RAD Service

RADS

Basic System Administration

https://github.com/radservice/rad-community/fork

Last Revised on Oct 2021

Prerequisites

1. Basic knowledge on how to create RADS App.
2. Proficient in server administration (i.e. Application server, Database Server, Networking).

Content

1. Typical stack for RADS
2. Basic Database Management (MariaDB)
3. Basic Application Server Management (Tomcat)
4. Web Log Viewer
5. Application Performance Monitoring (APM)

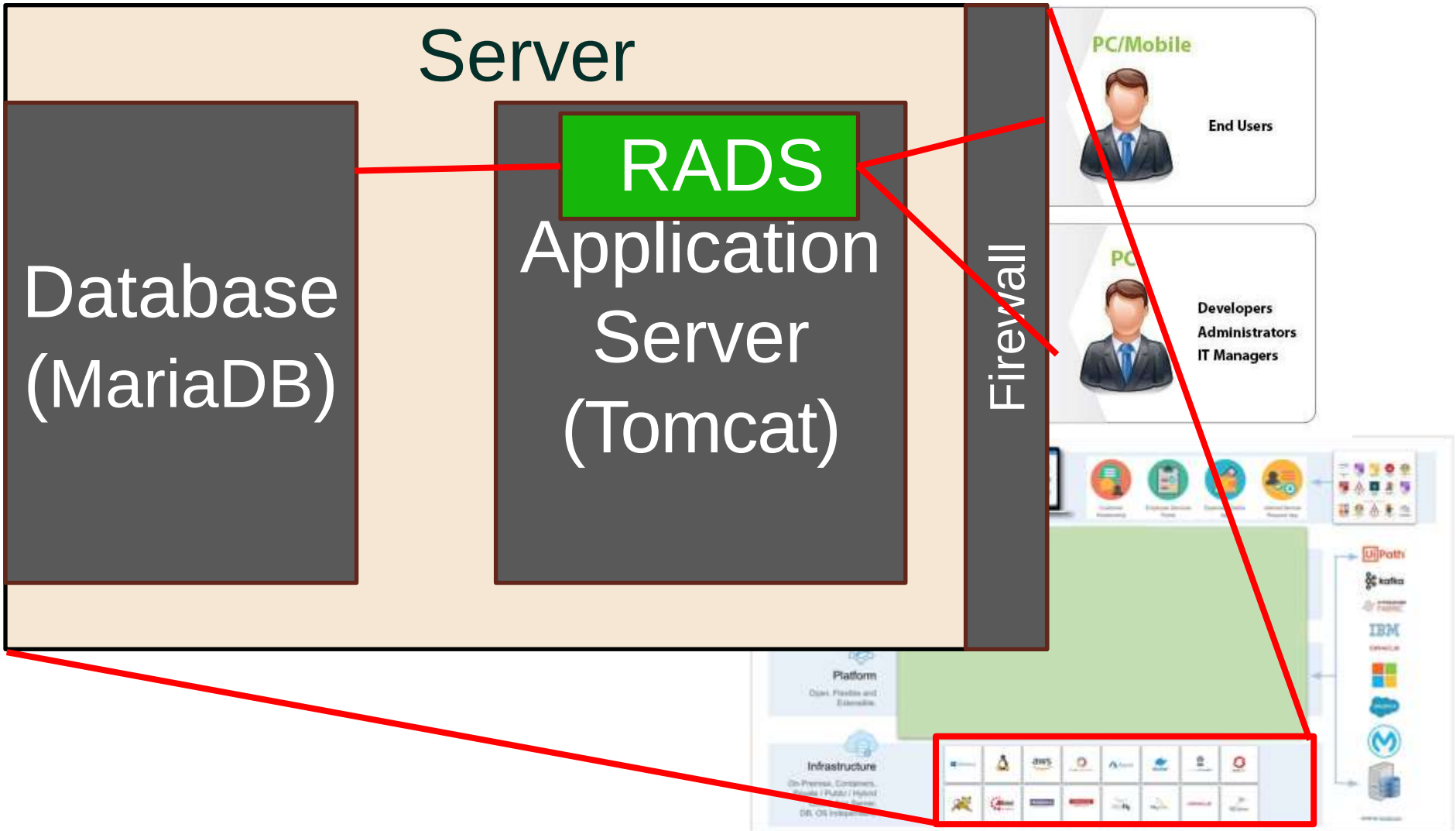
Chapter 1

Typical Stack for RADS

Typical Platform Requirement

Apache Tomcat	MariaDB	JDK

The Typical Stack



Chapter Review

- General understanding on how RADS is hosted.

Chapter 2

Basic Database Management (MariaDB)

