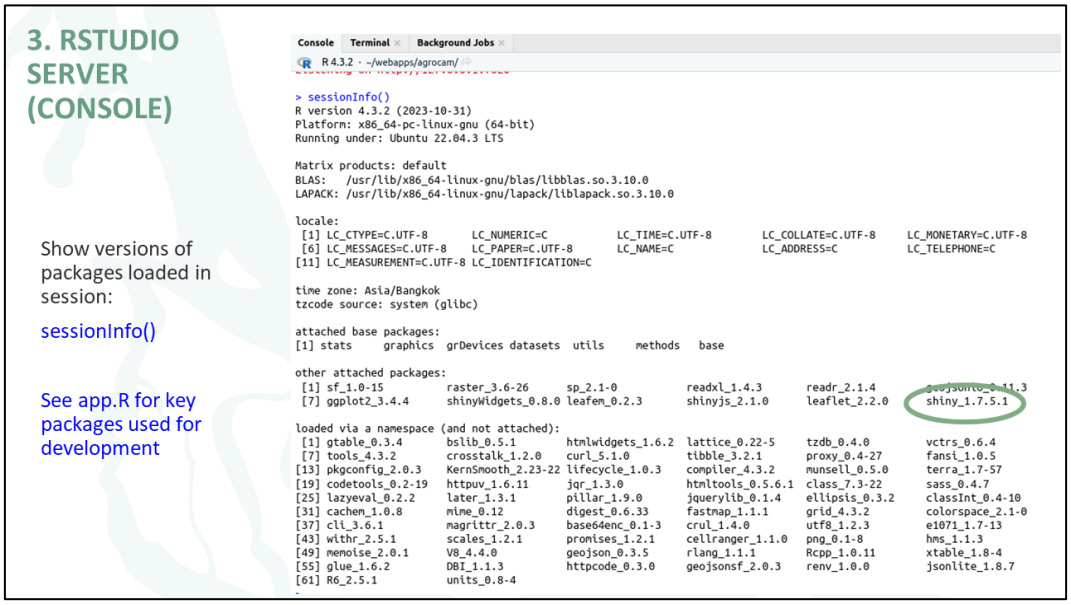


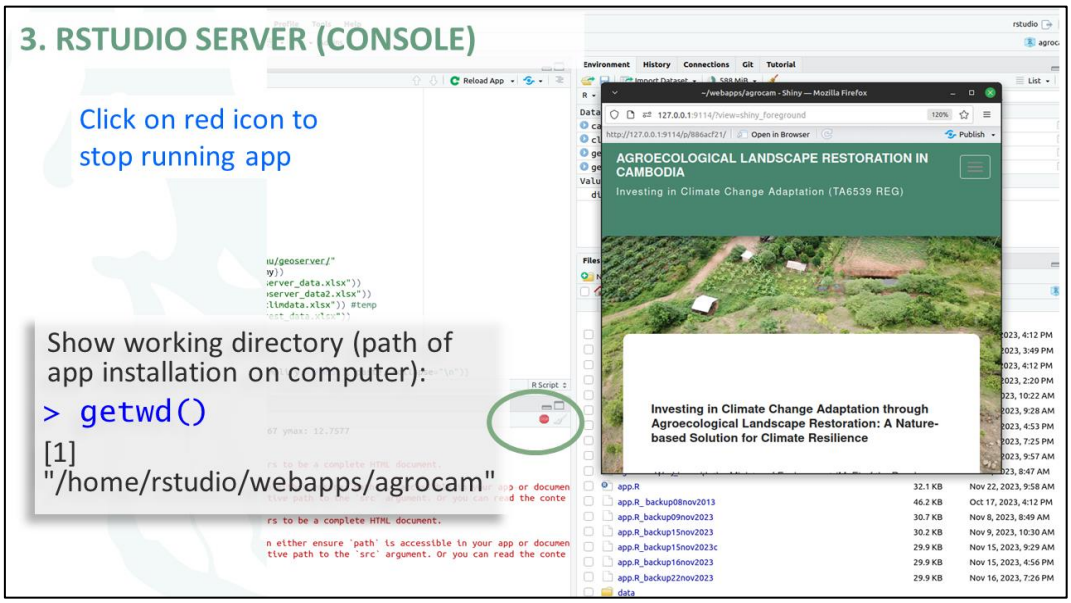


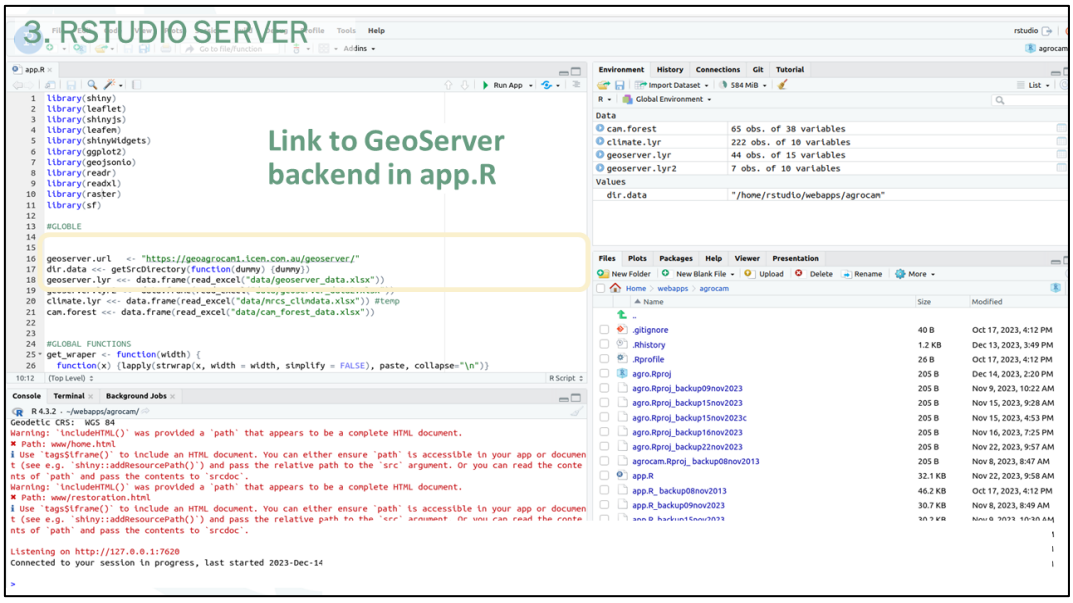
3. RSTUDIO SERVER (CONSOLE)

- Type commands in console tab > return
- E.g., show version of R:

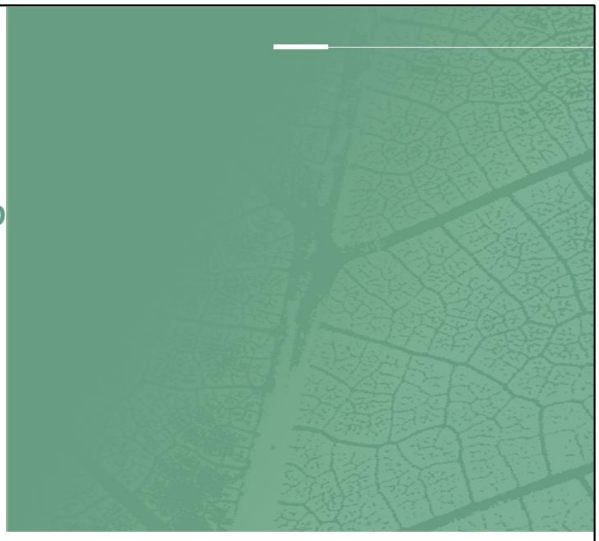
```
> R.Version()
[1] ""R version 4.3.2 (2023-10-31)"
```


3. RSTUDIO SERVER (CONSOLE)
Show versions of packages loaded in session:
sessionInfo()
See app.R for key packages used for development
Console
Terminal
Background Jobs
R 4.3.2 · ~/webapps/agrocam/
> sessionInfo()
R version 4.3.2 (2023-10-31)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 22.04.3 LTS
Matrix products: default
BLAS: /usr/lib/x86_64-linux-gnu/blas/libblas.so.3.10.0
LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.10.0
locale:
[1] LC_CTYPE=C.UTF-8 LC_NUMERIC=C LC_TIME=C.UTF-8 LC_COLLATE=C.UTF-8 LC_MONETARY=C.UTF-8
[6] LC_MESSAGES=C.UTF-8 LC_PAPER=C.UTF-8 LC_NAME=C LC_ADDRESS=C LC_TELEPHONE=C
[11] LC_MEASUREMENT=C.UTF-8 LC_IDENTIFICATION=C
time zone: Asia/Bangkok
tzcode source: system (glibc)
attached base packages:
[1] stats graphics grDevices datasets utils methods base
