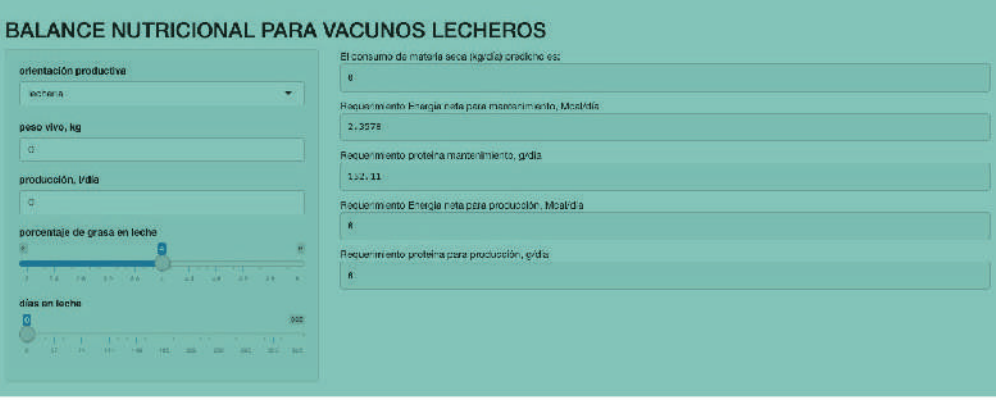


Capítulo 6.

Desarrollo de una aplicación web: herramienta para facilitar el proceso de aprendizaje

Ricardo Rosero Noguera
Zoot., Esp, MSc., D.Sc. Profesor asociado, Facultad de Ciencias Agrarias, Universidad de Antioquia, Grupo GRICA

Sandra Posada Ochoa
Zoot., Esp., MSc, D.Sc. Profesora asociada, Facultad de Ciencias Agrarias, Universidad de Antioquia, Grupo GRICA

Resumen

El objetivo de este documento es permitir a profesores y estudiantes explorar paso a paso el desarrollo de una aplicación web interactiva orientada a fortalecer y facilitar el proceso de enseñanza-aprendizaje en la virtualidad. Adicionalmente se discuten las ventajas y desventajas de la implementación de este tipo de tecnologías y se relata nuestra percepción como docentes y la percepción de los estudiantes sobre el empleo de estas en el curso Nutrición de rumiantes.

Palabras clave: *internet, navegador web, software, TIC, ambiente virtual de aprendizaje*

1. Introducción

El año 2020 será recordado como el año de la pandemia que encerró al mundo, pero también como el año que cambió la forma de trabajar, de interactuar, de hacer comercio y de enseñar. La necesidad de mantener el aislamiento y la distancia social obligó a los diferentes sectores de la economía y la sociedad a buscar nuevas alternativas para continuar con su tan necesaria actividad. En este escenario, las tecnologías de la información y la comunicación (TIC) cobraron mayor relevancia, pues permitieron la interacción humana a todo nivel entregando información confiable en tiempo real, evitando la interacción física y, de esta manera, desacelerando el ritmo de expansión del virus.

En la educación, la suspensión de clases presenciales modificó el sistema educativo al desplazar las aulas de clase de las escuelas a los hogares. Frente a este desafío, el gobierno nacional promovió iniciativas para la enseñanza remota y dar así continuidad al proceso educativo (Laboratorio de Economía de la Educación -LEE, 2021). La implementación de estas iniciativas a nivel nacional puso en evidencia debilidades en el sistema asociadas con el acceso a internet, el número y disponibilidad de los dispositivos electrónicos y la falta de competencias de los docentes y estudiantes en el manejo de las nuevas tecnologías.

A pesar de lo anterior, resulta innegable el importante papel que juegan las TIC en la educación. Para nosotros, internet es una herramienta educativa porque sus características coinciden en gran medida con los intereses centrales de la educación, como son el intercambio de información, la comunicación y la generación de conocimiento. En este contexto, el objetivo de este documento es ofrecer a profesores y estudiantes un paso a paso para el desarrollo de una aplicación web interactiva orientada a fortalecer y facilitar el proceso de enseñanza-aprendizaje en la virtualidad. Para ello se presenta un estudio de caso.

2. Metodología

El paquete *Shiny* de *R* fue empleado para desarrollar una aplicación web orientada al apoyo del desarrollo del curso Nutrición de rumiantes del programa de Zootecnia de la Facultad de Ciencias Agrarias de la Universidad de Antioquia. Uno de los objetivos de este curso es que los estudiantes establezcan las necesidades diarias de proteína y energía de vacunos lecheros mediante el empleo de las tablas de requerimientos nutricionales publicadas por el NRC (1989, 2001). Para alcanzar este objetivo el estudiante debe manejar los conceptos de consumo de materia seca (kg/día), peso vivo (kg), peso vivo metabólico (kg0.75), producción y composición de la leche, producción de leche corregida al 4% de grasa (l/día) y fisiología de la lactancia.

Por lo general, los estudiantes reciben en el salón de clase la explicación de cada uno de los conceptos antes descritos y su aplicación mediante el desarrollo de ejercicios prácticos. Durante nuestros años como docentes hemos notado que los estudiantes tienen dificultades en la aplicación de modelos matemáticos orientados a calcular el consumo de materia seca y determinar los requerimientos diarios de proteína y energía de vacas lactantes. Estas dificultades hacen que su desempeño en el componente práctico de esta asignatura no sea óptimo.

Ante las dificultades ocasionadas por la pandemia, los profesores del área de nutrición decidimos buscar alternativas innovadoras que facilitaran el proceso de adopción y aplicación de conceptos por parte de los estudiantes. Una de las alternativas fue el desarrollo de aplicaciones web orientadas a facilitar el proceso de aprendizaje.

A continuación se describe paso a paso la construcción de una aplicación web que, más que convertirse en un tutorial, busca motivar a profesores de cualquier área a desarrollar este tipo de herramientas. Finalmente, se

relata nuestra percepción como docentes y la percepción de los estudiantes en el empleo de estas aplicaciones en el curso Nutrición de rumiantes.