Typical Platform Requirement

Apache Tomcat	MariaDB	JDK

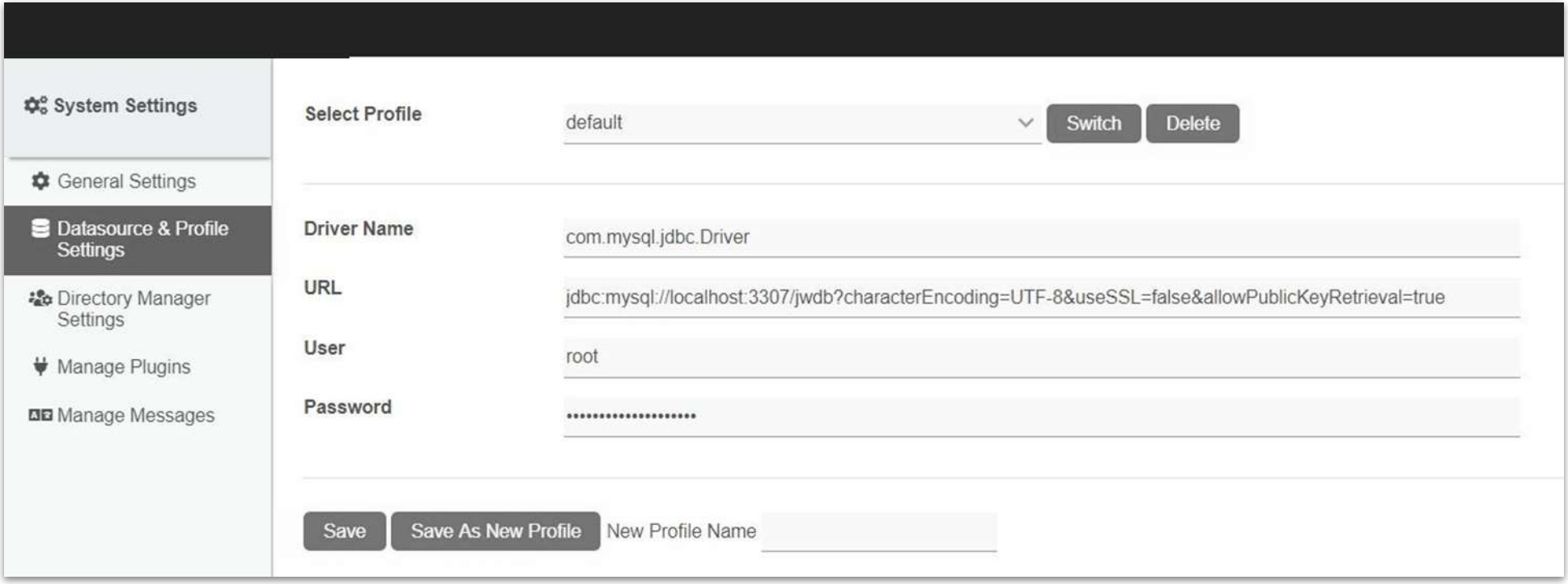
We are going to inspect the Datasource

Datasource Profile

- RADS supports configuration of ***multiple** datasource profiles, but only **1** profile can be activated at any point of time.

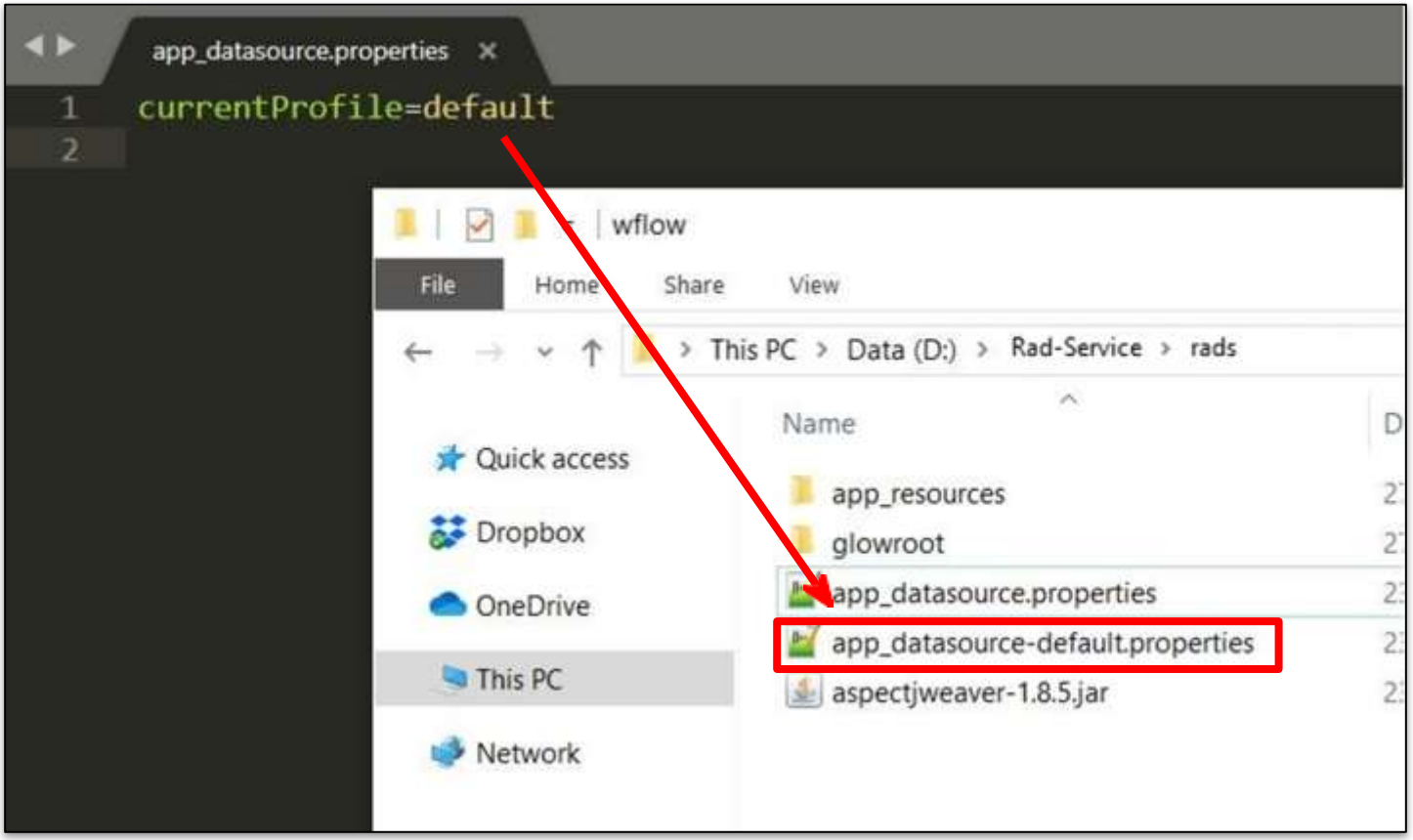
Datasource Profile Configuration

- The Datasource Profile can be located in **Settings -> Datasource & Profile.**



Datasource Profile Configuration in File System

- The Datasource Profile can also be located in the **wflow** folder.



Optional Exercise - Setting Up New Database

- Assuming that our current installation of RADS is connected to the MariaDB database named "`radsdb`"
- We are going to:
  1. Create a new database named "`wflowdb`"
  2. Create a new datasource profile that uses "`wflowdb`"
  3. Switch RADS's datasource profile to use "`wflowdb`"
- RADS will automatically initialize any new empty databases upon setup (no datasource profile found).