other attached packages:
[1] sf_1.0-15 raster_3.6-26 sp_2.1-0 readxl_1.4.3 readr_2.1.4
[7] ggplot2_3.4.4 shinyWidgets_0.8.0 leafem_0.2.3 shinyjs_2.1.0 leaflet_2.2.0 shiny_1.7.5.1
loaded via a namespace (and not attached):
[1] gtable_0.3.4 bslib_0.5.1 htmlwidgets_1.6.2 lattice_0.22-5 tzdb_0.4.0 vctrs_0.6.4
[7] tools_4.3.2 crosstalk_1.2.0 curl_5.1.0 tibble_3.2.1 proxy_0.4-27 fansi_1.0.5
[13] pkgconfig_2.0.3 KernSmooth_2.23-22 lifecycle_1.0.3 compiler_4.3.2 munsell_0.5.0 terra_1.7-57
[19] codetools_0.2-19 httpuv_1.6.11 jqr_1.3.0 htmltools_0.5.6.1 class_7.3-22 sass_0.4.7
[25] lazyeval_0.2.2 later_1.3.1 pillar_1.9.0 jquerylib_0.1.4 ellipsis_0.3.2 classInt_0.4-10
[31] cachem_1.0.8 mime_0.12 digest_0.6.33 fastmap_1.1.1 grid_4.3.2 colorspace_2.1-0
[37] cli_3.6.1 magrittr_2.0.3 base64enc_0.1-3 crul_1.4.0 utf8_1.2.3 e1071_1.7-13
[43] withr_2.5.1 scales_1.2.1 promises_1.2.1 cellranger_1.1.0 png_0.1-8 hms_1.1.3
[49] memoise_2.0.1 V8_4.4.0 geojson_0.3.5 rlang_1.1.1 Rcpp_1.0.11 xtable_1.8-4
[55] glue_1.6.2 DBI_1.1.3 httpcode_0.3.0 geojsonsf_2.0.3 renv_1.0.0 jsonlite_1.8.7
[61] R6_2.5.1 units_0.8-4


