

Implementación de RestFul con RStudio y Amazon Web Service

Diego Antonio Montañez Sáenz
 Maestría en Ciencias de la Información y las Comunicaciones
 Universidad Distrital Francisco José de Caldas
 Bogotá, Colombia
 Email: damontanezs@correo.udistrital.edu.co

Resumen—En este trabajo se presenta la implementación de una interface RestFul para alcanzar servicios de un servidor R montado sobre Amazon Web Service, se implementa sobre AWS una máquina para instalar posteriormente RStudio. Se genera un servicio sobre el servidor el cuál es alcanzado por medio de una interface RestFul.

Index Terms—RestFul, AWS, RStudio, Shiny Server, Plumb

1. INTRODUCCIÓN

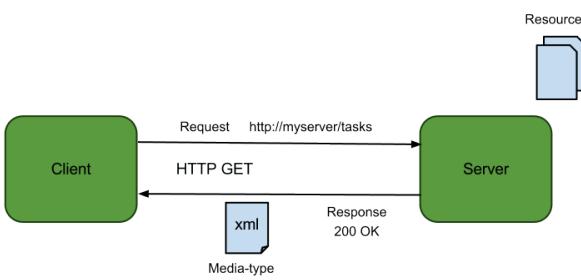


Figura 1: RestFul

REST define un set de principios arquitectónicos por los cuales se diseñan servicios web haciendo foco en los recursos del sistema, incluyendo cómo se accede al estado de dichos recursos y cómo se transfieren por HTTP hacia clientes escritos en diversos lenguajes. REST emergió en los últimos años como el modelo predominante para el diseño de servicios. De hecho, REST logró un impacto tan grande en la web que prácticamente logró desplazar a SOAP y las interfaces basadas en WSDL por tener un estilo bastante más simple de usar.

REST no tuvo mucha atención cuando Roy Fielding lo presentó por primera vez en el año 2000 en la Universidad de California, durante la charla académica “Estilos de Arquitectura y el Diseño de Arquitecturas de Software basadas en Redes”, la cual analizaba un conjunto de principios arquitectónicos de software para usar a la Web como una plataforma de Procesamiento Distribuido. Ahora, años después de su presentación, comienzan a aparecer varios frameworks REST y se convertirá en una parte integral de Java 6 a través de JSR-311.

Una implementación concreta de un servicio web REST sigue cuatro principios de diseño fundamentales

1. Utiliza los métodos HTTP de manera explícita
2. No mantiene estado

3. Expone URIs con forma de directorios
4. Transfiere XML, JavaScript Object Notation (JSON), o ambos [1]

2. IMPLEMENTACIÓN

Para la implementación del servidor se utiliza Amazon Web Service (AWS), es necesario tener una cuenta en el sistema para acceder a los servicios [2]. Se ingresa a la cuenta como se observa en la figura 2.

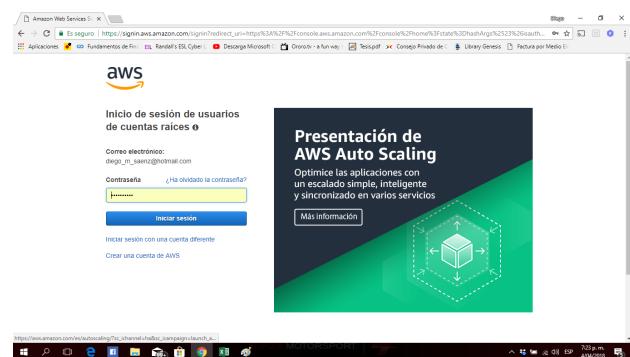


Figura 2: Login AWS

Una vez en AWS se despliega la vista que se muestra en la figura 3, donde se puede observar el panel con todas las funcionalidades que permite tener el desarrollo. Para este ejercicio en específico se utiliza EC2 de los servicios de computación y VPC de redes. Lo primero a crear es el VPC “Virtual Private Cloud” que es un espacio del servicio reservado para los desarrollos a realizar, para esto se selecciona el servicio del panel como se muestra en la figura 3.

RESTFUL CON RSTUDIO Y AWS

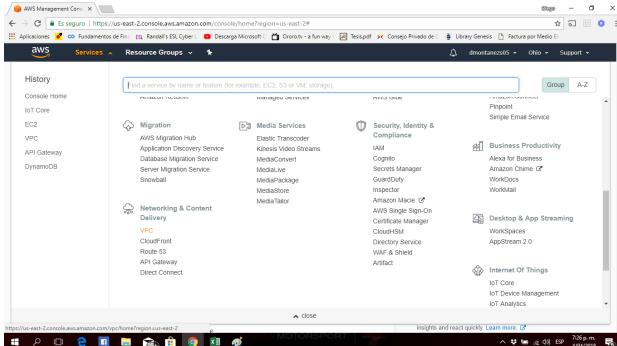


Figura 3: Panel servicios AWS

Se despliega la vista mostrada en la figura 4 y se selecciona “Start VPC Wizard”. Paso a seguir se da “Select” para crear una VPC estándar como se muestra en la figura 5, con esto se crea una red con máscara /16 y subredes /24.