3. Resultados y discusión

3.1. Shiny

La aplicación web fue desarrollada en *Shiny*, un paquete de *R* de código abierto que proporciona un marco web elegante y potente para crear aplicaciones web utilizando el paquete estadístico *R* (R Studio, 2020). Esta herramienta permite el desarrollo de aplicaciones web sin necesidad de conocimientos previos de HTML, CCS o JavaScript (Beeley, 2013; Resnizky, 2015).

El primer paso consistió en la instalación del paquete estadístico *R*, para lo cual es necesario que el lector se dirija a la página: https://cran.r-project.org, donde encontrará la última versión de *R*, así como sus versiones previas. *R* puede ser instalado en los sistemas operativos Linux, Windows y Mac OS X. Para hacer el entorno de trabajo más amigable se recomienda instalar *RStudio*, un entorno de desarrollo integrado para el lenguaje de programación *R*. *RStudi*o puede ser descargado de forma gratuita en la página: https://rstudio.com/products/rstudio/download/ .

El segundo paso corresponde a la instalación del paquete *Shiny*, lo que puede ser realizado directamente desde la consola de *R Studio* mediante el comando: install.packages("shiny")

De acuerdo con Gómez et al. (2016), algunas de las características de *Shiny* son:

- Las aplicaciones se programan empleando código *R*, pero también pueden ser programadas en HTML, CSS y JavaScript, dependiendo de las preferencias del usuario.

- Programación reactiva que posibilita el desarrollo de aplicaciones interactivas, permitiendo al usuario manipular los componentes de la aplicación sin recurrir a la modificación del código de programación.
- Las aplicaciones emplean el framework *Bootstrap*, el cual permite desarrollar aplicaciones que se adaptan a cualquier dispositivo.
- *Shiny* incorpora una amplia variedad de widgets, los cuales permiten la entrada de valores por parte del usuario de forma amigable.
- Permite compartir las aplicaciones de forma gratuita a través del servidor de *RStudio* (ShinyApss.io).

3.1.1. Estructura de una aplicación en Shiny

Cuando se crea una aplicación en *Shiny,* se crea una carpeta que contiene dos archivos (Figura 1):

ui.R: este archivo corresponde a la interfaz del usuario (UI) y es aquí donde se programa la presentación de la aplicación. La UI permite controlar la presentación visual de la aplicación y la distribución de cada uno de sus componentes, los cuales pueden incluir: texto, widgets, gráficas, barras de navegación, pestañas y cuadros de texto.

server.R: es un archivo en el que se programa la lógica de la aplicación, recibe los inputs del UI y con ellos genera los outputs, permitiendo controlar los datos que se mostrarán a través de la interfaz del usuario.

Aunque es posible crear en un solo archivo la UI y el server, es una buena práctica separarla en dos archivos para facilitar cualquier cambio en la programación y futuros mantenimientos o actualizaciones de la aplicación.

3.1.2. Desarrollo de la aplicación web

En primera instancia se debe abrir *Rstudio*. En el menú "File" desplazarse a "New File" y seleccionar la opción "Shiny Web App…" (Figura 2).

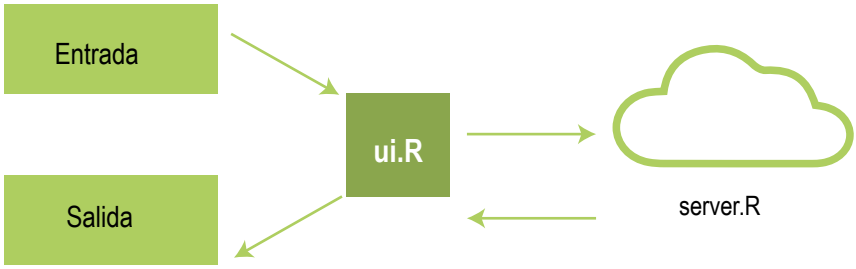


Figura 1. Esquema de una aplicación creada en Shiny
Fuente: elaboración propia

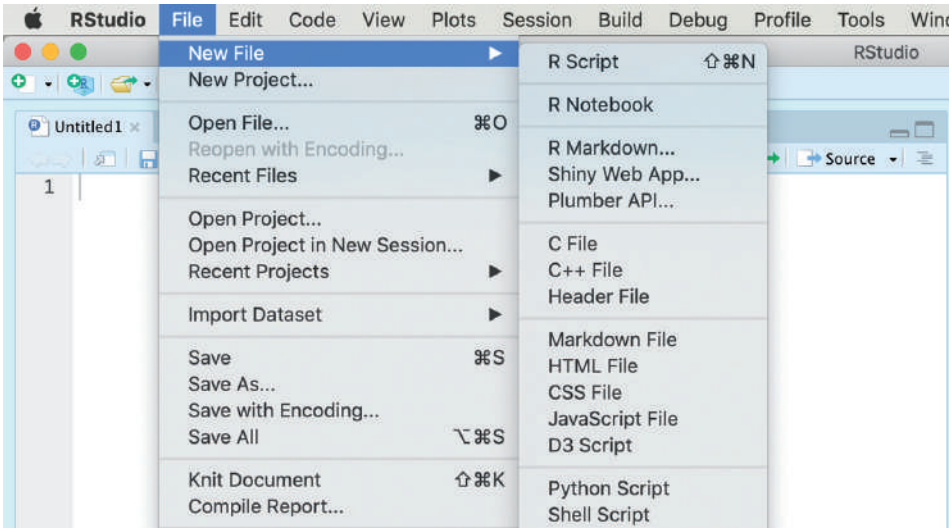


Figura 2. Menú para iniciar el desarrollo de la aplicación web [captura de pantalla]
Fuente: elaboración propia

Al seleccionar la opción "Shiny Web App…" se despliega un nuevo menú en el que se solicita el nombre de la aplicación y el lugar donde se desea guardar en el computador (Figura 3). Para el ejemplo se seleccionó el nombre "BALANCE" y el directorio donde se almacenará la aplicación será en el escritorio ("Desktop"). En este punto es recomendable marcar la opción "Multiple File (ui.R/server.R)" para crear dos archivos separados co-

rrespondientes a la interfaz de usuario (UI) y el server. Automáticamente, *Rstudio* crea dos archivos y carga un ejemplo llamado "Old Faithful Geyser Data". En tanto nuestro objetivo es construir una aplicación desde cero, procedemos a borrar el contenido completo de los archivos ui.R y server.R.