3. RSTUDIO SERVER (CONSOLE)
Click on red icon to stop running app
Show working directory (path of app installation on computer):
> getwd()
[1] "/home/rstudio/webapps/agrocam"
AGROECOLOGICAL LANDSCAPE RESTORATION IN CAMBODIA
Investing in Climate Change Adaptation (TA6539 REG)
Investing in Climate Change Adaptation through Agroecological Landscape Restoration: A Nature-based Solution for Climate Resilience


3. RSTUDIO SERVER
Link to GeoServer backend in app.R
geoserver.url <- "https://geoagrocam1.icem.com.au/geoserver/"
dir.data <- getSrcDirectory(function(dummy) {dummy})
geoserver.lyr <- data.frame(read_excel("data/geoserver_data.xlsx"))
climate.lyr <- data.frame(read_excel("data/mrcs_clindata.xlsx")) #temp
can.forest <- data.frame(read_excel("data/can_forest_data.xlsx"))
Listening on http://127.0.0.1:7620
Connected to your session in progress, last started 2023-Dec-14

4. OVERVIEW OF R AND RSTUDIO SERVER SOFTWARE SOURCES



19

4. OVERVIEW OF R AND RSTUDIO SERVER SOFTWARE SOURCES

- To show source of R and RStudio Desktop/RStudio Server software

<- Participants don't need to do this as ICEM has preinstalled these software in a Docker container (see Dockerfile) ->

4. INSTALLING R (LINUX)

https://cran.r-project.org/bin/linux/

Install R from Linux distro repository

Example: install of specific version of R in Ubuntu (Jammy: version 22.04):

$ sudo apt-get install r-base=4.3.2

Index of /bin/linux

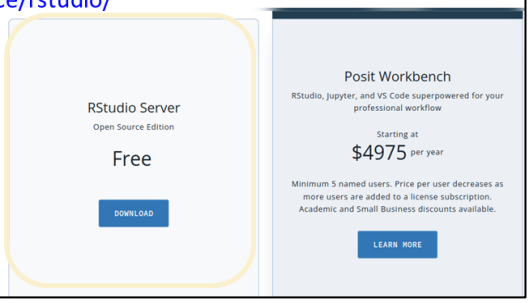
Name	Last modified	Size	Description
Parent Directory		-	
debian/	2023-03-17 23:17	-	
fedora/	2022-06-15 09:55	-	
redhat/	2022-06-15 09:55	-	
suse/	2012-02-16 15:09	-	
ubuntu/	2022-05-24 04:25	-	

Apache Server at cran.r-project.org Port 443

4. INSTALLING RSTUDIO

- RStudio is an integrated development environment (IDE) for **writing code in R**
- RStudio can be used from both desktop (called **RStudio Desktop**) and server (called **RStudio Server**) installations.
- https://posit.co/products/open-source/rstudio/

RStudio Server enables access to RStudio through a web browser



4. INSTALLING RSTUDIO SERVER (FOR ONLINE DEVELOPMENT ONLY)

- Does not work on Windows/macOS (use RStudio Desktop instead)
- Installing particular version (change as required) on Ubuntu:
- E.g.

```
$ wget -c https://download2.rstudio.org/server/jammy/amd64/rstudio-server-2023.03.0-386-amd64.deb && \
  gdebi -n rstudio-server-2023.03.0-386-amd64.deb
```

Reference: https://posit.co/download/rstudio-server/

5. DEPLOYMENT STEPS: RSTUDIO SERVER AND SHINY APP

24

5. HOW TO DEPLOY THE RSTUDIO SERVER DOCKER CONTAINER

Participants don't need to do this, but details included here are for reference on how to save an existing image for deploying to another machine

On ICEM's Hetzner server the following was saved from this Docker image: rstudio:r4.3.2

Example:

$ docker save -o OUTPUT.tar [IMAGE_NAME]

$ gzip < OUTPUT.tar > OUTPUT.tar.gz

Example:

docker save -o img_rstudio_r432_13dec2023.tar rstudio:r4.3.2

gzip < img_rstudio_r432_13dec2023.tar > img_rstudio_r432_13dec2023.tar.gz

5. RSTUDIO/SHINY SERVER CONFIGURATION

Also detailed in Dockerfile:

(A) Check UIDs/GID on host and container match (Linux)
(B) Copy Shiny web application to new host
(C) Load and run Docker image
(D) Run Docker image in DOCKER SWARM MODE
(E) Modify user password in Docker container
(F) Add new project to RStudio Server
(G) Set link to geoserver/POSTGRESQL backends
(H) Access Shiny app in RStudio Server and Shiny Server

(A) CHECK UIDS/GID ON HOST AND CONTAINER MATCH (LINUX)

Key users and group on host and in Docker container:

- **rstudio** <– used to log into RStudio Server r
- **shiny-app** <- rstudio and other users added to this group