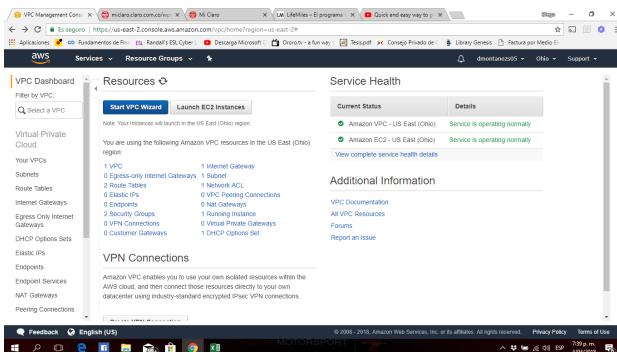


Figura 4: Panel VPC

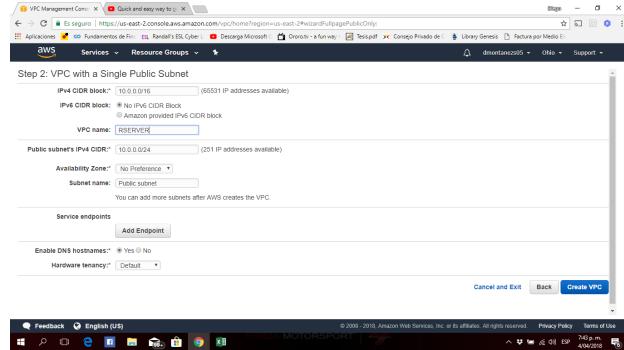


Figura 6: Colocar nombre a la VPC

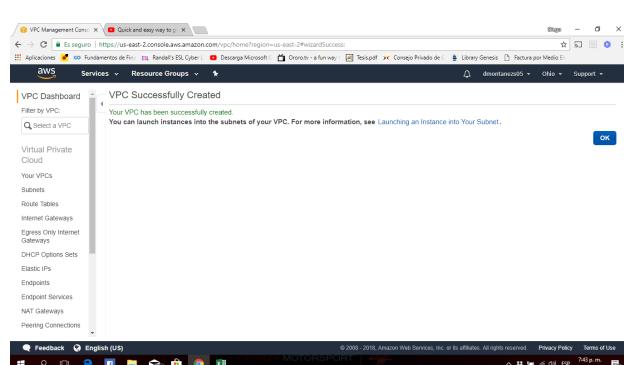


Figura 7: Confirmación creación VPC

El paso a seguir es crear es el EC2 “Amazon Elastic Compute Cloud” el cuál proporciona capacidad de computación escalable en la nube de AWS. El uso de Amazon EC2 elimina la necesidad de invertir inicialmente en hardware, de manera que puede desarrollar e implementar aplicaciones en menos tiempo. Para esto se selecciona en el panel principal de AWS el servicio EC2 como se observa en la figura 8.

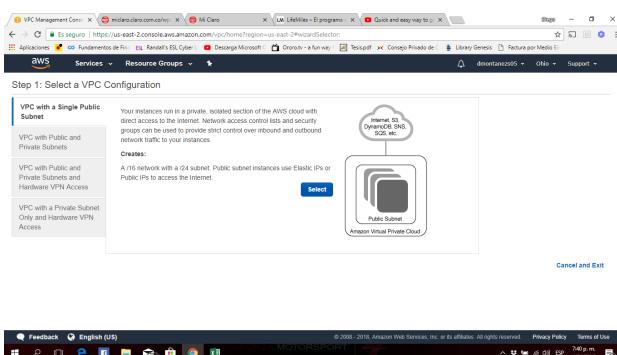


Figura 5: Seleccionar VPC estándar

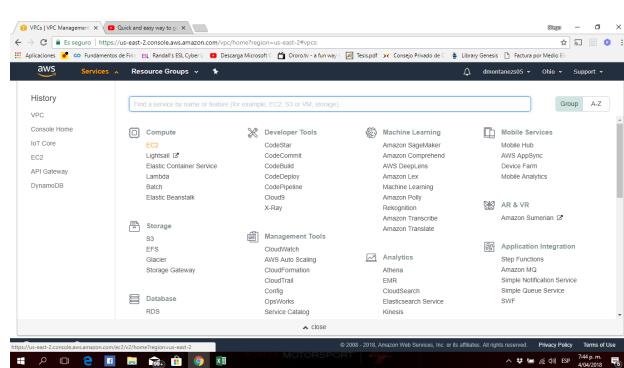


Figura 8: Selección servicio EC2 en panel AWS

Posteriormente se despliega la vista mostrada en la figura 9, donde asignamos el nombre de la VPC, para este caso se denomina “RStudio” y se selecciona “Create VPC”, se despliega la ventana mostrada en la figura 10 donde se confirma la creación de la VPC.

Al ingresar se muestra la vista de la figura 9, se procede a crear el servidor virtual seleccionado “Launch Instance”. Una vez realizado esto se despliega la vista mostrada en la figura 10.