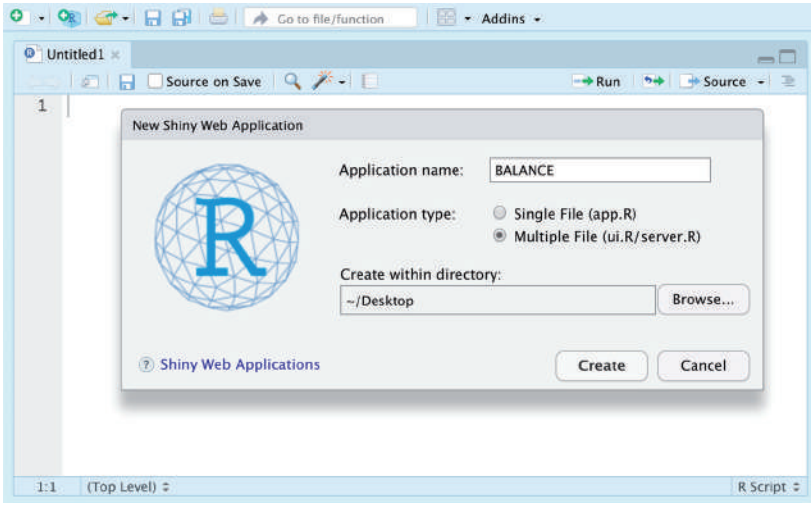


Figura 3. Creación y ubicación de la aplicación [captura de pantalla]
Fuente: elaboración propia

Para empezar con nuestra aplicación, cargamos la librería *Shiny* en la UI mediante el comando: library(shiny)

Luego procedemos a definir la UI, pero para ello primero necesitamos conocer algunos elementos de la UI:

- ***fluidPage:*** función que permite definir el layout general.
- ***titlePanel:*** función requerida para definir el título de la aplicación.
- ***sidebarLayout:*** definimos que el diseño de la app va a ser con una barra lateral (sidebarPanel) y un panel central (mainPanel).
- ***sidebarPanel:*** dentro del *sidebarPanel* definimos los elementos que van en la barra lateral.

- ***sliderInput:*** definimos que el input será ingresado desde un widget que puede ser de diferente tipo, dependiendo de la necesidad y preferencias del programador (Figura 4).
- ***mainPanel:*** dentro del *mainPanel* definimos los elementos que van a aparecer en el panel central y contiene las salidas que cambian conforme el usuario modifica los valores de entrada. Las salidas posibles pueden incluir texto, gráficos y mapas.

La Figura 4 muestra el diseño general de la aplicación. En esta figura se puede apreciar el título de la app (titlePanel), la sección destinada a registrar las entradas de información por parte del usuario (sidebarPanel) con múltiples ejemplos de los posibles widgets que permiten registrar las entradas de información y, finalmente, la sección del panel principal (mainPanel), donde se registran las salidas de la aplicación que incluyen valores numéricos, texto, gráficos y mapas.

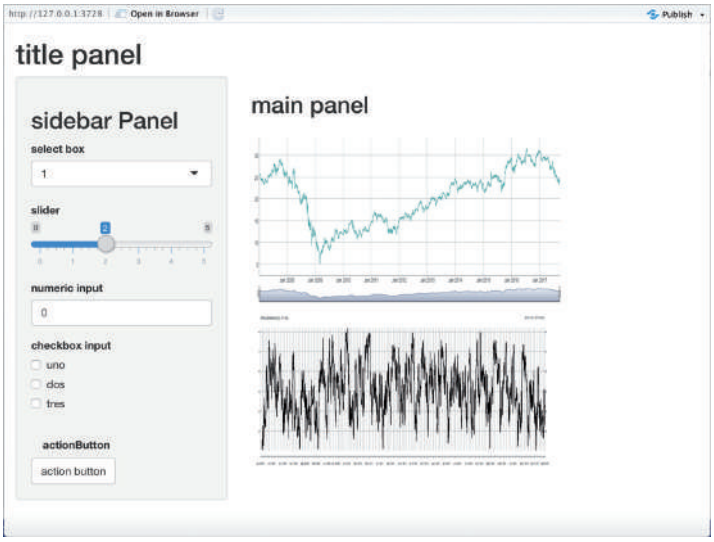
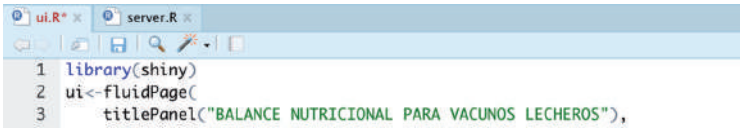


Figura 4. Esquema general de la aplicación web [captura de pantalla]
Fuente: elaboración propia

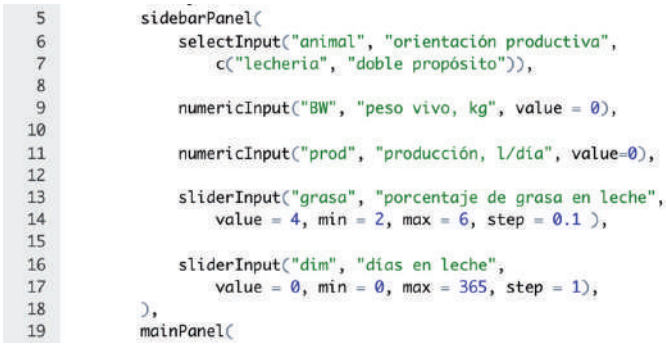
Conocidos los elementos de la UI, procederemos a programar nuestra aplicación. Para facilitar la comprensión del proceso de programación se seguirá el esquema presentado en la Figura 4.

