
MANUAL DE TÉCNICO ONNOTE

Bran-Bit-Lab & Gabmart1995

Versión 1.0

CARACAS NOVIEMBRE 2021

INTRODUCCIÓN

Dentro de cualquier organización sea grande o pequeña debe poseer cierta dirección de hacia donde se dirige. Que mercado desea impactar, para generar un servicio de calidad y lograr su expansión. Para lograrlo se necesita detenerse un momento y analizar cuáles son los objetivos que persigue la organización, trabajando diligentemente para cumplirlos. Porque no tiene sentido emprender sin tener alguna visión estratégica de lo que se desea lograr transmitir a tus clientes, empleados y proveedores controlados por normativas y procedimientos internos.

Propósito del Documento

El propósito de este documento es promover de un registro o manual técnico especializado acerca de las tecnologías que se utilizaron durante el proceso de desarrollo del producto, el ciclo de vida de la aplicación hasta su despliegue.

Alcance del Documento

El alcance de este documento detalla todos los procesos técnicos que conlleva a la instalación del producto para el entorno de producción.

REQUISITOS DE LA APLICACIÓN ONNOTE

En esta sección se detallan todos los requisitos que se necesitan para que el desarrollador inicie con el desarrollo:

Descripción de la aplicación:

Nombre del proyecto:	OnNote
Tipo de aplicación	Escritorio
Versión:	1,0
FrameWork o Librería de Desarrollo	NodeJS y ElectronJS

Motor de compilación	Electron-Forge e InnoSetup
Lenguaje de desarrollo	HTML, CSS y JavaScript
Gestor de Base de Datos	Mysql o MariaDB
Plataformas soportadas	Multiplataforma
Sistemas Operativos soportados	Windows 7, 8, 8,1, 10 de 32 o 64bits, Debian 9 o superior de 64bits Ubuntu 16.04 o superior de 64bits

Área de desarrollo

Para el área de desarrollo se necesita de las siguientes herramientas instaladas:

- **NodeJS versión 12 o superior:** corresponde a un entorno de desarrollo en el lenguaje de programación JavaScript para el lado del servidor. Para obtenerlo puede consultar: <https://nodejs.org/en/> se encuentra disponible para Linux, Windows y Mac.
- **NPM:** este corresponde al gestor de paquetes de NodeJS, esta herramienta ya viene con la instalación.
- **Editor de código:** cualquier editor de código que permita modificar la sintaxis de JavaScript.
- **Terminal o linea de comandos:** es obligatorio porque permite iniciar el sandbox de desarrollo, iniciar las pruebas y despliegue de la aplicación.
- **Git:** es la herramienta de control de versiones por excelencia, permite obtener una copia del repositorio para trabajar en el área de desarrollo. Disponible para los 3 sistemas operativos, en el enlace: <https://git-scm.com/book/en/v2/Getting-Started-Installing-Git>

Área de Producción

- **Gestor de base de datos MYSQL o MariaBD:** la aplicación necesita de un gestor de Base de Datos, para poder almacenar los datos generados por la aplicativo.
- **Archivo Fuente SQL:** corresponde a la estructura de la BD de la aplicativo, se entrega con una copia del instalador de la aplicacion

Instalación del área de desarrollo:

A continuación se procede a mostrar los pasos para realizar el funcionamiento en el área de desarrollo. Atención los pasos aplican para los 3 sistemas operativos:

Clonar el proyecto desde la fuente:

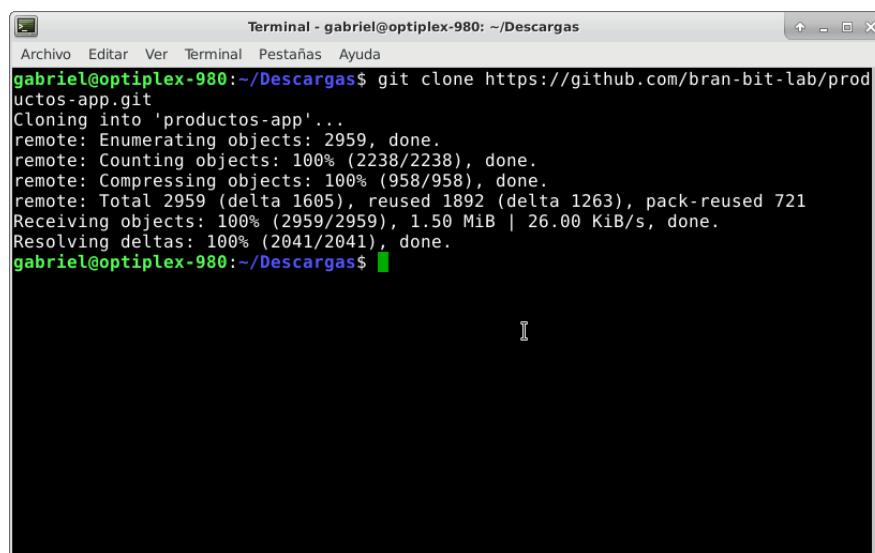
Una vez instaladas las herramientas en la sección anterior se procede a clonar el repositorio desde el servidor desde la siguiente dirección:

<https://github.com/bran-bit-lab/productos-app.git>

Instalado git en tu computadora, en tu espacio de trabajo inicias una consola de comandos y ejecutas el siguiente instrucción:

```
git clone https://github.com/bran-bit-lab/productos-app.git
```

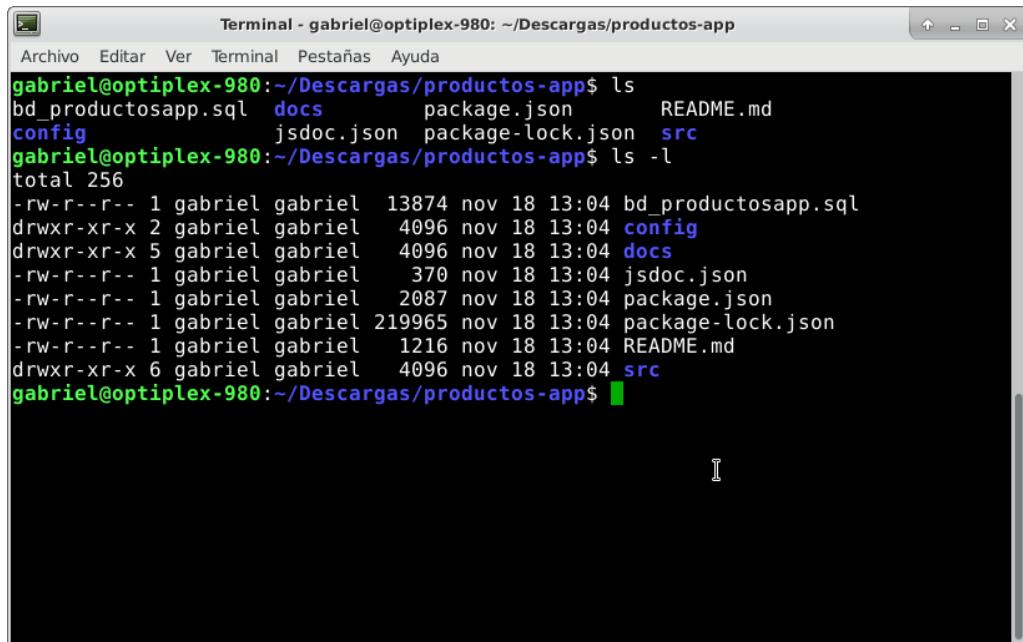
Automáticamente comenzará a descargar el repositorio con los cambios colocados dentro de la rama master, en la misma se encuentran los cambios estables de la aplicación.



The screenshot shows a terminal window titled "Terminal - gabriel@optiplex-980: ~/Descargas". The window contains the following command and its execution:

```
gabriel@optiplex-980:~/Descargas$ git clone https://github.com/bran-bit-lab/productos-app.git
Cloning into 'productos-app'...
remote: Enumerating objects: 2959, done.
remote: Counting objects: 100% (2238/2238), done.
remote: Compressing objects: 100% (958/958), done.
remote: Total 2959 (delta 1605), reused 1892 (delta 1263), pack-reused 721
Receiving objects: 100% (2959/2959), 1.50 MiB | 26.00 KiB/s, done.
Resolving deltas: 100% (2041/2041), done.
gabriel@optiplex-980:~/Descargas$
```