- The UIDs/GID must match in both host and container

(A) CHECK UIDS/GID ON HOST AND CONTAINER MATCH (LINUX)

(i) If shiny/rstudio users or shiny-app group DO NOT EXIST on host:

- Add these users to the host with the following IDs:
  $ sudo useradd –m -u 5000 rstudio <- match UID of 5000 in container
  $ sudo useradd –m -u 5001 user2 <- match UID of 5001 in container
  $ id user_name <- check UID of user
- Check UIDs on host
  $ awk -F: '{printf "%s:%s\n",$1,$3}' /etc/passwd <- use to list UIDs on HOST/CONTAINER

$ cat /etc/passwd <- show all users

(A) CHECK UIDS/GID ON HOST AND CONTAINER MATCH (LINUX)

- Add **shiny-app** group
  $ sudo groupadd -g 5020 shiny-app <- match GID in container
- Add users to shiny-app group
  Users added to group to enable access to app folder and contents
  $ sudo usermod -a -G shiny-app rstudio
  $ sudo usermod -a -G shiny-app user2
  - Check GID on host
    $ awk -F: '{printf "%s:%s\n",$1,$3}' /etc/group # use to list GIDs on HOST/CONTAINER

$ cat /etc/group <- show all groups/GIDs and members

(A) CHECK UIDS/GID ON HOST AND CONTAINER MATCH (LINUX)

- **What if host already has an existing user (rstudio/user2), or group (shiny-app) or has allocated the associated UIDs/GID?**
- In Linux, user and group names, and UIDs and GID need to match between host and Docker container as **data is stored on the host machine** to ensure data persistence.
- (i) Modify UID/GIDs in container if required and create a new image using 'docker commit'. <- relatively straightforward
- (ii) Alternatively, build a new Docker image ('docker build') with matching UIDs/GID to those on host <- more complicated, but see 'build' details in Dockerfile

(A) CHECK UIDS/GID ON HOST AND CONTAINER MATCH (LINUX)

(ii) If shiny/rstudio users or shiny-app group ALREADY EXIST on host (create new image with modified UID/GIDs)

(a) Access running container (see 'load' and 'run' commands later) from host to change UIDs/GID:

$ docker exec -u root -it [CONTAINER NAME] bash

(A) CHECK UIDS/GID ON HOST AND CONTAINER MATCH (LINUX)

(b) In container modify UIDs/GID as follows (insert UID/GID number in square brackets):

groupmod -g [new UID] shiny-app

usermod –g [new UID] rstudio

usermod –g [new UID] user2

usermod –u [new UID] –g [new GID] rstudio

usermod –u [new UID] –g [new GID] user2

id user rstudio

(c) In container change file/folder ownership of existing content

find / -uid [old UID] -exec chown -v -h [new UID] '{}' \;

find / -gid [old GID] -exec chgrp -v –h [new GID] '{}' \;

(d) The modified container can then be converted to a new image on the host using 'docker commit':

docker commit -p [CONTAINER NAME] [NEW IMAGE NAME]

e.g., docker commit -p rstudio_r4.3.2 rstudio_r4.3.2_v2

(A) CHECK UIDS/GID ON HOST AND CONTAINER MATCH (LINUX)

(e) Check image is loaded: $ docker images

```
rcooper@richard-XPS:/home/rstudio/webapps/nepalams_dockerfiles$ docker images | grep rstudio
rstudio                          r4.2.2                    2bd96d376528   21 hours ago    3.7GB
```

(f) Check running containers: $ docker ps

```
rcooper@richard-XPS:/home/rstudio/webapps/nepalams_dockerfiles$ docker ps | grep rstudio
96c06b0fdb87   rstudio:r4.2.2              "/bin/sh -c ./start_…"   21 hours ago   Up 21 hours   0.0.0.0:9103->3838/tcp, :::9103->3838/tcp, 0.0.0.0:9102->8787/tcp,
 :::9102->8787/tcp           rstudio_r4.2.2
```

(g) Stop any previously running rstudio docker container to avoid port conflict

$ docker stop rstudio_r4.3.2

(h) Run newly created image with changed UIDs/GID

e.g., $ docker run -d -it -v "/home/rstudio/webapps:/home/rstudio/webapps" --name rstudio_r4.3.2_v2 -p 9112:8787 rstudio:r4.3.2_v2

Also see 'docker run' command below for more details

(B) COPY SHINY WEB APPLICATION TO NEW HOST

Participants don't need to do the following 2 steps as the *tar.gz file can be downloaded

On CURRENT HOST machine with web app to be migrated, in RStudio console run:

> renv::snapshot() <- to create latest list of packages and versions in lock file (renv.lock)

- Create archive of app on CURRENT HOST machine:

In RStudio Server terminal or server's command line terminal:

$ cd /home/rstudio/webapps/[proj_name]

$ sudo tar -czvf [proj_name]_shinyapp_backupDDMMYY.tar.gz . --exclude "*.tar.gz" --exclude "*backup*"

- Check contents of archive:

$ sudo tar -tvf [proj_name]_shinyapp_backupDDMMYY.tar.gz

(B) COPY SHINY WEB APPLICATION TO NEW HOST

- On host (Windows) create path, for example, D:\path\to\rstudio\webapps\[proj_name]

- On host (Linux) create folder path

/home/rstudio/webapps/[proj_name]

- On host extract tar.gz to [proj_name] directory

$ sudo tar -xzvf ~ /[proj_name]_shinyapp_backupDDMMYY.tar.gz -C /home/rstudio/webapps/[proj_name]

Contents after extracting:

e.g., /home/rstudio/webapps/[proj_name]/app.R

/home/rstudio/webapps/[proj_name]/www

- Modify permissions of extracted app (for Linux):

$ cd /home/rstudio/webapps/[proj_name]

[proj_name] $ sudo chown -R rstudio:shiny-app . <- modify group ownership (don't miss '.' at end)

[proj_name] $ sudo chmod -R ug+rwx . <- modify group permissions

[proj_name] $ sudo chmod -R g+s .