titlePanel: empezamos definiendo en el archivo ui.R el layout general y el título de nuestra aplicación con el siguiente código de programación:



Ahora necesitamos definir el diseño de nuestra aplicación. Para este ejemplo dividiremos el layout general en dos secciones: sidebarPanel y mainPanel (Figura 4). Procedemos de la siguiente manera:

sidebarPanel: en esta sección programaremos las entradas que el usuario debe aportar para alimentar a la aplicación y, de esta forma, proceder a procesar la información y entregar el resultado. El código requerido es el siguiente:



Como puede verificarse dentro de sidebarPanel, hemos programado varios widgets que permiten al usuario entrar la información a la aplicación. Describamos cada uno de ellos:

selectInput: nos permite crear una caja de selección de variables. La variable recibirá el nombre de "animal", el título de la caja de selección será "orientación productiva" y, dentro de la caja, las opciones que el usuario podrá escoger son "lechería" y "doble propósito" .

numericInput: permite una entrada numérica. En el sistema el nombre de la variable será "BW" (abreviación de *Body Weight)*. Es importante aclarar que el programador determina el nombre de las variables. El título de la entrada numérica que orienta al usuario sobre la variable que debe ingresar y sus unidades es "peso vivo, kg" y el valor de inicio para esta variable es cero (value = 0).

sliderInput: nos permite crear una barra deslizante. El nombre de la variable para el sistema será "grasa", el título de la barra será "porcentaje de grasa en leche" y los códigos value = 4, min = 2, max = 6, step = 0.1. Los códigos indican que el valor de inicio para esta variable es 4, que puede tomar valores entre 2 y 6 y que los incrementos o decrementos en el valor serán de 0.1 unidades.

mainPanel: en esta sección programaremos los espacios donde la aplicación reportará los resultados para el usuario. Para programar las salidas emplearemos el siguiente código de programación:

```
            "El consumo de materia seca (kg/día) predicho es:",
            verbatimTextOutput("CMS"),

            "Requerimiento Energía neta para mantenimiento, Mcal/día ",
            verbatimTextOutput("ENlm"),

            "Requerimiento proteina mantenimiento, g/día",
            verbatimTextOutput("PBm"),

            "Requerimiento Energía neta para producción, Mcal/día",
            verbatimTextOutput("ENlp"),

            "Requerimiento proteina para producción, g/día",
            verbatimTextOutput("PBp"),
        ),
    )
)
```

Para facilitar la comprensión de la programación, tomaremos como ejemplo la primera parte del código. El texto entre comillas ("El consumo de materia seca (kg/día) predicho es:") identifica la información que se presenta en la salida. verbatimTextOutput representa una variable de salida reactiva como texto literal dentro de la página de aplicación. Finalmente, el texto "CMS" corresponde al nombre con que el servidor de *Shiny* reconocerá la variable de salida.

Hasta aquí hemos programado la interfaz de usuario UI y nuestra aplicación debería verse como se muestra en la Figura 5.

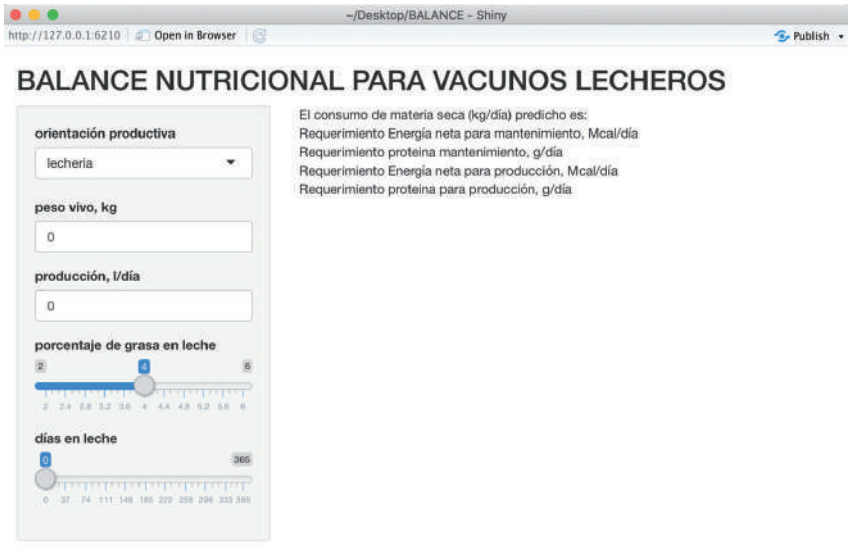


Figura 5. Vista previa de la aplicación [captura de pantalla]
Fuente: elaboración propia

El paso final es programar las salidas reactivas en el archivo server.R. Se llaman salidas reactivas porque cualquier modificación que el usuario realice en la UI automáticamente modifica la salida.

El código de programación para el archivo server.R es el siguiente:

ui.R* server.R

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

  output$CMS<- renderText({
    ((0.372* ((0.4*input$prod)+15*(input$prod)*((input$grasa)/100)))+
       (0.0968*(input$BW^0.75)))*
      (1-(exp(-0.192*((input$dim/7)+3.67))))
  })
  output$ENlm<- renderText({
    ((0.0122*input$BW)+2.3578)
  })
  output$PBm <- renderText({
    ((0.4217*input$BW)+152.11)
  })
  output$ENlp<-renderText({
```

Las funciones del servidor *Shiny* incluyen los parámetros de entrada, salida y sesión (input, output, sesion). La sesión es un entorno que se utiliza para acceder a la información y funcionalidad de la aplicación. Input y output corresponden a las entradas y salidas.

Para comprender la programación tomaremos nuevamente como ejemplo una fracción del código, el cual se aplica de la misma manera para las fracciones restantes. El código que describiremos busca calcular el consumo de materia seca (CMS) en vacas lecheras, según el modelo matemático del NRC (2001).