1. Setup Empty Wflowdb

- Assuming MariaDB is installed in **C:\RADS\mariadb.**
- Run command prompt

```
cd C:\RADS\mariadb\bin
mysql -u root
mysql> create database wflowdb;
Query OK, 1 row affected (0.05 sec)
mysql> exit;
Bye
mysql -u root wflowdb < C:\RADS\data\radsdb-empty.sql
```

Import to this database

Location of the SQL import file

2. Verify Creation of W flowdb

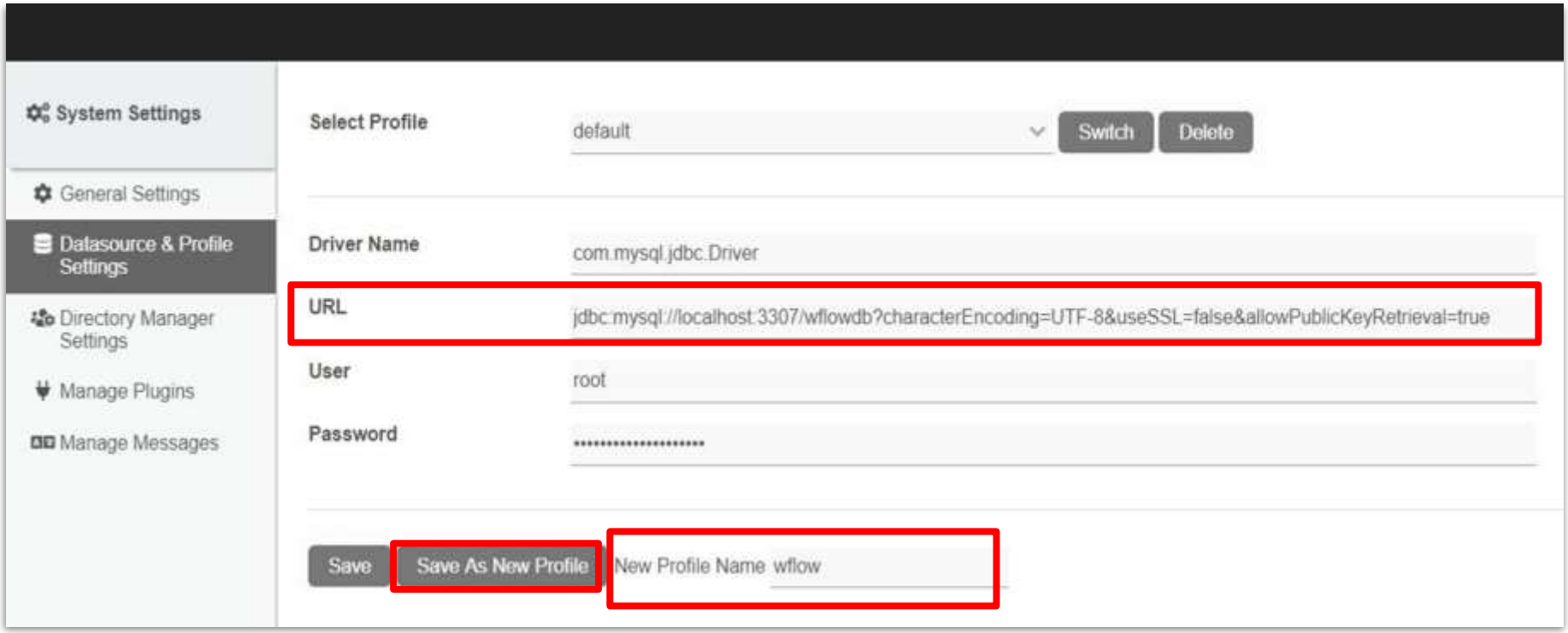
- Verify existence of new database.

```
mysql -u root
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| radsdb             |
| wflowdb            |
| performance_schema |
+--------------------+
5 rows in set (0.00 sec)
mysql> exit;
Bye
```

3. Create A New Datasource Profile

- Login to **RADS** as Admin user.
- Navigate to **System Settings > Datasource & Profile Setting.**
- Amend the URL field and change “radsdb” to “wflowdb” (without quotation marks).
- Specify a new profile name – “wflow”.
- Click on the “Save As New Profile” button.

3. Create A New Datasource Profile



4. Compare The Changes in File System

- The value of **currentProfile** in `/rads/app_datasource.properties` is changed to **wflow.**
- A new file, `/rads/app_datasource-wflow.properties` is created, with configurations to **wflow** datasource.

Database User / Password Changed?

- Open to edit `/rads/app_datasource-`*`profileName`*`.properties` file in text editor.
- To update database username, amend the values for **`workflowUser`**.
- To update database password, amend the values of **`workflowPassword`**.
- Restart RADS server (or Apache Tomcat) for changes to take effect.

Discussion

- Can we use other Application Server container other than Tomcat? (e.g: JBoss, Glassfish, etc...)
- Can we use other Database Systems other than MariaDB?
- Must we use RDBMS?
- How does RADS talk to MariaDB?
- How do users access RADS?
- Can we separate Application Server and Database Server?

Chapter Review

- General understanding on how RADS connects to the Database system.

Chapter 3

Basic Application Server Management (Tomcat)

Typical Platform Requirement

Apache Tomcat	MariaDB	JDK

We are going to inspect Tomcat

RADS Application Files

- In Apache Tomcat, there are 1 RADS web application file to be noted:
  - {Tomcat}/webapps/**rad**.war

Updating RADS

- Essentially, just replace the **.war** files with the latest available version.
- You can update your RADS by following these steps:
  1. Stop RADS.
  2. Delete "**rad**" folder and "**rad.war**" in "[RADS Installation Directory]/[tomcat]/webapps".
  3. Delete "**rad**" folder in "[RADS Installation Directory]/[tomcat]/work/Catalina/localhost".
  4. Place your newly downloaded "**rad.war**" in "[RADS Installation Directory]/[tomcat]/webapps".
  5. Start RADS.

wflow.home Directory

- The location where the following RADS physical files are stored:
  - Datasource and profile configurations.
  - XML definition and HTML rendering of forms designed using Form Builder.
  - Plugins.
  - Graphical image and thumbnail of all processes.
  - Uploaded Files Attachments.

Where is wflow.home Directory?