(C) LOAD AND RUN DOCKER IMAGE (LINUX)

- Load Docker image

$ cd to directory holding Docker tar.gz file

$ docker load < img_rstudio_r432_DDMMYY.tar.gz

```
rcooper@desktop_rc:~/Downloads/9C9T_ATLAS/rstudio_server/docker_image_tar$ docker load < img_rstudio_r4.0.5-1_17nov2021.tar.gz
9f32931c9d28: Loading layer [==================================================>]  75.27MB/75.27MB
dbf2c0f42a39: Loading layer [==================================================>]  15.36kB/15.36kB
02473afd360b: Loading layer [==================================================>]  3.072kB/3.072kB
44b9aeb542ec: Loading layer [==================================================>]  1.647MB/1.647MB
44abfc850658: Loading layer [==================================================>]  1.644MB/1.644MB
```

- View Docker image is loaded:

$ docker images <- check if loaded

```
rcooper@desktop_rc:~/Downloads/9C9T_ATLAS/rstudio_server/docker_image_tar$ docker images | grep img_rstudio
img_rstudio r4.0.5-1 17nov2021              latest                    0caa12869174   13 days ago     4.18GB
```

(C) LOAD AND RUN DOCKER IMAGE (WINDOWS)

- Load Docker image
  Change to directory holding Docker tar.gz file

```
PS C:\Windows\system32> docker load -i "D:\path\to\imagedownload\img_rstudio_r432_DDMMYY.tar.gz"
```

- View Docker image is loaded:

```
PS C:\Windows\system32> docker images
```

(C) LOAD AND RUN DOCKER IMAGE (LINUX)

- Run Docker image to create running container

  Modify -p ports accordingly

```
$ docker run -d –it -v "/home/rstudio/webapps:/home/rstudio/webapps" --
name rstudio_r432 -p 9112:8787 img_rstudio:r4.3.2
```

add -p 9112:3838 (3838 specified in Rprofile.site)

port 9112: access RStudio Server in browser

Check name of image loaded is correct using 'docker images' command

(C) LOAD AND RUN DOCKER IMAGE (WINDOWS)

- Run Docker image to create running container

  Modify -p ports as noted above.

```
$ docker run -d –it -v "D:\path\to\rstudio\webapps:/home/rstudio/webapps"
rstudio_r432 -p 9112:8787 img_rstudio:r4.3.2
```

Check name of image loaded is correct using 'docker images' command

(C) LOAD AND RUN DOCKER IMAGE

- Check for running Docker container
  $ docker ps

  Example of output:

```
cooper@desktop_rc:~/Downloads/9C9T_ATLAS/rstudio_server/docker_image_tar$ docker ps | grep r4.0.5-1
3ad8f0f714ac   img_rstudio_r4.0.5-1_17nov2021:latest   "/bin/sh -c ./start_…"   2 minutes ago   Up About a minute
 0.0.0.0:4445->3838/tcp, :::4445->3838/tcp, 0.0.0.0:9495->8787/tcp, :::9495->8787/tcp
          rstudio_r4.0.5-1
```

(C) LOAD AND RUN DOCKER IMAGE

- RStudio Server and Shiny Server will start automatically on running the 'docker run' command above
- Check IP of running docker container from HOST terminal:

  `$ docker ps`

  `$ docker inspect <input containername OR ID> | grep "IPAddress"`

  Example:

  `$ docker ps`

  CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

  68c7571ee803 <- container ID

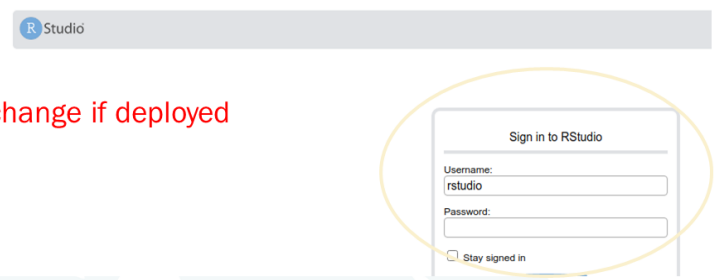
  Check container by ID:

  `$ docker inspect 68c | grep "IPAddress"`<- enter 3+ characters of container ID

  $ "IPAddress": "172.17.0.1"

(C) LOAD AND RUN DOCKER IMAGE

- Go to browser to access RStudio Server and login with rstudio user:
- http://127.0.0.1:9112 <- change port according to run command –p flag



Username: rstudio
Password: rstudio (change if deployed online!)