```
2  output$CMS<- renderText({
```

El código output$CMS<- crea una salida para la etiqueta "CMS" definida previamente en la UI. renderText({, le indica a nuestra aplicación que se trata de una salida reactiva tipo texto que debe ejecutarse automáticamente cada vez que el input cambie.

El modelo propuesto por el NRC (2001) para predecir el CMS en vacas lecheras es el siguiente:

$$MS=((0.372 * PLC4\%) + (0.0968 * BW^{0.75})) *(1-exp\ exp\ (-0.192 * (SL+3.67)))$$

Donde:
CMS = consumo de materia seca, kg/día
PLC4% = producción de leche corregida al 4% de grasa*
BW0.75 = peso vivo metabólico, kg
SL = semana de lactancia

*Para calcular la producción de leche corregida al 4% de grasa se emplea la siguiente expresión matemática: PLC4% = 0.4 * producción de grasa en leche (kg/día) + 15 * producción de leche (l/día)

Conocido el modelo, procedemos a programarlo en el server:

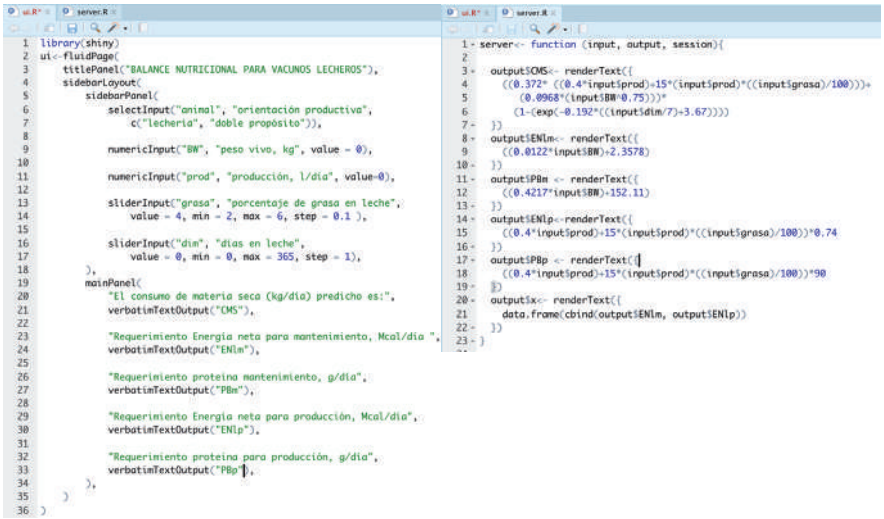
```
4     ((0.372* ((0.4*input$prod)+15*(input$prod)*((input$grasa)/100)))+
5        (0.0968*(input$BW^0.75)))*
6       (1-(exp(-0.192*((input$dim/7)+3.67))))
```

En el código podemos notar que las variables del modelo PLC4%, BW y SL son calculadas tomando como valores las entradas proporcionadas por el usuario en la UI (input$prod, input$grasa, input$BW, input$dim).

Los requerimientos de energía neta para mantenimiento (ENm) y producción (ENp), expresados en Mcal/día, y los requerimientos de proteína bruta para mantenimiento (PBm) y producción (PBp), expresados en g/día, fueron calculados a través de ecuaciones de regresión establecidas de acuerdo con las recomendaciones para vacunos lecheros del NRC (1989).

El código completo de programación de los archivos ui.R, server.R y la vista definitiva de nuestra aplicación se encuentra en las Figuras 6 y 7.

Ahora nuestro aplicativo está listo para ser empleado en el momento que se requiera. La aplicación está alojada en nuestro computador, que actúa como servidor y permite su ejecución cada vez que nosotros



```
library(shiny)
ui<-fluidPage(
    titlePanel("BALANCE NUTRICIONAL PARA VACUNOS LECHEROS"),
    sidebarLayout(
        sidebarPanel(
            selectInput("animal", "orientación productiva",
                c("lecheria", "doble propósito")),

            numericInput("BW", "peso vivo, kg", value = 0),

            numericInput("prod", "producción, l/día", value=0),

            sliderInput("grasa", "porcentaje de grasa en leche",
                value = 4, min = 2, max = 6, step = 0.1 ),

            sliderInput("dim", "días en leche",
                value = 0, min = 0, max = 365, step = 1),
        ),
        mainPanel(
            "El consumo de materia seca (kg/día) predicho es:",
            verbatimTextOutput("CMS"),

            "Requerimiento Energía neta para mantenimiento, Mcal/día ",
            verbatimTextOutput("ENlm"),

            "Requerimiento proteína mantenimiento, g/día",
            verbatimTextOutput("PBm"),

            "Requerimiento Energía neta para producción, Mcal/día",
            verbatimTextOutput("ENlp"),

            "Requerimiento proteína para producción, g/día",
            verbatimTextOutput("PBp"),
        )
    )
)
```

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

  output$CMS<- renderText({
    ((0.372* ((0.4*input$prod)+15*(input$prod)*((input$grasa)/100)))+
      (0.0968*(input$BW^0.75)))*
      (1-(exp(-0.192*((input$dim/7)+3.67))))
  })
  output$ENlm<- renderText({
    ((0.0122*input$BW)+2.3578)
  })
  output$PBm <- renderText({
    ((0.4217*input$BW)+152.11)
  })
  output$ENlp<-renderText({
    ((0.4*input$prod)+15*(input$prod)*((input$grasa)/100))*0.74
  })
  output$PBp <- renderText({
    ((0.4*input$prod)+15*(input$prod)*((input$grasa)/100))*90
  })
  output$x<- renderText({
    data.frame(cbind(output$ENlm, output$ENlp))
  })
}
```

Figura 6. Código de programación de los archivos ui.R y server.R en *Shiny* [captura de pantalla]
Fuente: elaboración propia

como usuarios la invoquemos. Sin embargo, el objetivo de una aplicación es compartir su contenido con múltiples usuarios sin importar la hora, el lugar o el tipo de dispositivo.