RESTFUL CON RSTUDIO Y AWS

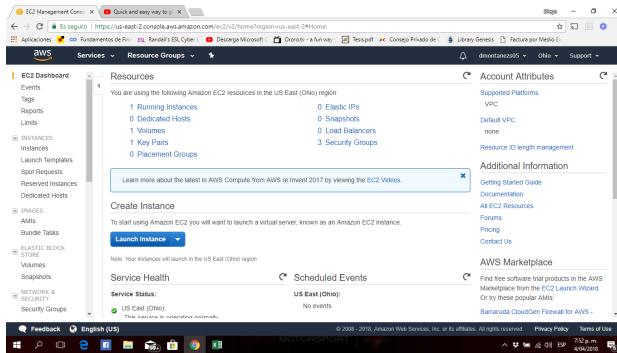


Figura 9: Crear EC2

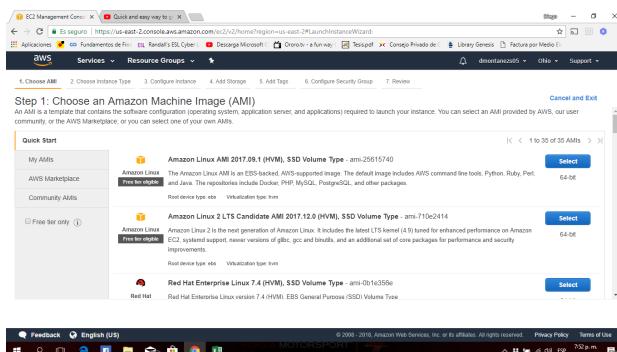


Figura 10: Seleccionar imagen máquina virtual

Se selecciona la máquina virtual que se desea montar, para el caso de este ejercicio se escoge “Amazon Linux AMI 2017...” teniendo en cuenta que tiene Phyton, Ruby, PHP, MySQL entre otras previamente instaladas que pueden ser útiles para futuros desarrollos.

Una vez se selecciona se despliega la vista mostrada en la figura 11 donde se puede seleccionar la capacidad de memoria, procesamiento entre otros, para este caso se selecciona la segunda opción ya que es gratuita¹.

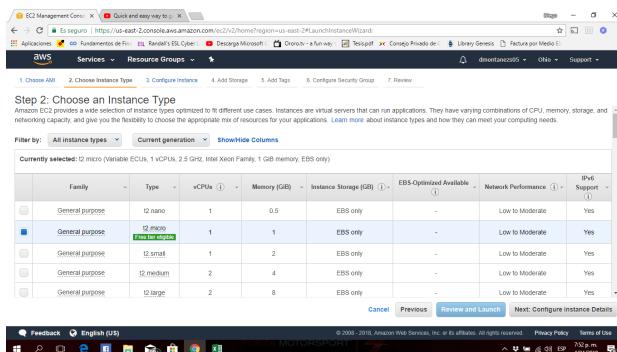


Figura 11: Selección tipo de instancia EC2

Se continúa con la configuración de la máquina seleccionando “Next: Configure Instance Details”, se despliega lo visualizado en la figura 12, en este paso en “Network” se selecciona la VPC creada anteriormente, para el caso de este

1. Las demás opciones tienen costo por lo que se debe ser muy cuidadoso con la selección, si utilizamos más recursos de los necesarios se pueden elevar los costos del proyecto

ejercicio se denominó “RStudio”. Igualmente en el campo “Auto-assign Public IP” se debe seleccionar “Enable” para poder ingresar a la máquina desde afuera con una red pública.

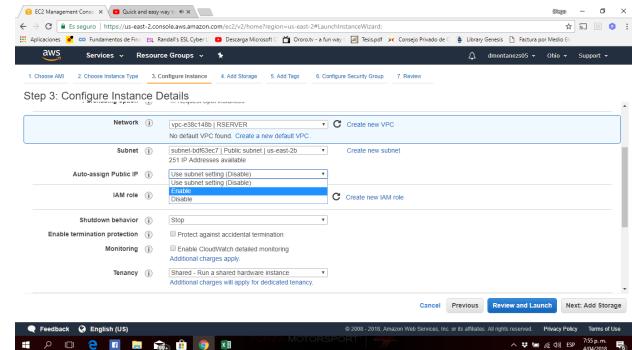


Figura 12: Selección VPC para EC2 y direccionamiento

Para continuar el proceso se selecciona “Review and Launch” y se visualiza lo mostrado en la figura 13, ahora es importante validar que la máquina tenga abierto el puerto 22 para SSH ya que es la forma como se accede a la máquina desde una red externa. Para esto se debe seleccionar “Edit security groups”, se despliega la ventana mostrada en la figura 14. Por defecto el puerto 22 viene abierto pero en caso de que no fuese así en “Add Rule” lo podemos agregar, en “Type” debe ser SSH y en “Source” se escoge “anywhere” para que se pueda acceder desde cualquier IP.

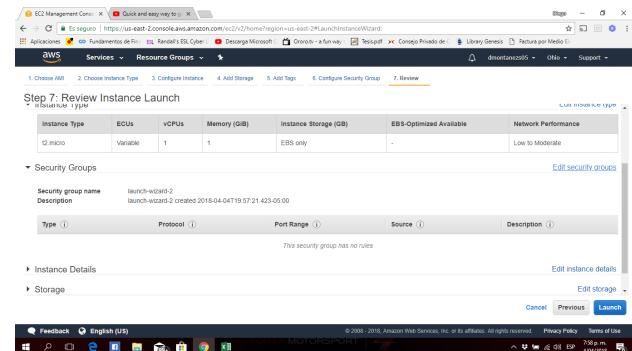


Figura 13: Configuración EC2

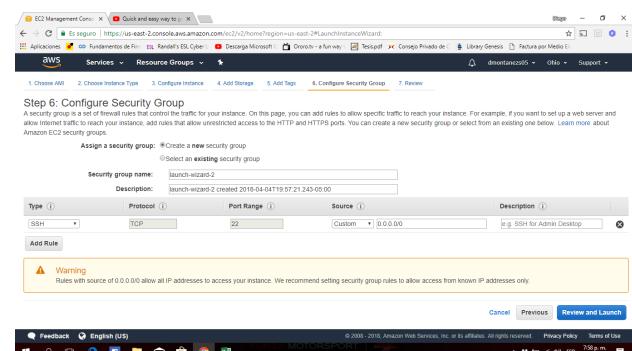


Figura 14: Configuración EC2

Se selecciona posteriormente “Review and Launch” volviendo a la ventana mostrada en la figura 13 y se da