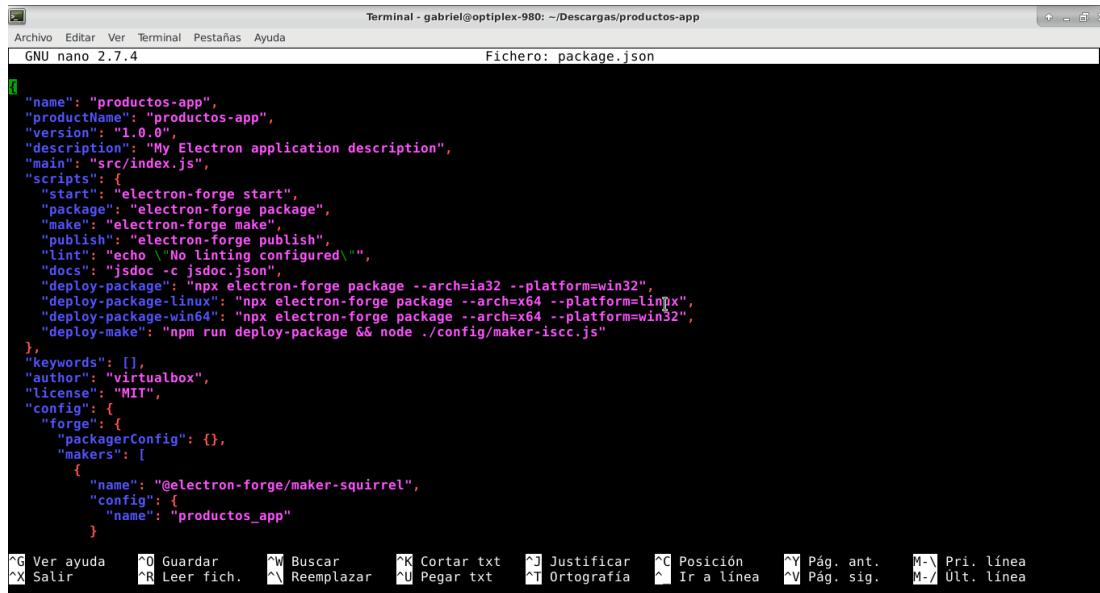
Una vez descargado el proyecto llegara con los siguientes archivos:



```
Terminal - gabriel@optiplex-980: ~/Descargas/productos-app
Archivo Editar Ver Terminal Pestañas Ayuda
gabriel@optiplex-980:~/Descargas/productos-app$ ls
bd_productosapp.sql  docs      package.json    README.md
config                jsdoc.json  package-lock.json  src
gabriel@optiplex-980:~/Descargas/productos-app$ ls -l
total 256
-rw-r--r-- 1 gabriel gabriel 13874 nov 18 13:04 bd_productosapp.sql
drwxr-xr-x 2 gabriel gabriel 4096 nov 18 13:04 config
drwxr-xr-x 5 gabriel gabriel 4096 nov 18 13:04 docs
-rw-r--r-- 1 gabriel gabriel 370 nov 18 13:04 jsdoc.json
-rw-r--r-- 1 gabriel gabriel 2087 nov 18 13:04 package.json
-rw-r--r-- 1 gabriel gabriel 219965 nov 18 13:04 package-lock.json
-rw-r--r-- 1 gabriel gabriel 1216 nov 18 13:04 README.md
drwxr-xr-x 6 gabriel gabriel 4096 nov 18 13:04 src
gabriel@optiplex-980:~/Descargas/productos-app$
```

Construcción de los módulos de desarrollo con NPM:

Para construir los módulos es necesario que tengas una conexión a Internet ya que NPM lee el archivo package.json y se conectará al servidor para obtener las dependencias de desarrollo.



```
Terminal - gabriel@optiplex-980: ~/Descargas/productos-app
Archivo Editar Ver Terminal Pestañas Ayuda
GNU nano 2.7.4
Fichero: package.json
{
  "name": "productos-app",
  "productName": "productos-app",
  "version": "1.0.0",
  "description": "My Electron application description",
  "main": "src/index.js",
  "scripts": {
    "start": "electron-forge start",
    "package": "electron-forge package",
    "make": "electron-forge make",
    "publish": "electron-forge publish",
    "lint": "echo \\No linting configured\\",
    "docs": "jsdoc -c jsdoc.json",
    "deploy-package": "npx electron-forge package --arch=ia32 --platform=win32",
    "deploy-package-linux": "npx electron-forge package --arch=x64 --platform=linux",
    "deploy-package-win64": "npx electron-forge package --arch=x64 --platform=win32",
    "deploy-make": "npm run deploy-package && node ./config/maker-iscc.js"
  },
  "keywords": [],
  "author": "virtualbox",
  "license": "MIT",
  "config": {
    "forge": {
      "packagerConfig": {},
      "makers": [
        {
          "name": "@electron-forge/maker-squirrel",
          "config": {
            "name": "productos_app"
          }
        }
      ]
    }
  }
}
^G Ver ayuda  ^O Guardar  ^W Buscar  ^K Cortar txt  ^J Justificar  ^C Posición  ^Y Pág. ant.  M-\ Pri. línea
^X Salir  ^R Leer fich.  ^A Reemplazar  ^U Pegar txt  ^T Ortografía  ^I Ir a línea  ^V Pág. sig.  M-/ Últ. línea
```