En una aplicación web, el proceso usual es que el usuario abre el navegador de su preferencia y señala la dirección de la página. Al especificar la dirección, el navegador se conecta al servidor y obtiene el contenido que se ejecuta en el navegador. Cada vez que el usuario interactúa con la aplicación web se generan nuevas llamadas al servidor, usualmente para solicitar alguna información. El servidor recibe la llamada, la procesa y genera un resultado que luego el navegador del usuario recibe y muestra en pantalla con el formato apropiado (Ramírez, 15 de agosto de 2017).

Para que nuestro aplicativo esté completo necesitamos alojarlo en la nube. *RStudio* ofrece la posibilidad de alojarlo en los servidores de R de

Figura 7. Vista final del aplicativo creado en *Shiny* [captura de pantalla]
Fuente: elaboración propia

forma gratuita. Para ello es necesario remitirse a la página web https://www.shinyapps.io, en la que se debe crear una cuenta y escoger diferentes planes para alojar sus aplicaciones. En nuestro caso escogeremos la opción gratuita que permite mantener simultáneamente cinco aplicaciones con 25 horas de actividad por mes.

El procedimiento es el siguiente:

En la parte superior del editor de *RStudio* encontramos un ícono que nos muestra las opciones de publicación (Figura 8). En este menú hacemos clic en "Publish Application" e inmediatamente se nos solicita conectar nuestro computador con una cuenta de *ShinyApps* previamente creada.

Para realizar la conexión es necesario solicitar un "token" y un código secreto "show secret" (Figura 9).

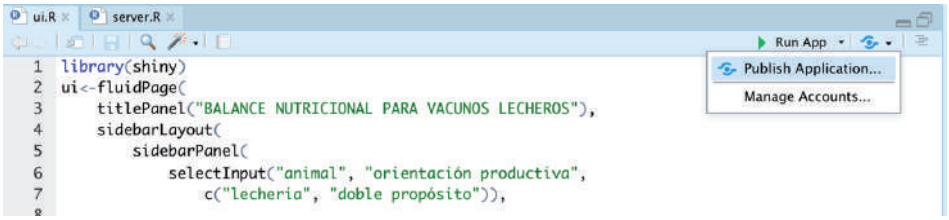


Figura 8. Publicación de la aplicación web [captura de pantalla]
Fuente: elaboración propia

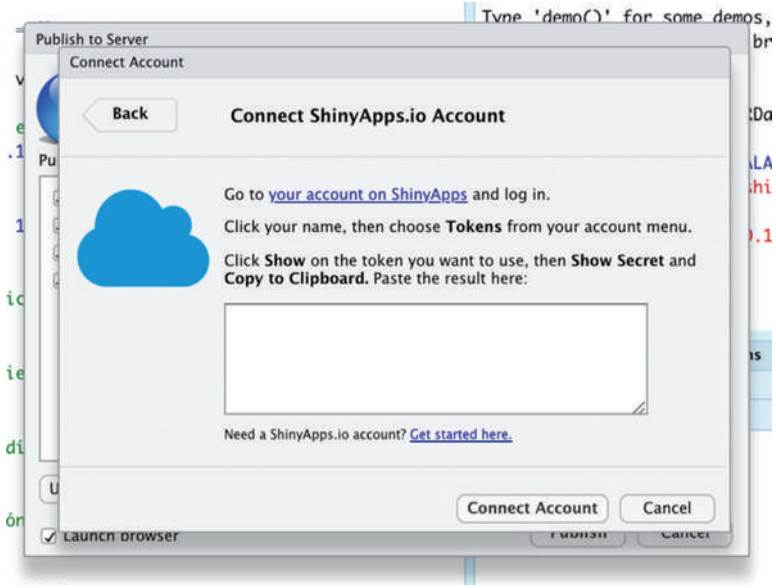


Figura 9. Ventana emergente para registrar token y código secreto [captura de pantalla]
Fuente: elaboración propia

El token y el código secreto se solicitan desde la cuenta en *ShinyApps* haciendo clic en Account > Tokens > Add Tokens (Figura 10).

Al hacer clic en Add Tokens se genera automáticamente un token y el código secreto. Hacemos clic en "Show secret" y en "Copy to clipboard" (Figura 11)

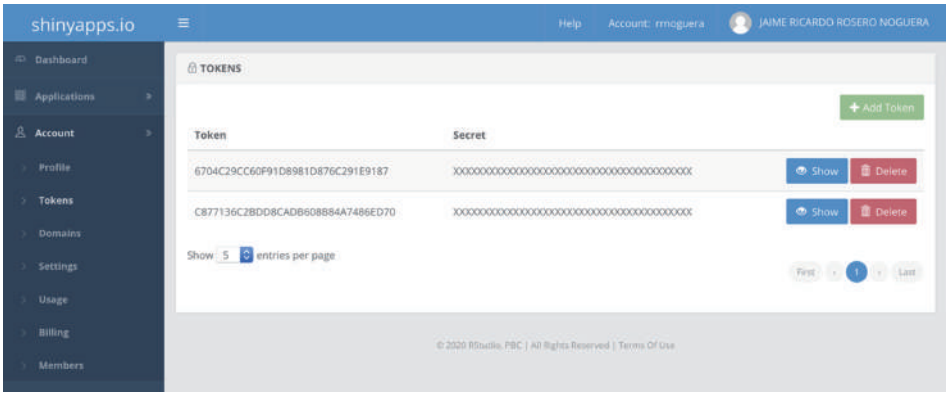


Figura 10. Solicitud token y código secreto en *ShinyApps* [captura de pantalla]
Fuente: elaboración propia

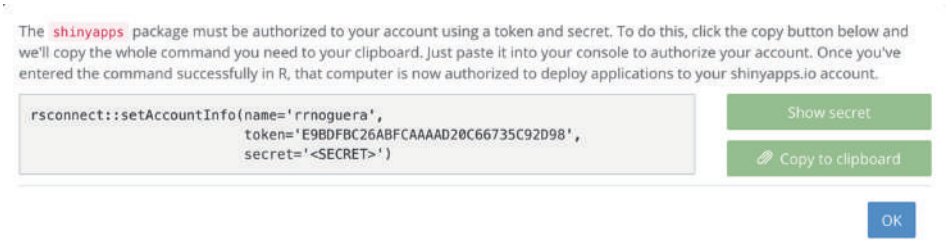


Figura 11. Token y código secreto [captura de pantalla]
Fuente: elaboración propia

Ahora regresamos al menú de *RStudio* donde se nos solicitaba pegar el token y el código secreto (Figura 9). Pegamos la información en el espacio indicado y hacemos clic en "Connect Account" (Figura 12).