RESTFUL CON RSTUDIO Y AWS

“Launch” para crear la máquina. Se despliega la ventana mostrada en la figura 15 donde en la lista desplegable se debe seleccionar “Create a new key pair”, se coloca el nombre deseado en “Key pair name” y se selecciona “Download Key Pair”, se despliegara un “Filebox” para seleccionar el lugar donde se desea sea guardada la clave. El proceso de descargar la clave es fundamental para proteger la máquina ya que esta clave será utilizada para acceder al sistema desde una red externa.

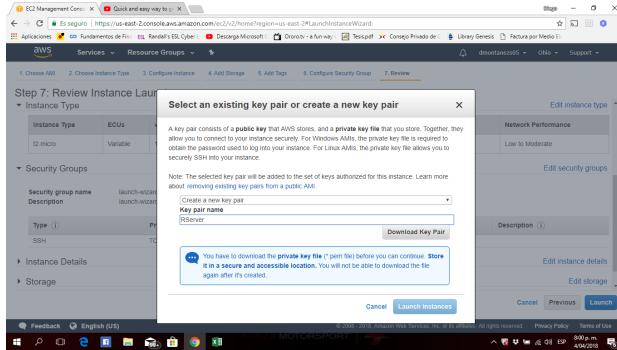


Figura 15: Configuración EC2

Continuando con el proceso seleccionando “Launch instance” despliegándose la vista presentada en la figura 16, en la parte inferior se debe escoger “View instance” donde se observará ya la máquina creada como se muestra en la figura 17²

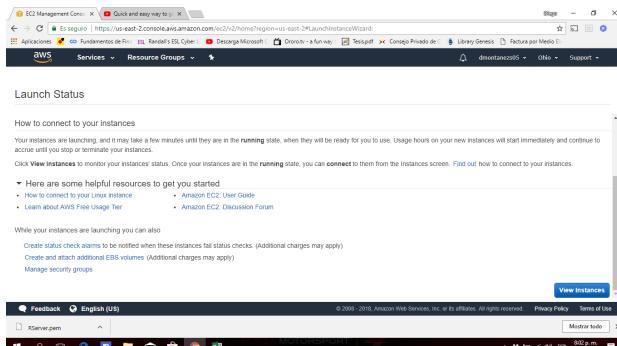


Figura 16: Guardar clave instancia EC2

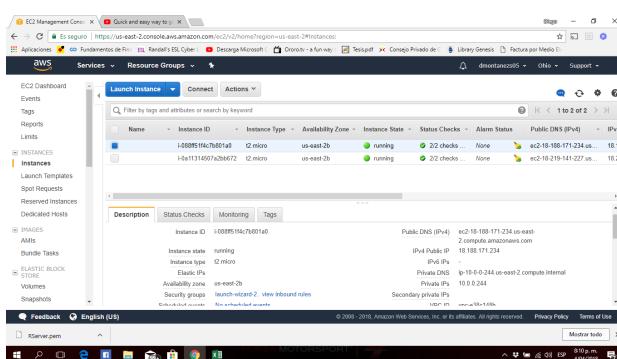


Figura 17: Creación instancia EC2

2. Se debe esperar unos minutos (esto depende de la configuración de la máquina), mientras se inicializa la instancia

Igualmente en la parte inferior se puede observar los datos de interés de nuestra máquina como lo son la IP pública y el DNS, para este caso la IP es 18.188.171.234, con estos datos se procede a conectarse con la máquina [3] para instalar el RStudio y demás sistemas necesarios para el funcionamiento de la APP RestFul.

Se utiliza Putty para establecer la conexión [4], lo primero que se debe realizar es autenticar la clave, para esto se abre PuttyGen como se muestra en la figura ?? y se selecciona “Load”. se selecciona el archivo guardado al momento de crear la instancia EC2 y posteriormente se selecciona “Save private Key”. Se despliegara un cuadro de diálogo como se observa en la figura 18 donde se debe seleccionar “Si”, un nuevo “FileBox” se abrirá donde se debe seleccionar el lugar y nombre con el que queremos guardar la nueva clave (es un archivo con extensión ppk).