A continuación se necesita correr el comando siguiente:

[npm install](#)

Este comando construye todas las dependencias utilizadas en tu proyecto esta operación puede tardar un poco, dependiendo de la velocidad de la Internet. A continuación se detallaran los comandos básicos que posee la aplicación:

npm start: arranca el sandbox de desarrollo y compila el aplicativo en el entorno permite realizar la creación, edición de código fuente y ejecutar pruebas.

Nota: en Linux para configurar el sandbox, debes otorgar permisos a root para el ejecutable chrome-sandbox con los siguientes comandos como superusuario:

[chown root:root node_modules/electron/dist/chrome-sandbox](#)

[chmod 4755 node_modules/electron/dist/chrome-sandbox](#)

El ultimo parámetro corresponde al directorio del sandbox.

Comandos NPM:

npm start: ejecuta Electron JS en modo de desarrollo, generando las ventanas principales de la aplicación en modo de prueba.

npm run package: permite la empaquetación de los archivos de salida de la aplicación de escritorio, toma el sistema operativo anfitrón y lo empaqueta a esa plataforma. (Este comando es flexible puede compilar para múltiples SO agregando el objetivo de empaquetador, agregando banderas antes de la ejecución del comando).

npm run make: permite la creación de distribuibles para la instalación en el sistema operativo anfitrión. (Este comando es flexible puede compilar para múltiples SO agregando el objetivo de empaquetador, agregando banderas antes de la ejecución del comando).

npm run docs: permite la generación de la documentación técnica a los programadores, generando un directorio de salida llamado docs, puedes visualizar este contenido en el navegador.

Comandos NPM específicos:

npm run deploy-package: es similar a run package, pero empaqueta específicamente para arquitectura Windows 32bits y procesadores i386.

npm run deploy-package-win64: es similar a run package, pero empaqueta específicamente para arquitectura Windows 64bits apuntando a arquitectura modernas.

npm run deploy-package-linux: es similar a run package, pero empaqueta específicamente para arquitectura Linux 64bits.

npm run deploy-make: La finalidad de este comando es crear un ejecutable para 32bits en Windows, debido a que empaquetador que viene con Electron-Forge dejó de ofrecer soporte a esa arquitectura y todavía se sigue utilizando.

Al ejecutar *npm start* Electron comenzará a compilar en el sandbox y el resultado lo muestra por consola y al mismo tiempo crea la primera ventana de la aplicación. Pero en este campo puede que muestre un error debido a que la Base de datos no está configurada. Si muestra un error al momento de conexión pasar a la sección **Configuración de la Base de datos.**

Estructuras de archivos de la aplicación:

En esta sección se detalla la estructura utilizada para la creación del proyecto, en el primer nivel de la aplicación se contienen las siguientes partes:

Visualización del primer nivel.

node_modules: son todos los módulos y dependencias que necesitan para correr la aplicación, en el entorno de desarrollo. Se generan cuando el desarrollador usa “npm install”

config: este directorio contiene las configuraciones de creación de distribuibles de la aplicación.

docs: es la salida de la documentación generado por el comando docs en la sección anterior. Si se vuelve a correr el comando sobrescribe el valor de la carpeta.

src: esta directorio contiene todo el código fuente de la aplicación, los desarrolladores prueban y ejecutan los cambios en esta sección.

.gitignore: este archivo permite que el software git ignorar ciertos archivos para que no se suban al servidor remoto.

package.json y package.lock.json: son los archivos de configuración que utiliza NPM para localizar las dependencias e instalarlas localmente en la carpeta node_modules.