Inmediatamente después se despliega un menú que nos muestra los archivos de *RStudio* que migrarán a nuestra cuenta en *ShinyApps* (Figura 13).

Finalmente, hacemos clic en "Publish" y de inmediato la aplicación aparece publicada en nuestra cuenta *ShinyApps* (Figura 14). Ahora nuestra aplicación está alojada en un servidor de *RStudio* y estará disponible para cualquier usuario con conexión a internet y un navegador, simplemente digitando el siguiente link: https://rrnoguera.shinyapps.io/BALANCE/

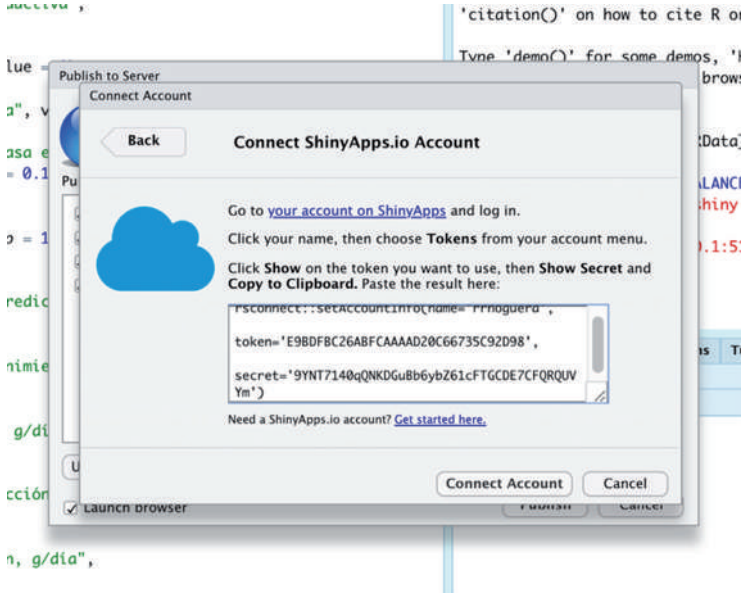


Figura 12. Conexión de la aplicación con la cuenta de *ShinyApps* mediante token y código secreto [captura de pantalla]
Fuente: elaboración propia

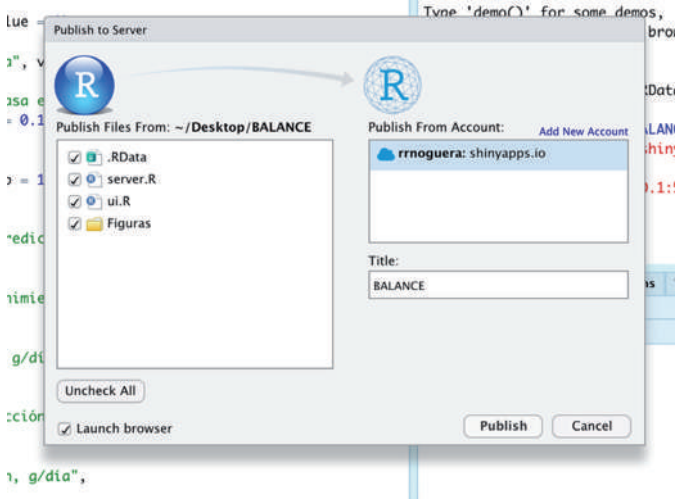


Figura 13. Archivos de *RStudio* que migrarán a cuenta *ShinyApps* [captura de pantalla]
Fuente: elaboración propia

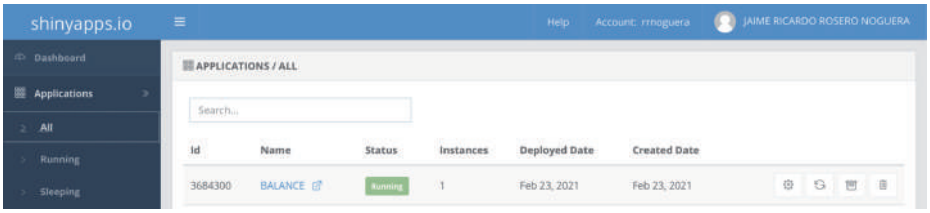


Figura 14. Aplicación web alojada en servidor de *RStudio* [captura de pantalla]
Fuente: elaboración propia

3.2. Ventajas de las aplicaciones web en el proceso de aprendizaje

Entre las ventajas que estas tecnologías ofrecen para profesores y estudiantes podemos destacar su disponibilidad de acceso: la aplicación puede ser consultada en cualquier momento una vez se aloja en un servidor web. Otro punto a destacar es su interactividad, ya que los estudiantes pueden simular diferentes escenarios y evidenciar cómo los cambios en las entradas de información provocan cambios en las salidas, lo cual ayuda a comprender relaciones de dependencia y asociación entre variables y procesos.

Por otra parte, las aplicaciones web son muy prácticas dado que se pueden acceder desde cualquier navegador web sin importar el sistema operativo o tipo de dispositivo empleado. Son fáciles de actualizar y mantener, puesto que no es necesario distribuir o instalar software entre los usuarios potenciales. El consumo de recursos informáticos es bajo porque no es necesario instalar software en nuestro computador y las operaciones y procesos se realizan de forma remota en un servidor externo.