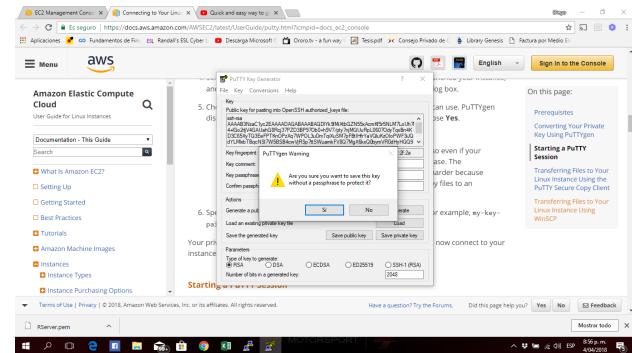


Figura 18: Autenticar clave

Con la clave autenticada se procede a establecer la conexión, para esto se abre Putty como se muestra en la imagen 19, por defecto el puerto de conexión es el 22 por el tipo de conexión SSH, en “Host Name” se debe colocar “user_name@public_dns_name”, para el caso puntual de una máquina Amazon Linux como la instalada el “user_name” es “ec2user” y el dominio se observa en las características de la instancia de la figura 17.

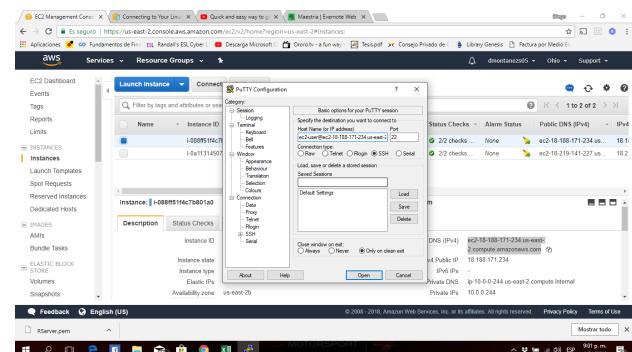


Figura 19: Conexión por Putty

Adicional a esto se debe cargar la clave generada en el paso anterior como se observa en la figura 20, en la parte derecha en la sección “Connection-SSH-Auth”, se selecciona “Browse” y se carga el archivo ppk. Finalmente se da “Open” estableciendo la conexión con la máquina como se observa en la figura 21.

RESTFUL CON RSTUDIO Y AWS

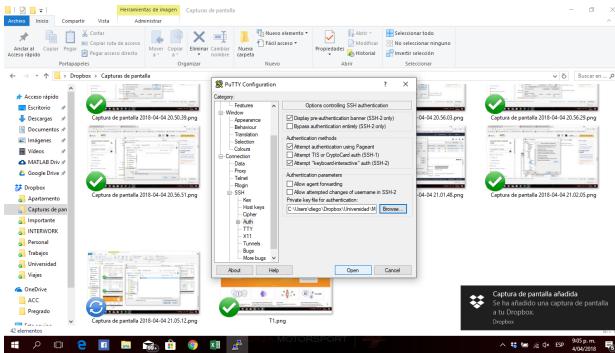


Figura 20: Conexión por Putty

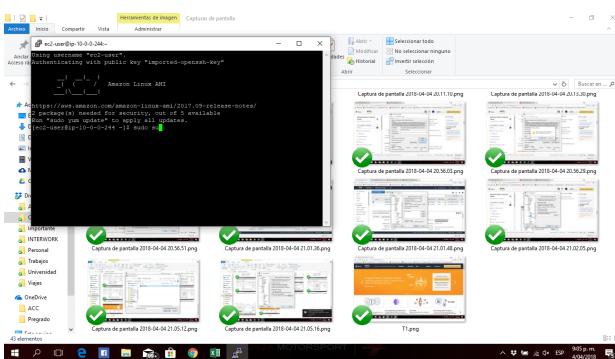


Figura 21: Conexión por Putty

El paso a seguir es instalar R, RStudio (Para la última versión consultar [5]) y ShinyServer en la máquina [6], estas herramientas son necesarias para el desarrollo de la APP RestFul. Los comandos a ingresar sobre la consola son los siguientes:

```
#!/bin/bash
#instalacion de R
yum install -y R

#instalacion R Studio Server Version:
1.1.442 Released: 2018-03-12
wget https://download2.rstudio.org/rstudio-server-rhel-1.1.442-x86_64.rpm
yum install -y --nogpgcheck rstudio-server-rhel-1.1.442-x86_64.rpm
rm rstudio-server-rhel-1.1.442-x86_64.rpm

#instalacion shiny server (2017-08-25)
R -e "install.packages('shiny', repos='http://cran.rstudio.com/')"
wget https://download3.rstudio.org/centos5.9/x86_64/shiny-server-1.5.4.869-rh5-x86_64.rpm
yum install -y --nogpgcheck shiny-server-1.5.4.869-rh5-x86_64.rpm
rm shiny-server-1.5.4.869-rh5-x86_64.rpm

#Agregar usuarios , para este ejemplo se agrego un username (Diego) con password (Diego)
```

```
yum install -y curl-devel
useradd Diego
echo Diego:Diego | chpasswd
```

En la figura 22 se muestra la creación del usuario.