README.md: contiene información sobre el entorno de desarrollo, y las herramientas a utilizar, sirve de guia a los desarrolladores para conocer ciertos aspectos de la aplicación.

bd_productosapp_dev.sql: es el archivo SQL de la estructura de la Base de Datos con datos de prueba, usalo para importar la Base de Datos dentro del gestor MYSQL o MariaDB.

bd_productosapp_prod.sql: es el archivo SQL de la estructura de la Base de Datos con datos para producción, usalo para importar la Base de Datos dentro del gestor MYSQL o MariaDB.

jsdoc.json: es un archivo de configuracion de la documentación que indica la forma de estructuracion de la misma dentro del direcotorio docs.

Estructura del proyecto:

controllers: Esta sección contiene el diseño de todos los controladores de la aplicación permite la comunicación con los servicios externos como la Base de Datos, fluyendo la información entre proceso principal y el renderizado.

database: Esta sección contiene los archivos de parámetros de conexión y las clases necesarias para gestionar las consultas hacia la Base de Datos.

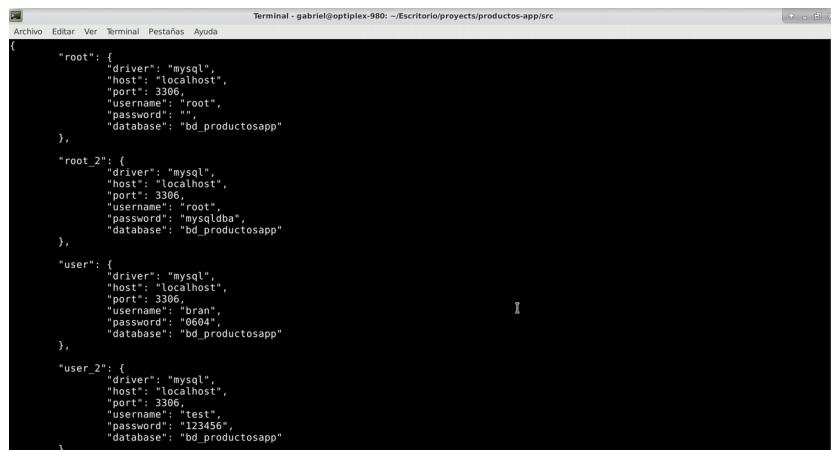
views: contiene todas las vistas del aplicativo, componentes interfaces, modales, formularios, entre otros elementos que necesita el usuario para que pueda interactuar con el aplicación.

util-functions: esta sección incluye aquellas funciones que permiten realizar operaciones repetitivas durante la ejecución de la aplicación, manejo del tiempo, transformación de variables, cálculos matemáticos, etc.

Configuración de la Base de datos:

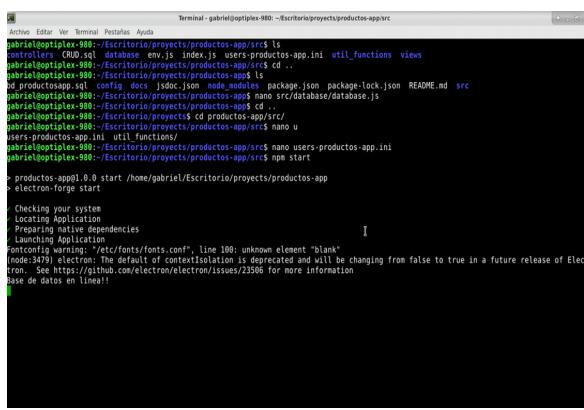
users-products.ini: es el archivo de entorno y acceso a la Base de Datos.

ATENCION: se debe cambiar los valores de acceso y verificar si el servicio está activo, sino la aplicación muestra un error de conexión a la base de datos. Se debe modificar antes de realizar de empaquetar o hacer un distribuible de la aplicación.



```
Terminal - gabriel@optiplex-980: ~/Escritorio/projects/productos-app/src
Archivo  Editar  Ver  Terminal  Pestañas  Ayuda
{
    "root": {
        "driver": "mysql",
        "host": "localhost",
        "port": 3306,
        "username": "root",
        "password": "",
        "database": "bd_productosapp"
    },
    "root_2": {
        "driver": "mysql",
        "host": "localhost",
        "port": 3306,
        "username": "root",
        "password": "mysqldba",
        "database": "bd_productosapp"
    },
    "user": {
        "driver": "mysql",
        "host": "localhost",
        "port": 3306,
        "username": "bran",
        "password": "0604",
        "database": "bd_productosapp"
    },
    "user_2": {
        "driver": "mysql",
        "host": "localhost",
        "port": 3306,
        "username": "test",
        "password": "123456",
        "database": "bd_productosapp"
    }
}.
```