- Open **RADS-start.bat** using text editor, and look for the **JAVA_OPTS** parameters.
- wflow.home is configured using the **-Dwflow.home** option.
- default **rads** folder location is in the root of RADS folder.

```
REM Start Tomcat
set JAVA_HOME=.\jre11.0.2
set CATALINA_HOME=.\apache-tomcat-8.5.41
set JAVA_OPTS=-Xmx768M -Dwflow.home=./rads/ -javaagent:./rads/aspectjweaver-1.8.5.jar
REM set JAVA_OPTS=-XX:MaxPermSize=128m -Xmx1024M -Xdebug -Xnoagent -Djava.compiler=NONE
-javaagent:./rads/aspectjweaver-1.8.5.jar -javaagent:./rads/glowroot/glowroot.jar
ECHO == Starting Tomcat from %CATALINA_HOME% ==
ECHO.
%CATALINA_HOME%\bin\catalina.bat run
```

Backup

- To backup RADS, you should include:
  - Use database dump as a snapshot:
    - `mysqldump -u root wflowdb > backup.sql`
  - Backup entire `wflow.home` directory.

Log Files

- When RADS is running in console, all log messages are shown on the console window.
- Log messages from **Apache Tomcat** are written to {Tomcat}/logs/**catalina**.yyyy-MM-dd.log.
- Log messages from **RADS** are written to {Tomcat}/logs/**localhost**.yyyy-MM-dd.log.
- To customize the logging behaviors, edit {Tomcat}/conf/**logging**.properties file.

What is a Stack Trace?

```
ERROR 02 Sep 2010 10:19:51 org.hibernate.util.JDBCExceptionReporter  - Cannot create PoolableConnectionFactory (Access denied for user 'roo'@'localhos
ERROR 02 Sep 2010 10:19:51 org.springframework.web.context.ContextLoader  - Context initialization failed
org.springframework.beans.factory.BeanCreationException: Error creating bean with name 'setupSessionFactory' defined in class path resource [commonsAp
        at org.springframework.beans.factory.support.AbstractAutowireCapableBeanFactory.initializeBean(AbstractAutowireCapableBeanFactory.java:1336)
        at org.springframework.beans.factory.support.AbstractAutowireCapableBeanFactory.doCreateBean(AbstractAutowireCapableBeanFactory.java:471)
        at org.springframework.beans.factory.support.AbstractAutowireCapableBeanFactory$1.run(AbstractAutowireCapableBeanFactory.java:409)
        at java.security.AccessController.doPrivileged(Native Method)
        at org.springframework.beans.factory.support.AbstractAutowireCapableBeanFactory.createBean(AbstractAutowireCapableBeanFactory.java:380)
        at org.springframework.beans.factory.support.AbstractBeanFactory$1.getObject(AbstractBeanFactory.java:264)
        at org.springframework.beans.factory.support.DefaultSingletonBeanRegistry.getSingleton(DefaultSingletonBeanRegistry.java:220)
        at org.springframework.beans.factory.support.AbstractBeanFactory.doGetBean(AbstractBeanFactory.java:261)
        at org.springframework.beans.factory.support.AbstractBeanFactory.getBean(AbstractBeanFactory.java:185)
        at org.springframework.beans.factory.support.AbstractBeanFactory.getBean(AbstractBeanFactory.java:164)
        at org.springframework.beans.factory.support.DefaultListableBeanFactory.preInstantiateSingletons(DefaultListableBeanFactory.java:423)
        at org.springframework.context.support.AbstractApplicationContext.finishBeanFactoryInitialization(AbstractApplicationContext.java:729)
        at org.springframework.context.support.AbstractApplicationContext.refresh(AbstractApplicationContext.java:381)
        at org.springframework.web.context.ContextLoader.createWebApplicationContext(ContextLoader.java:255)
        at org.springframework.web.context.ContextLoader.initWebApplicationContext(ContextLoader.java:199)
        at org.springframework.web.context.ContextLoaderListener.contextInitialized(ContextLoaderListener.java:45)
        at org.apache.catalina.core.StandardContext.listenerStart(StandardContext.java:3843)
        at org.apache.catalina.core.StandardContext.start(StandardContext.java:4342)
        at org.apache.catalina.core.ContainerBase.addChildInternal(ContainerBase.java:791)
        at org.apache.catalina.core.ContainerBase.addChild(ContainerBase.java:771)
        at org.apache.catalina.core.StandardHost.addChild(StandardHost.java:525)
        at org.apache.catalina.startup.HostConfig.deployDescriptor(HostConfig.java:627)
        at org.apache.catalina.startup.HostConfig.deployDescriptors(HostConfig.java:553)
        at org.apache.catalina.startup.HostConfig.deployApps(HostConfig.java:488)
        at org.apache.catalina.startup.HostConfig.start(HostConfig.java:1149)
        at org.apache.catalina.startup.HostConfig.lifecycleEvent(HostConfig.java:311)
        at org.apache.catalina.util.LifecycleSupport.fireLifecycleEvent(LifecycleSupport.java:117)
        at org.apache.catalina.core.ContainerBase.start(ContainerBase.java:1053)
        at org.apache.catalina.core.StandardHost.start(StandardHost.java:719)
        at org.apache.catalina.core.ContainerBase.start(ContainerBase.java:1045)
```

What is a Stack Trace?

- When an error is thrown, a stack trace will depict a sequence of events executed in the code level, which can precisely suggest the point where an exception is caught.
- Stack trace could also suggest meaningful error message to help troubleshooting.
- In the example above, indicative error message is shown before stack trace:

```
ERROR 02 Sep 2019 10:19:51 org.hibernate.util.JDBCExceptionReporter  -
Cannot create PoolableConnectionFactory (Access denied for user
'roo'@'localhost' (using password: NO))
```

Troubleshooting

- When you're seeking for help on troubleshooting, copy the whole stack trace (**all log messages printed at the same date time**) and share it out. This can help the troubleshooter to have a better idea on the error.