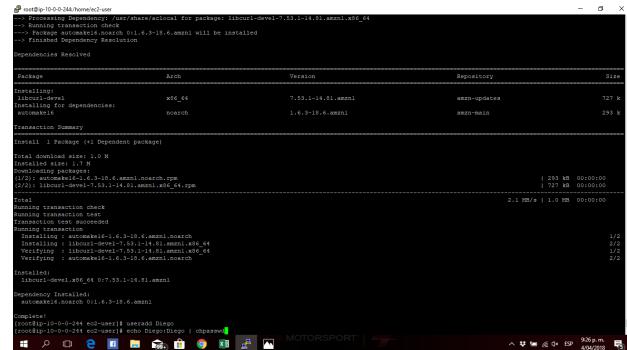


Figura 22: Instalación RStudio y demás en la máquina

Ya se tiene el servidor instalado, sin embargo, para acceder a los servicios es necesario abrir los puertos 8787 y 3838 de la máquina instalada, para esto en la vista principal de la instancia se selecciona el grupo de seguridad que para este caso es “*launch-wizard-2*” como se observa en la figura 23

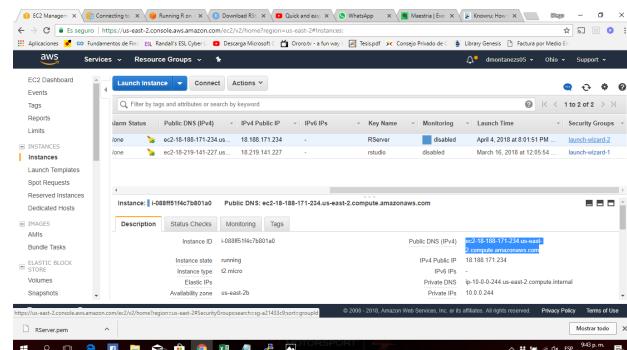


Figura 23: Configuración puertos

Una vez ingresado se visualiza lo mostrado en la figura 24, se selecciona “*Inbound*” y “*Edit*”

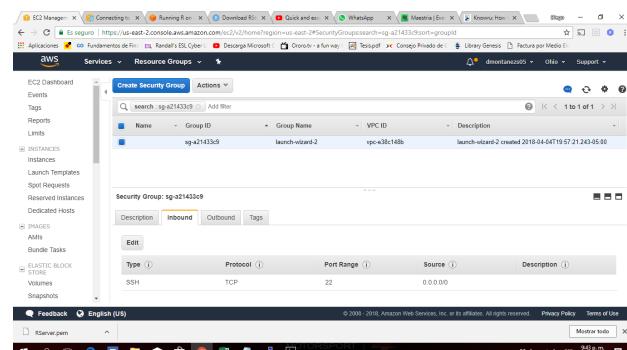


Figura 24: Configuración puertos

Se visualiza lo mostrado en la figura 25, se selecciona “*Add Rule*” y se agregan los puertos como se muestran en la figura 26.

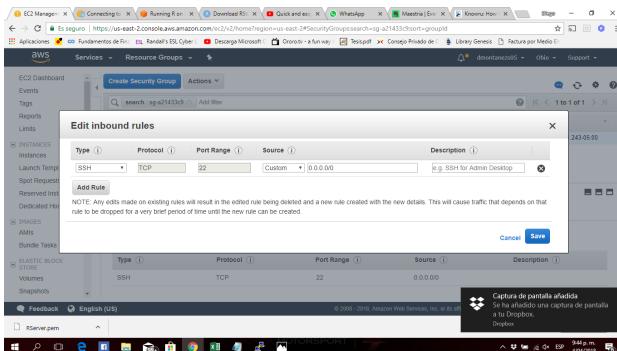


Figura 25: Configuración puertos

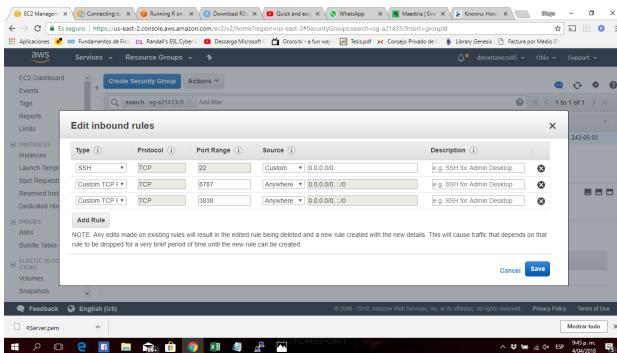


Figura 26: Configuración puertos

Una vez se hayan guardado los cambios, se puede ingresar vía web al RStudio instalado, para esto se debe utilizar la IP pública de la máquina la cuál se visualiza en la vista principal de la instancia, ver figura 17, se debe agregar el número del puerto que se abrió para R el cuál es 8787 como se muestra en la figura 27.

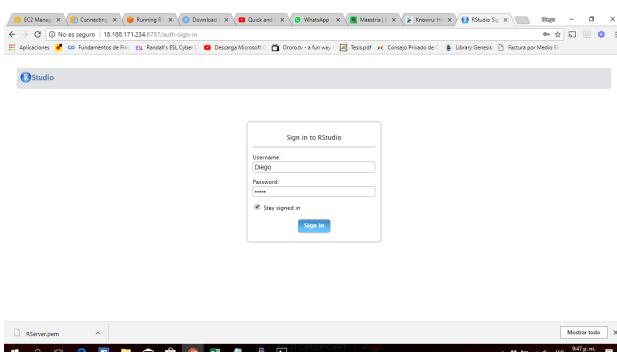


Figura 27: Ingreso a RStudio

Se ingresa con el usuario creado (Username: Diego y Password: Diego) y se tiene acceso a R sobre la máquina de Linux como se observa en la figura 28.