(D) RUN DOCKER IMAGE IN DOCKER SWARM MODE

The following slides show how to deploy RStudio in Docker swarm mode (same as production mode for Shiny app and GeoServer)

- Copy **docker-compose-rstudio.yml** file to rstudio Shiny app directory (/home/rstudio/webapps/agrocam)
- [OPTIONAL] Copy docker-compose-traefik yml file
  NOTE: This is optional and should have been set up before for other app components.
- Modify the docker-compose-rstudio.yml (see other presentations for guidance)
- Change to directory containing above docker-compose yml file

43

(D) RUN DOCKER IMAGE IN DOCKER SWARM MODE

$ cd /home/rstudio/webapps/agrocam

- On host (Windows) go to for example, D:\path\to\rstudio\webapps\agrocam

$ docker stack deploy -c docker-compose-rstudio.yml rstudio
Make sure traefik service is also running (set up with other apps, see other presentations)

Access RStudio Server in browser **as per URL stated in docker-compose-rstudio.yml file**:
e.g., http://**rstudio**.domainname.topleveldomain

Check service in Docker logs: $ docker service logs rstudio_agrocam_service

44

(D) MODIFY USER PASSWORD IN DOCKER CONTAINER

Change password for rstudio user (this login page is accessible on web):

- Log into running image (container) from HOST MACHINE:

$ docker exec -u root -it <container_name> bash

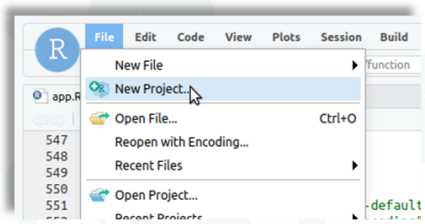
e.g., $ docker exec -u root -it rstudio_r432 bash

- Modify password example:

root@xxxxxxxxx:/home/rstudio# echo rstudio:<insert_passwd> | chpasswd

(E) ADD NEW PROJECT TO RSTUDIO SERVER

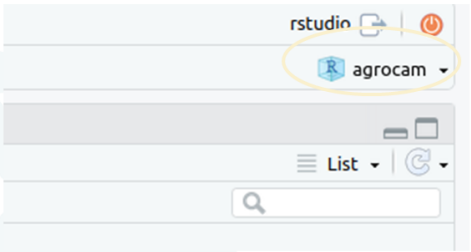
- Log into RStudio Server and create New Project' from existing '.. /project_name directory



select project_name directory as location for installation

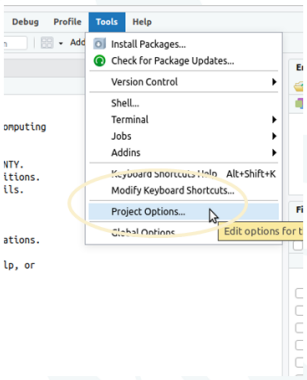
(E) ADD NEW PROJECT TO RSTUDIO SERVER

- In RStudio Server, ensure the project is selected in dropdown menu located to top-right of RStudio Server before running Shiny app, otherwise the app will not run.

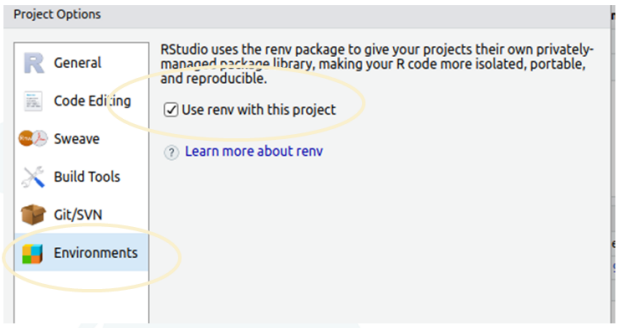


(E) ADD NEW PROJECT TO RSTUDIO SERVER

- Select Project Options:



- Select 'renv' box:



(E) ADD NEW PROJECT TO RSTUDIO SERVER

Restore package libraries (versions included in renv.lock file) :

- In console of RStudio:
  > renv::settings$use.cache(FALSE) <- to ensure local renv repository is used for packages
  > .libPaths() <- check path is to 'renv' directory in [proj_name] directory
  e.g., /home/rstudio/webapps/nepalams/renv/library/R-4.3/x86_64-pc-linux-gnu/

  > renv::restore()
  Select yes to install <- ~15-20+ mins to install

  May need to run the following command to remove warning messages:\
  install.packages("codetools")

(E) ADD NEW PROJECT TO RSTUDIO SERVER

Before running Shiny app:

(i) Clear workspace: click on Session > Clear Workspace

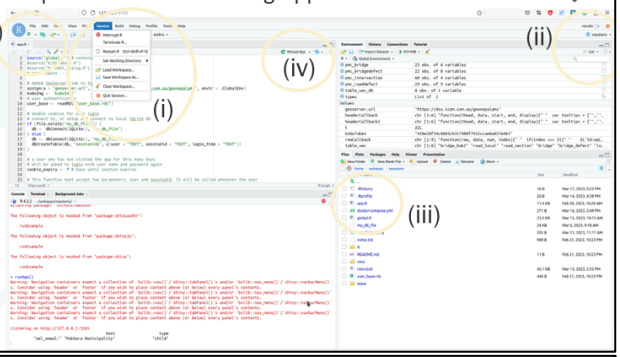
(ii) Ensure project is selected

(iii) Select app.R in the Files tab

(iv) Click on RunApp in RStudio Server to open window containing app

(v) In popup window, click on Open in Browser to view app.