Memory Allocation

- Open **RADS-start.bat** using text editor, and look for the **JAVA_OPTS** parameters.
- Memory allocation is configured using the **-Xmx** option.

```
9   REM Start Tomcat
10  set JAVA_HOME=.\jre11.0.2
11  set CATALINA_HOME=.\apache-tomcat-8.5.41
12  set JAVA_OPTS=-Xmx768M -Dwflow.home=./rads/ -javaagent:./rads/aspectjweaver-1.8.5.jar
13  REM set JAVA_OPTS=-XX:MaxPermSize=128m -Xmx1024M -Xdebug -Xnoagent -Djava.compiler=NONE
    -javaagent:./rads/aspectjweaver-1.8.5.jar -javaagent:./rads/glowroot/glowroot.jar
14  ECHO == Starting Tomcat from %CATALINA_HOME% ==
15  ECHO.
16  %CATALINA_HOME%\bin\catalina.bat run
```

Optimize Tomcat

- Edit {Tomcat}/conf/**server.xml** using text editor.
- Look for the **HTTP/1.1** connector configuration:
- Try to add MaxThreads="" to set your preferred maximum thread count.

```
<Connector port="8080" protocol="HTTP/1.1"
               connectionTimeout="20000"
               maxThreads="2000"
               compression="on"
               useSendfile="false"
               redirectPort="8443" />
```

- NOTE: One size does not fits all. Every environment need to be fine tuned accordingly. Read more at

https://docs.rads.purwana.net/RADS+Clustering+and+Performance+Testing+on+AWS

Changing Apache Tomcat HTTP Port

- Edit {Tomcat}/conf/**server.xml** using text editor.
- Look for the **HTTP/1.1** connector configuration:

```
<Connector port="8080" protocol="HTTP/1.1"
```

- Change the port number to your preferred one, and restart RADS Server (Apache Tomcat).

Optional Exercise - Setting up SSL

- Assuming that you are running RADS with the default bundled Tomcat.
- We are going to:-
  1. Generate a key store file.
  2. Configure Tomcat to support SSL.

Reference:
https://docs.rads.purwana.net/Setting+Up+SSL+on+Tomcat

1. Generate a key store file

- First of all, we will need to generate a key store file. You may want to generate it with or without a SSL certificate purchased from your SSL certificate provider. This is an example on generating one by ourselves.

1. Generate a key store file

```
C:\Program Files\Java\jdk\bin>keytool -genkey -alias tomcat -keyalg RSA
Enter keystore password: password
Re-enter new password: password
What is your first and last name?
  [Unknown]:  Robert
What is the name of your organizational unit?
  [Unknown]:  home
What is the name of your organization?
  [Unknown]:  home
What is the name of your City or Locality?
  [Unknown]:  SF
What is the name of your State or Province?
  [Unknown]:  CA
What is the two-letter country code for this unit?
  [Unknown]:  US
Is CN=Robert, OU=home, O=home, L=SF, ST=CA, C=US correct?
  [no]:  yes
Enter key password for <tomcat>
        (RETURN if same as keystore password): password
Re-enter new password: password
```

2. Configure Tomcat to support SSL

- Make sure that your server is not running. Open up **\apache-tomcat\conf\server.xml**, uncomment and edit the following lines accordingly.

2. Configure Tomcat to support SSL

- **Port**: 8443 to 443 (If you intend to browse to https://yourDomain instead of https://yourDomain:8443)

  keystoreFile: Path to the .keystore file

  keystorePass: The password defined earlier

```
<!-- Define a SSL HTTP/1.1 Connector on port 8443
         This connector uses the JSSE configuration, when using APR,
the
         connector should be using the OpenSSL style configuration
         described in the APR documentation -->

    <Connector port="443" protocol="HTTP/1.1" SSLEnabled="true"
               maxThreads="150" scheme="https" secure="true"
               clientAuth="false" sslProtocol="TLS"
              keystoreFile="C:/Users/Robert/.keystore"
              keystorePass="password" />
```

2. Configure Tomcat to support SSL

- Start your server.
- You may now surf to your RADS at https://yourDomain/rad or https://yourDomain:8443/rad, depending on your configuration.

Chapter Review

- General understanding on how RADS is “hosted” runs under application server – Tomcat.