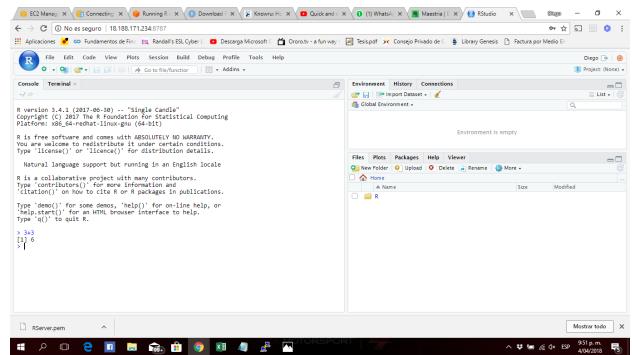


Figura 28: R sobre AWS

Con el servidor funcionando se procede con la creación del servicio, para esto se toma un ejemplo de la literatura [7]. En este ejercicio se plantea obtener la probabilidad de pago de una persona para un crédito teniendo en cuenta las condiciones del préstamo y los históricos de pago. Primero se debe crear el modelo de inteligencia computacional, en la consola se coloca el código descrito a continuación:

```

url <- "https://archive.ics.uci.edu/ml/
machine-learning-databases/statlog/
german/german.data"
col.names <- c(
  'Status.of.existing.checking.account', '
  Duration.in.month', 'Credit.history'
  , 'Purpose', 'Credit.amount', 'Savings-
  account/bonds'
  , 'Employment.years', 'Installment.rate.in-
  percentage.of.disposable.income'
  , 'Personal.status.and.sex', 'Other-
  debtors/_guarantors', 'Present-
  residence.since'
  , 'Property', 'Age.in.years', 'Other-
  installment.plans', 'Housing', 'Number
  .of.existing.credits.at.this.bank'
  , 'Job', 'Number.of.people.being.liable.to
  .provide.maintenance.for', 'Telephone'
  , 'Foreign.worker', 'Status'
)
# Get the data
data <- read.csv(
  url
  , header=FALSE
  , sep=' '
  , col.names=col.names
)
library(rpart)
# Build a tree
# I already figured these significant
# variables from my first iteration (not
# shown in this code for simplicity)
decision.tree <- rpart(
  Status ~ Status.of.existing.checking.
  account + Duration.in.month + Credit.
  history + Savings.account.bonds
  , method="class"
  , data=data
)

```

```

)
install.packages("rpart.plot")
library(rpart.plot)
# Visualize the tree
# 1 is good, 2 is bad
prp(
  decision.tree
, extra=1
, varlen=0
, faclen=0
, main="Decision Tree for German Credit Data"
)
save(decision.tree, file='decision_Tree_for_german_credit_data.RData')

```

El resultado se muestra en la figura 29, paso a seguir se crea la función que recibe los datos y se las pasa al modelo para el cálculo de la probabilidad, para esto se crea un *Script* con el nombre “*deploy_ml_credit_model.R*” agregando el siguiente código como se observa en la figura 30.

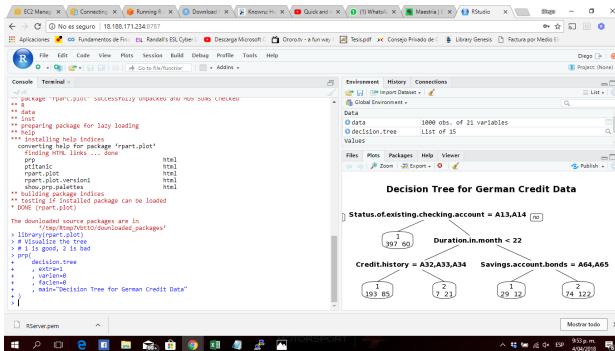


Figura 29: Creación modelo inteligencia computacional

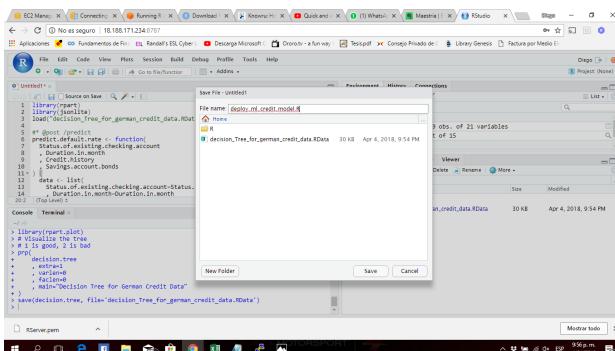


Figura 30: Creación Script

```

library(rpart)
library(jsonlite)
load("decision_Tree_for_german_credit_data.RData")

## @post /predict
predict.default.rate <- function(
  Status.of.existing.checking.account
, Duration.in.month
)

```

```

, Credit.history
, Savings.account.bonds
) {
  data <- list(
    Status.of.existing.checking.account=
      Status.of.existing.checking.account
    , Duration.in.month=Duration.in.month
    , Credit.history=Credit.history
    , Savings.account.bonds=Savings.account.bonds
  )
  prediction <- predict(decision.tree, data)
  return(list(default.probability=unbox(
    prediction[1, 2])))
}

```

Luego en la consola se instala el paquete de *Plumber* por medio del siguiente comando.

```
install.packages("plumber")
```

Plumber permite convertir las funciones de R en interfaces de RestFul. El último paso es iniciar el servidor, se crea un objeto con *Plumber* donde se asigna una función a la interface. En la consola se debe agregar el siguiente código como se muestra en la figura 31 (Se comparte el código para descarga en <https://github.com/diegoanmonsa/Restful>).

```

library(plumber)
r <- plumb("deploy_ml_credit_model.R")
r$run(host="0.0.0.0", port=8000)