Otro aspecto importante es la portabilidad: las aplicaciones web solo requieren de una conexión a internet y un navegador web para funcionar, se puede acceder a ellas desde cualquier computador, tablet o celu-

lar. Dado que no es necesario instalar software en nuestro dispositivo, el riesgo de ser atacados por virus o malware (software malicioso) es menor.

Finalmente, y tal vez una de las mayores ventajas de las aplicaciones web, es que permiten la colaboración entre usuarios. Al estar alojada en la nube, la aplicación permite el acceso multiusuario, lo cual posibilita compartir y comparar datos y resultados.

3.2. Desventajas de las aplicaciones web

Como toda herramienta, las aplicaciones web también tienen desventajas:

- Las aplicaciones web desaparecen si así lo requiere el desarrollador.
- El usuario no tiene libertad para elegir la versión de la aplicación web que quiere usar, pues siempre emplea la versión más actualizada.
- Las aplicaciones web generalmente ofrecen menos funcionalidades que las aplicaciones de escritorio. Esto se debe a que las funcionalidades que se pueden realizar desde un navegador son más limitadas que las que se realizan directamente desde el sistema operativo.
- La disponibilidad de la aplicación depende de un tercero, que es quien ofrece el servicio de conexión o quien aloja la aplicación en el servidor.

3.3. Experiencias en el empleo de aplicaciones web en el proceso de enseñanza

Antes de relatar nuestra experiencia en el empleo de aplicaciones web, es necesario describir algunas características del estudiante universitario del siglo XXI. La generación actual de estudiantes universitarios

nació y creció en ambientes bombardeados de tecnología. De acuerdo con Ibañez et al. (2008), los estudiantes actuales consideran a la educación como una mercancía que puede ser consumida y adquirida, por lo que esperan que el aprendizaje sea rápido, sencillo y entretenido. Desde esta percepción, el estudiante actual busca recibir la información de forma ágil e inmediata y tiene preferencia por los contenidos interactivos, gráficos, multimedia y colaborativos.

Cuando a los estudiantes del curso de nutrición se les presentó la posibilidad de practicar los conceptos de clase en un aplicativo web, mostraron gran interés y motivación. Argumentaron que este tipo de herramientas les permite mayor flexibilidad en el manejo del tiempo, puesto que pueden acceder a los contenidos en cualquier lugar y horario. Además, manifestaron que dedicaron más horas al estudio, ya que el aplicativo les permitía simular diferentes escenarios y revisar los contenidos cuantas veces desearan. El contenido interactivo de la aplicación les permitió comprender las relaciones de dependencia entre las diferentes variables y fortalecer los conceptos teóricos de la clase.

Como docentes, la oportunidad de ofrecer una nueva herramienta didáctica en el curso de nutrición nos permitió comprender que:

- Los estudiantes actuales necesitan ser partícipes activos de su proceso formativo.
- Es necesario incrementar el número de metodologías que promuevan el aprendizaje activo y colaborativo.
- Las actividades de clase deben acompañar los conceptos teóricos con aplicaciones prácticas que le permitan al estudiante la apropiación del concepto y faciliten su utilización y aplicación práctica en la solución de problemas.

Los aplicativos web interactivos facilitan los métodos de enseñanza centrados en el desarrollo de competencias.

4. Conclusión

El surgimiento de nuevas tecnologías de la información y la comunicación han impactado profundamente a la sociedad humana y el proceso educativo no es ajeno a este cambio. El avance vertiginoso de la tecnología ha propiciado el surgimiento de nuevas formas de educación. Hoy estudiantes y profesores poseen otros escenarios de encuentro e interacción diferentes al aula de clase, en estos escenarios se debate, se comparte información, se crea conocimiento. El reto para los docentes del siglo XXI es adoptar estas herramientas tecnológicas buscando motivar a esta nueva generación de estudiantes en la apropiación de conocimientos, su integración y aplicación práctica orientada a la solución de problemas y centrada en el desarrollo de saberes: saber pensar, saber interpretar, saber desempeñarse y saber actuar en diferentes escenarios.

En nuestra experiencia, la creación de aplicativos web en conjunto con los estudiantes ha demostrado ser un medio útil para que los estudiantes sean agentes activos de su proceso de formación, comprendan más fácilmente procesos biológicos explicados desde la creación de algoritmos, que paso a paso explican el comportamiento de variables complejas.

La crisis generada por la pandemia nos ha enseñado que es posible apoyar los ejes misionales de las universidades con la adopción de las TIC en los procesos educativos, lo cual nos permite superar las barreras de espacio y tiempo, promover el trabajo cooperativo y hacer más facil y equitativo el acceso a la información.

Referencias bibliográficas

Beeley, C. (2013). Web Application Development with R Using Shiny. Birmingham: Packt Publishing.

Gómez, D.S., Molina, M.D., Mulero, J., Nueda, M.J., Pascual, A. (2016). Aplicaciones diseñadas con Shiny: un recurso docente para la enseñanza de la estadística, XIV Jornadas de redes de Investigación en Docencia Universitaria, Alicante. Recuperado de https://web.ua.es/es/ice/jornadas-redes-2016/documentos/tema-2/807250.pdf

Ibáñez, E., Cuesta, M., Taglibaue, R. & Zangaro, M. (2008). La generación actual en la universidad: el impacto de los millennials. V Jornadas de Sociología de la UNLP. Universidad Nacional de La Plata. Facultad de Humanidades y Ciencias de la Educación. Departamento de Sociología. La Plata. Recuperado de https://www.aacademica.org/000-096/261.pdf

Laboratorio de Economía de la Educación -LEE. (2021). Cambios y retos que enfrentaron los docentes durante el cierre de colegios por la pandemia. Pontificia Universidad Javeriana. Recuperado de https:// economiadelaeducacion.org/docs/

Ramírez A. (15 de agosto de 2017). Aplicaciones web en R con Shiny. https://synergy.vision/corpus/shiny/2017-08-15-shiny.html

Resnizky, C. (2015). *Learning Shiny*. Birmingham: Packt Publishing.