Chapter 4

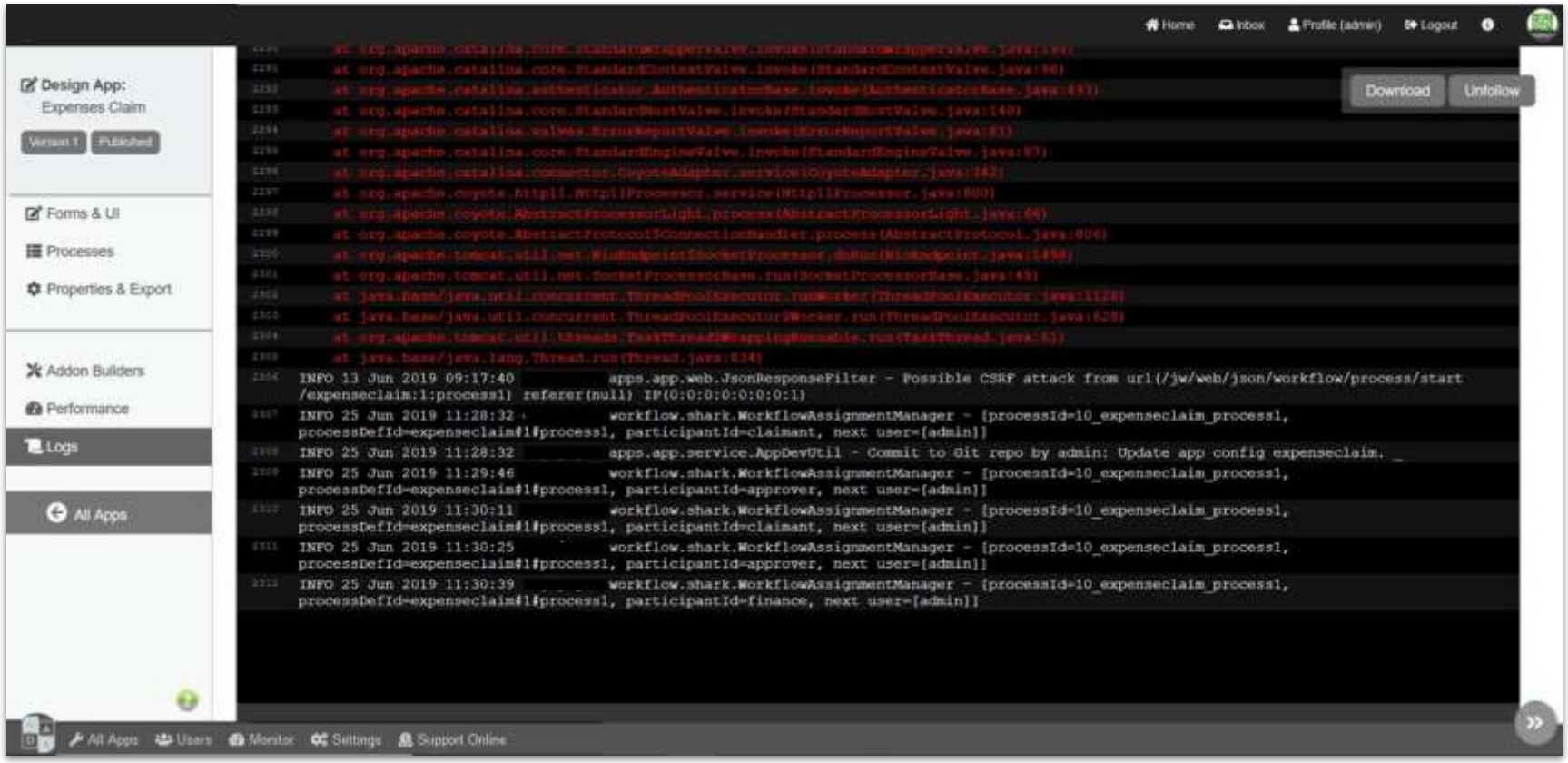
Web Log Viewer

Accessing Log using Web Log Viewer

- New RADS Feature.
- Web App Log Viewer enables the administrators to view the logs on the web console for viewing and finding errors.

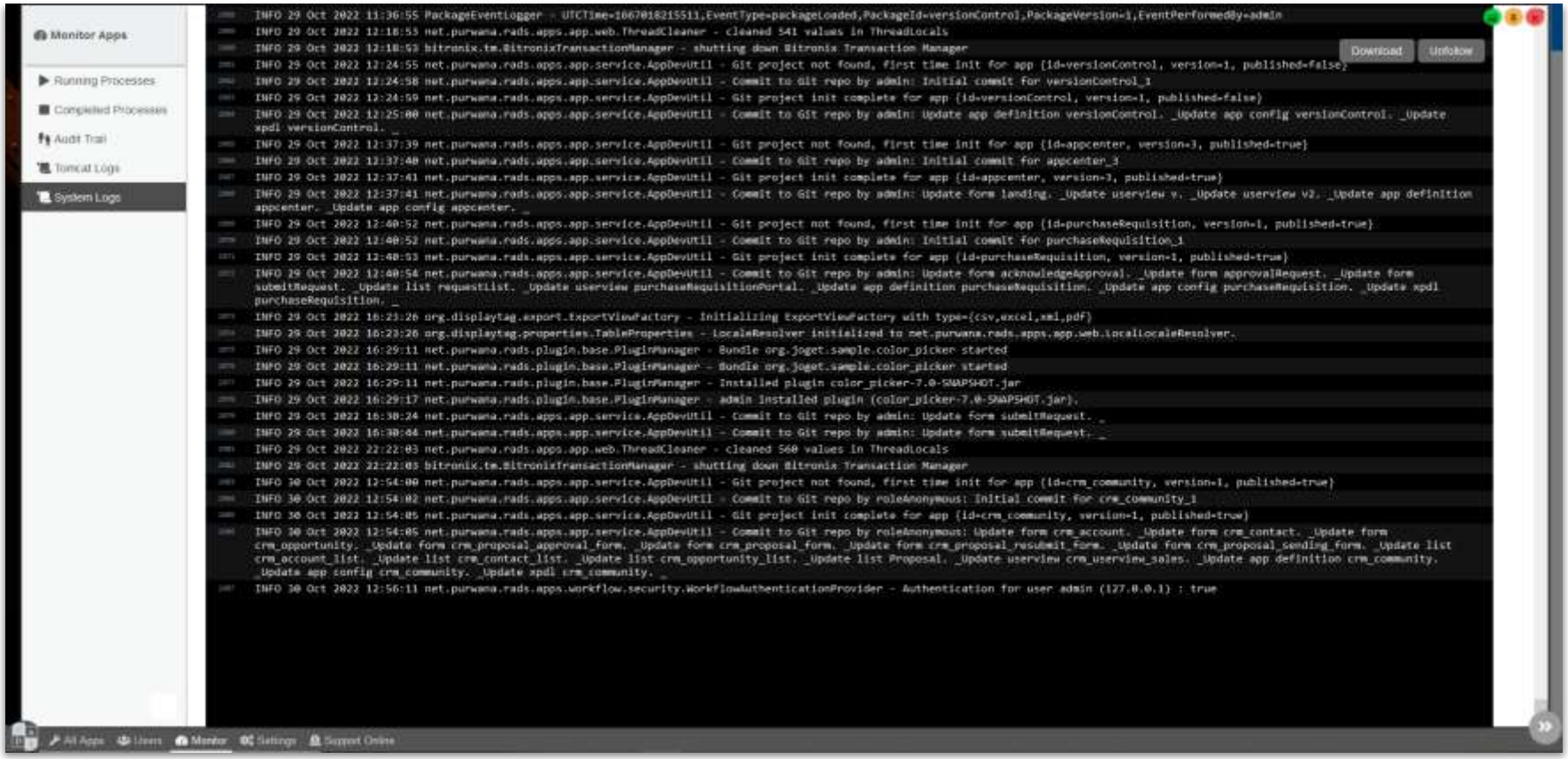
App specific Log

- Displays log information and errors related to the specific App.



System Logs

- Displays system-wide log information and errors inclusive of all Apps.
- Admin Bar > Monitor > System Logs.



What can you do with the Web App Log Viewer?

- Download the log file by clicking the **Download** button.
- Click **Follow** button to display newly added lines from a log file in real-time on the web console.

Chapter Review

- General understanding on how to access log via Web Log Viewer.