Ensure project is selected



(F) SET LINK TO GEOSERVER/POSTGRESQL BACKENDS

app.R

Reload App

```
4  library(leafem)
5  library(shinyWidgets)
6  library(ggplot2)
7  library(geojsonio)
8  library(readr)
9  library(readxl)
10 library(raster)
11 library(sf)
12
13 #GLOBLE
14
15
16 geoserver.url   <- "https://geoagrocam1.icem.com.au/geoserver/"
17 dir.data <- getSrcDirectory(function(dummy) {dummy})
18 geoserver.lyr <<- data.frame(read_excel("data/geoserver_data.xlsx"))
19 geoserver.lyr2 <<- data.frame(read_excel("data/geoserver_data2.xlsx"))
20 climate.lyr <<- data.frame(read_excel("data/mrcs_climdata.xlsx")) #temp
21 cam.forest <<- data.frame(read_excel("data/cam_forest_data.xlsx"))
22
```

10:12 (Top Level) R Script

Console Terminal Background Jobs

(G) ACCESS TO SHINY APP IN RSTUDIO SERVER AND SHINY SERVER

On laptop:

- http://127.0.0.1:9112 <- RStudio Server login page

On server:

- RStudio Server can be installed on a server and the URLs configured accordingly for accessing the RStudio Server login page
- Reverse proxy 9112 to http using Traefik (see later tutorial)

6. DEPLOYMENT STEPS: SHINY SERVER

- **Optional:** needed only for setting up online development version of Shiny app
- Useful for development but cannot be scaled up for large groups of users (see other presentation for setting up Shiny app in production mode)

53

(A) BACKGROUND NOTES

Participants don't need to do this, but details included here are for reference on how to save an existing image for deploying to another machine

Example:

$ docker save -o OUTPUT.tar [IMAGE_NAME]

$ gzip < OUTPUT.tar > OUTPUT.tar.gz

Example:

docker save -o img_shinyserver_r432_19dec2023.tar shinyserver:r4.3.2

gzip < img_shinyserver_r432_19dec2023.tar > img_shinyserver_r432_19dec2023.tar.gz

(A) DOWNLOAD SHINY SERVER DOCKER IMAGE (AND ASSOCIATED DOCKERFILE)

Download from cloud

Link: https://1drv.ms/f/s!AoHzL3uXbH31hKtXMAXxdFPUz8CUGQ?e=7wPuMP

Password: workshop_sangker

55

(B) CHECK UIDS/GID ON HOST AND CONTAINER MATCH (LINUX)

(i) If shiny user does not exit on host:

- Add shiny user to the host with the following IDs:
  $ sudo shiny -m -u 5001 shiny <- match UID of 5001 in container
  $ id shiny <- check UID of user
- Check UIDs on host
  $ cat /etc/passwd <- show all users

(B) CHECK UIDS/GID ON HOST AND CONTAINER MATCH (LINUX)

- Add shiny user to shiny-app group
  Users added to group to enable access to app folder and contents
  $ sudo usermod -a -G shiny-app shiny
  - Check GID on host
    $ cat /etc/group <- show all groups/GIDs and members

(B) CHECK UIDS/GID ON HOST AND CONTAINER MATCH (LINUX)

- **What if host already has an existing shiny user?**
- In Linux, user and group names, and UIDs and GID need to match between host and Docker container as **data is stored on the host machine** to ensure data persistence.
- (i) Modify UID/GIDs in container if required and create a new image using 'docker commit'. <- relatively straightforward
- (ii) Alternatively, build a new Docker image ('docker build') with matching UIDs/GID to those on host <- more complicated, but see 'build' details in Dockerfile

(B) CHECK UIDS/GID ON HOST AND CONTAINER MATCH (LINUX)

(i) If shiny user already exists on host (create new image with modified UID/GIDs)

(a) Access running container (see 'load' and 'run' commands later) from host to change UIDs/GID:

```
$ docker exec -u root -it [CONTAINER NAME] bash
```

(B) CHECK UIDS/GID ON HOST AND CONTAINER MATCH (LINUX)

(b) In container modify UIDs/GID as follows (insert UID/GID number in square brackets):

```
# usermod –g [new UID] shiny
# usermod –u [new UID] –g [new GID] shiny
# id user shiny
```

(c) In container change file/folder ownership of existing content

```
# find / -uid [old UID] -exec chown -v -h [new UID] '{}' \;
# find / -gid [old GID] -exec chgrp -v –h [new GID] '{}' \;
```

(d) The modified container can then be converted to a new image on the host using 'docker commit':

```
# docker commit -p [CONTAINER NAME] [NEW IMAGE NAME]
e.g., docker commit -p shinyserver_r4.3.2 shinyserver:r4.3.2_v2
```

(C) LOAD AND RUN DOCKER IMAGE (LINUX)

- Load Docker image
  $ cd to directory holding Docker tar.gz file
  $ docker load < img_shinyserver_r432_DDMMYY.tar.gz

- View Docker image is loaded:
  $ docker images <- check if loaded

(C) LOAD AND RUN DOCKER IMAGE (WINDOWS)