```

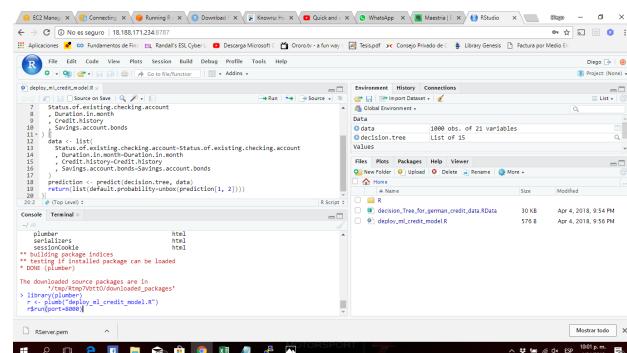


Figura 31: Servidor corriendo

Se aconseja agregar *host="0.0.0.0"* para asegurar que se pueda alcanzar desde cualquier IP y no solo de forma local, igualmente en la máquina se debe abrir el puerto 8000 ya que es el puerto por el cuál se abrió el servicio; se debe realizar de la misma forma como se abrió el puerto 8787.

Para finalizar se debe probar el servicio, se utiliza el complemento de google chrome *R Client* el cuál es una interface de RestFul para enviar solicitudes al servidor.

RESTFUL CON RSTUDIO Y AWS

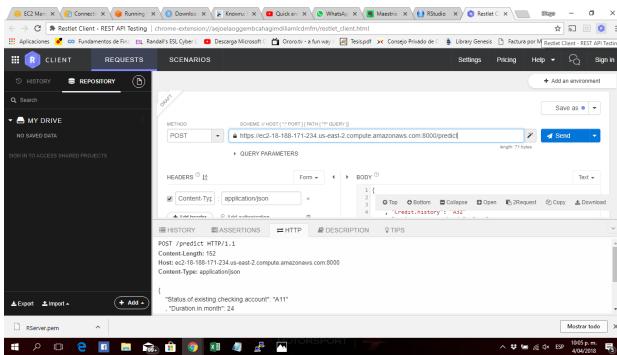


Figura 32: R client

Como se muestra en la figura 32 se escoge el método "POST" para enviar información por medio de las primitivas "HTTP" al servidor, la URL es la IP o DNS de la máquina con el puerto que se abrió y la función a utilizar.

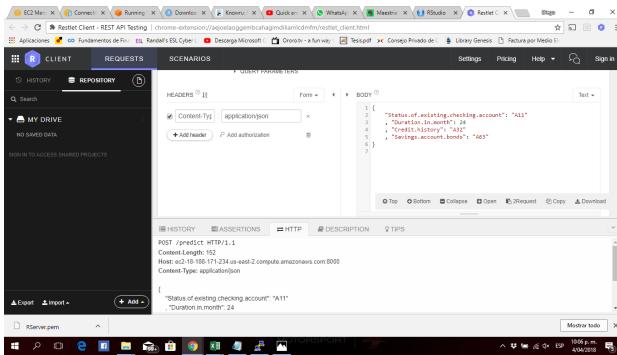


Figura 33: Parámetros de entrada

Se deben agregar las entradas a la función como se muestra en la figura 33 y se envía la solicitud al servidor. La respuesta se muestra en la figura 34 donde el resultado coincide con lo descrito en la literatura de acuerdo a las entradas dadas.

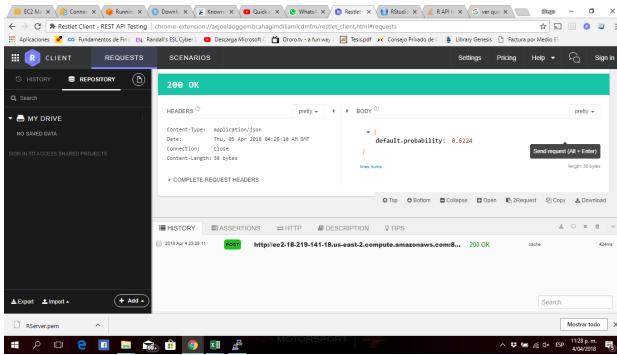


Figura 34: Respuesta del servidor

Con esto se finaliza la implementación del servicio Rest-Ful con R sobre AWS.

REFERENCIAS

- [1] Dos ideas, personas y software. <https://dosideas.com/noticias/java/314-introduccion-a-los-servicios-web-restful>, Agosto de 2018.
- [2] Amazon Web Services, Inc. https://docs.aws.amazon.com/es_es/rekognition/latest/dg/setting-up.html, Agosto de 2018.
- [3] Amazon Web Services, Inc. <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AccessingInstances.html>, Agosto de 2018.
- [4] Putty.org. <https://www.putty.org/>, Agosto de 2018.
- [5] RStudio. <https://www.rstudio.com/products/rstudio/download-server/>, Agosto de 2018.
- [6] Amazon Web Services, Inc. <https://aws.amazon.com/es/blogs/big-data/running-r-on-aws/>, Agosto de 2018.
- [7] KNOWRU LIMITED. <https://www.knowru.com/blog/how-create-restful-api-for-machine-learning-credit-model-in-r/>, Agosto de 2018.