Chapter 5

Application Performance Monitoring (APM)

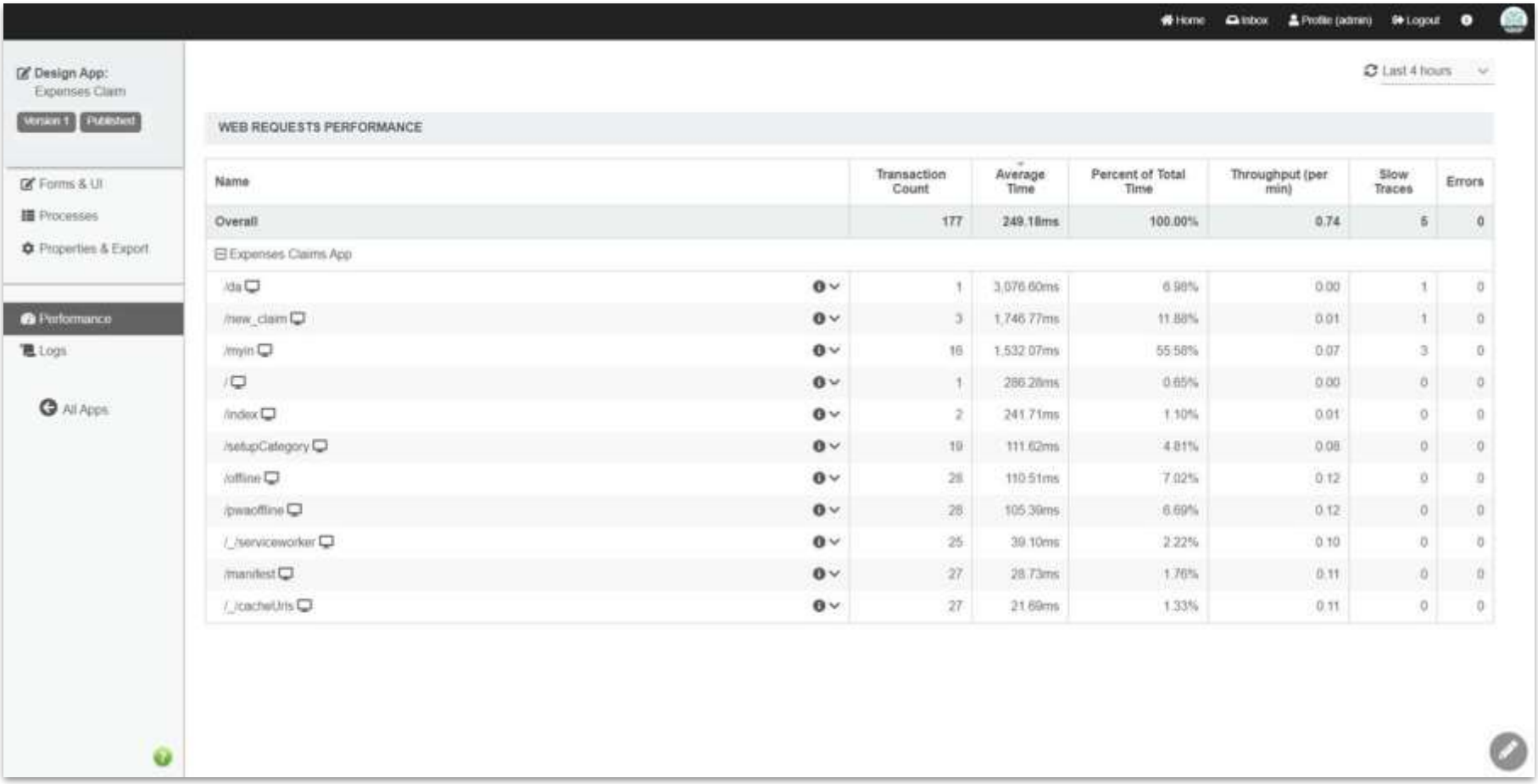
What is APM?

- New DX Feature.
- Built-in feature that automatically monitors system and application performance during runtime.
- Alert notification can be configured when user-defined threshold have exceeded based on various metrics.

Reference: https://docs.rads.purwana.net/Application+Performance+Management

App-level Performance Monitoring

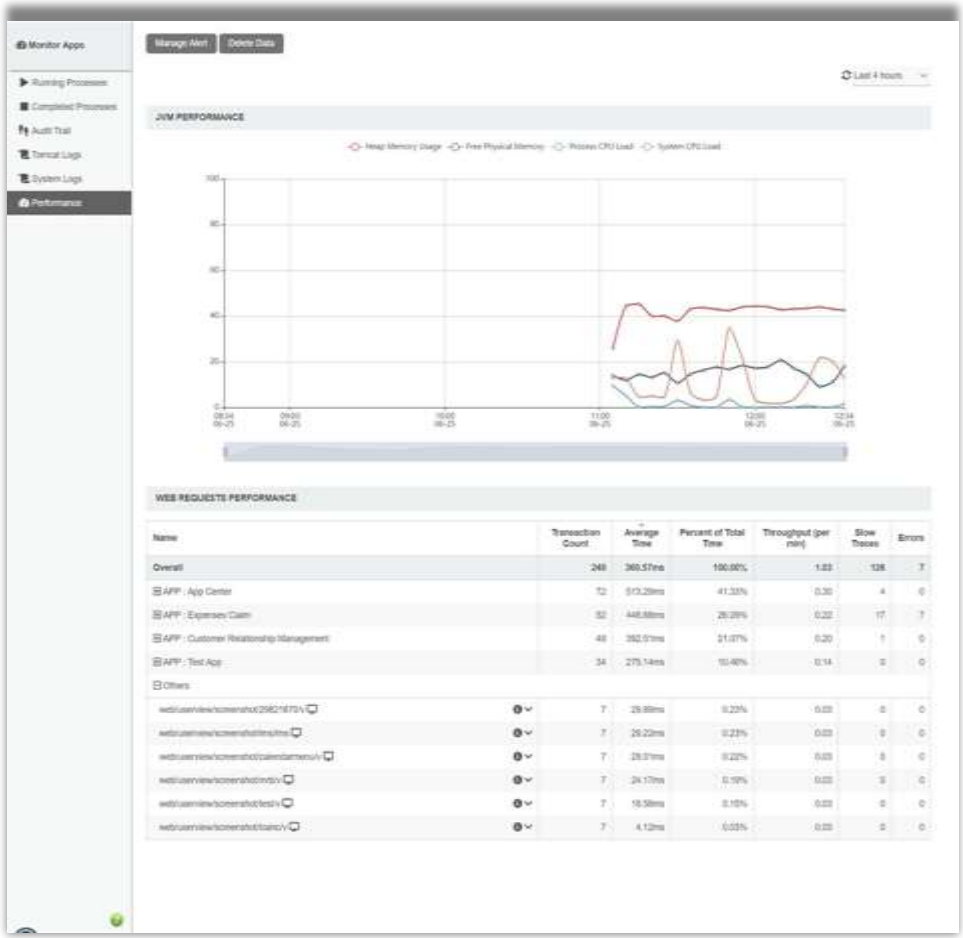
- Can be accessed by selecting the respective App > Design App > Performance.