- Load Docker image
  Change to directory holding Docker tar.gz file

PS C:\Windows\system32> docker load -i
"D:\path\to\imagedownload\img_shinyserver_r432_DDMMYY.tar.gz"

- View Docker image is loaded:

PS C:\Windows\system32> docker images

(C) LOAD AND RUN DOCKER IMAGE (LINUX)

- Run Docker image to create running container
  Modify -p ports accordingly

$ docker run -d –it -v "/home/rstudio/webapps:/home/rstudio/webapps" –name shinyserver_r432 -p 9113:3838 img_shinyserver:r4.3.2

add -p 9113:3838 (3838 specified in Rprofile.site)

port 9113: access Shiny Server in browser

Check name of image loaded is correct using 'docker images' command

(C) LOAD AND RUN DOCKER IMAGE (WINDOWS)

- Run Docker image to create running container

  Modify -p ports as noted above.

$ docker run -d –it -v "D:\path\to\rstudio\webapps:/home/rstudio/webapps" shinyserver_r432 -p 9113:3838 img_shinyserver:r4.3.2

Check name of image loaded is correct using 'docker images' command

(C) LOAD AND RUN DOCKER IMAGE

- Check for running Docker container

$ docker ps

Example of output:

```
cooper@desktop_rc:~/Downloads/9C9T_ATLAS/rstudio_server/docker_image_tar$ docker ps | grep r4.0.5-1
3ad8f0f714ac   img_rstudio_r4.0.5-1_17nov2021:latest   "/bin/sh -c ./start_…"   2 minutes ago   Up About a minute
 0.0.0.0:4445->3838/tcp, :::4445->3838/tcp, 0.0.0.0:9495->8787/tcp, :::9495->8787/tcp
            rstudio_r4.0.5-1
```

(C) LOAD AND RUN DOCKER IMAGE

- Go to browser to access Shiny Server :
- http://127.0.0.1:9113 <- change port according to run command –p flag

(C) LOAD AND RUN DOCKER IMAGE

- Go to browser to access Shiny Server :
- http://127.0.0.1:9113 <- change port according to run command -p flag

- If error, check permissions in container/host of /home/rstudio directory

```
docker exec -it -u root [SHINYSERVER_CONTAINER]
root@f9c01ca7358f:/home/shiny# cd /home/rstudio
root@f9c01ca7358f:/home/rstudio# chown rstudio:shiny-app .
```

(C) LOAD AND RUN DOCKER IMAGE

- RStudio Server and Shiny Server will start automatically on running the 'docker run' command above
- Check IP of running docker container from HOST terminal:

$ docker ps

$ docker inspect <input containername OR ID> | grep "IPAddress"

Example:

$ docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

68c7571ee803 <- container ID

Check container by ID:

$ docker inspect **68c** | grep "IPAddress"<- enter 3+ characters of container ID

$ "IPAddress": "172.17.0.1"

(D) RUN DOCKER IMAGE (DOCKER SWARM MODE)

The following slides show how to deploy Shiny Server in Docker swarm mode (same as production mode for Shiny app and GeoServer)

- Copy **docker-compose-rstudio.yml** file to rstudio Shiny app directory (/home/rstudio/webapps/agrocam)
- [OPTIONAL] Copy docker-compose-traefik yml file
  NOTE: This is optional and should have been set up before for other app components.
- Modify the docker-compose-shiny-server.yml (see other presentations for guidance)
- Change to directory containing above docker-compose yml file

69

(D) RUN DOCKER IMAGE (DOCKER SWARM MODE)

$ cd /home/rstudio/webapps/agrocam

- On host (Windows) go to for example, D:\path\to\rstudio\webapps\agrocam

$ docker stack deploy -c docker-compose-shiny-server.yml rstudio

Make sure traefik service is also running (set up with other apps, see other presentations)

Access RStudio Server in browser **as per URL stated in docker-compose-rstudio.yml file**:
e.g., http://**shinyserver**.domainname.topleveldomain

Check service in Docker logs: $ docker service logs shinyserver_agrocam_service

70

NOTES

If permissions error on running docker container:

Permissions on rstudio directory IN DOCKER CONTAINER should be: rstudio:shiny-app or permission error in browser. This needs to be done after deploying container from image

Access container (or host) and run chown command:

root@4f3b3bf55fb9:/home/rstudio# chown rstudio:shiny-app .

NOTES

If internet connection in the Docker container is lost, then restart the Docker process:

$ wget google.com <- Check internet connection using the terminal within RStudio's interface.

Restart Docker process

$ sudo systemctl restart docker

Remember to restart any other previously running containers as service restart will stop all running containers on computer (add to cron job for automatic restart)

7. Q&A SESSION

73

Thank you!

Annex V: Traefik Configuration

Record specific configuration details (TBD after deployment to server)

Annex VI: Post-development Configuration

Record specific configuration details (TBD after deployment to server)

Aerial view of agricultural farms in Dontret, Cambodia. Drone photo taken in the 3rd field mission to pilot farms (photo by ICEM).

Correspondence
26/86, To Ngoc Van Street,
Tay Ho District, Hanoi, Vietnam
(t) +84 24 3823 9127
(f) +84 24 3719 0367
info@icem.com.au
www.icem.com.au