Name	Transaction Count	Average Time	Percent of Total Time	Throughput (per min)	Slow Traces	Errors
Overall	177	249.18ms	100.00%	0.74	5	0
Expenses Claims App						
/da	1	3,076.60ms	6.98%	0.00	1	0
/new_claim	3	1,746.77ms	11.88%	0.01	1	0
/myin	16	1,532.07ms	55.58%	0.07	3	0
/	1	286.28ms	0.65%	0.00	0	0
/index	2	241.71ms	1.10%	0.01	0	0
/setupCategory	19	111.62ms	4.81%	0.08	0	0
/offline	28	110.51ms	7.02%	0.12	0	0
/pwaoffline	28	105.39ms	6.69%	0.12	0	0
/_/serviceworker	25	39.10ms	2.22%	0.10	0	0
/manifest	27	28.73ms	1.76%	0.11	0	0
/_/cacheUrls	27	21.69ms	1.33%	0.11	0	0

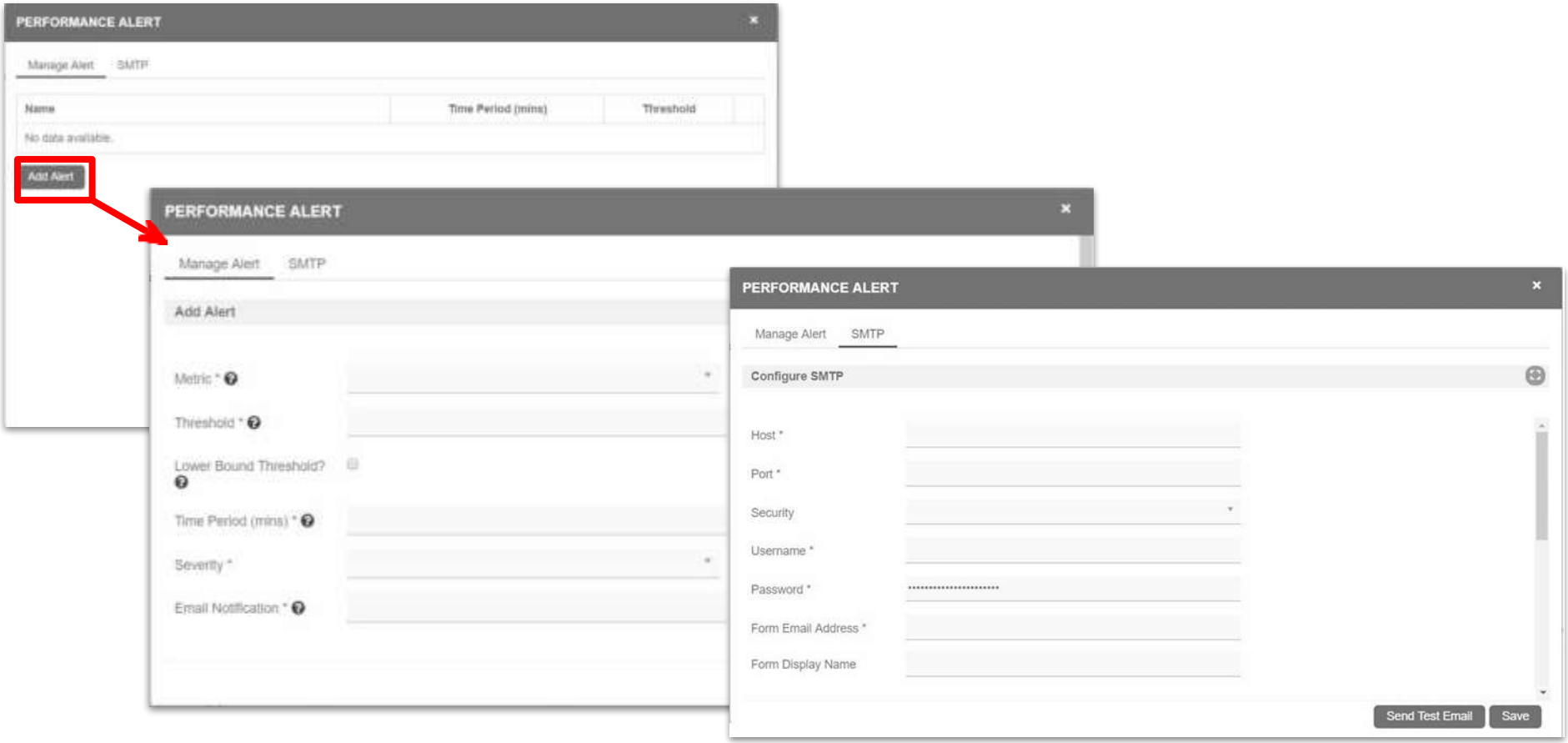
Platform-level Performance Monitoring

- Can be accessed via Admin Bar > Monitor > Performance.



Manage Alerts

- At the platform-level, Admin can configure alert email notifications based on a selection of metrics.



Chapter Review

- General understanding on how APM works in RADS.

Module Review

1. Typical stack for RADS.
2. Basic Database Management (MariaDB).
3. Basic Application Server Management (Tomcat).
4. Web Log Viewer.
5. Application Performance Monitoring.

Recommended Further Learning

- Best Practices on Application Building
- Server performance tuning and hardening.
- Database server performance tuning.

Stay Connected With RADS

- rads.purwana.net
- https://github.com/radservice/rad-community