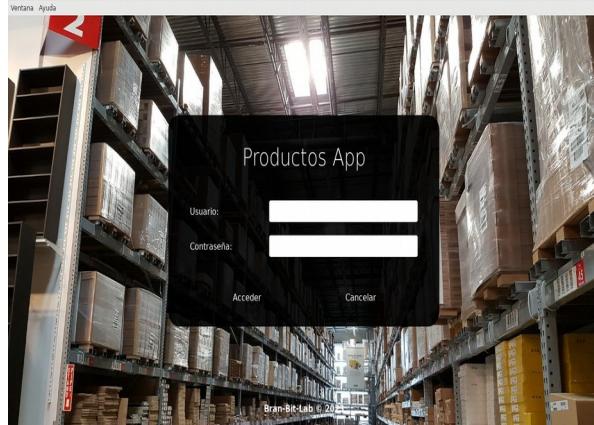
Por defecto el usuario seleccionado es el root, si existe otro usuario de acceso u otra dirección IP, se debe cambiar según los campos especificados. El aplicativo tiene la ventaja de poder conectarse remotamente a la Base de Datos si se encuentra en otro equipo dentro de la red. Se guarda los cambios, se vuelve a ejecutar `npm run start` debe por consola un mensaje: Base de datos en Linea!!



```

Terminal - gabriel@optiplex-900 - Escritorio/projects/productos-app/src
Archivo Editar Ver Terminal Pestañas Ayuda
gabriel@optiplex-900:~/Escritorio/projects/productos-app/src$ controllers CRUD.sql database env.js index.js users-products-app.ini util_functions views
gabriel@optiplex-900:~/Escritorio/projects/productos-app/src$ cd ..
gabriel@optiplex-900:~/Escritorio/projects$ cd productos-app
gabriel@optiplex-900:~/Escritorio/projects/productos-app$ config docs idoc.json node modules package.json package-lock.json README.md src
gabriel@optiplex-900:~/Escritorio/projects/productos-app$ nano src/database/database.js
gabriel@optiplex-900:~/Escritorio/projects/productos-app$ nano src/controllers/CRUD.js
gabriel@optiplex-900:~/Escritorio/projects/productos-app$ nano src/controllers/database.js
gabriel@optiplex-900:~/Escritorio/projects/productos-app$ nano src/controllers/index.js
gabriel@optiplex-900:~/Escritorio/projects/productos-app$ nano src/controllers/users-products-app.js
gabriel@optiplex-900:~/Escritorio/projects/productos-app$ npm start
> productos-app@0.0.0 start /home/gabriel/Escritorio/proyectos/productos-app
> electron-forge start
[ 1] Checking your system
[ 2] Executing system
[ 3] Preparing application
[ 4] Preparing application dependencies
[ 5] Launching Application
Fontconfig warning: "/etc/fonts/fonts.conf", line 180: unknown element "blank"
NodeDeprecationWarning: The default of contextuallyDeprecate is deprecated and will be changing from false to true in a future release of Electron. See https://github.com/electron/electron/issues/23500 for more information
Base de datos en linea!

```



Con eso queda listo el entorno de desarrollo. Para ajustar los cambios debes volver a levantar la aplicación cada vez que el desarrollador realiza un ajuste.

DESPLIEGUE DE LA APLICACIÓN

Electron-Forge: es una herramienta que permite crear, publicar e instalar aplicaciones desarrolladas con Electron. Para el despliegue de la aplicación se debe entender que existen 2 procesos de salida que son la empaquetación y los distribuibles:

La empaquetación o package: es el proceso de generación de los archivos fuentes de salida, la aplicación en ese punto se encuentra lista para ejecutarse sin necesidad de tener el sandbox instalado. Se generá un directorio llamado “out” y dentro de la misma esta el código fuente para la plataforma especificada. Ejecutas el binario dentro del directorio dependiendo del sistema operativo y comenzará a correr la aplicativo.

El distribuible o make: es el proceso de compresión y compilación de los archivos de fuentes de salida. Genera un instalador para el sistema operativo seleccionado, este proceso lleva tiempo dependiendo de las capacidades del equipo. La salida se colocará dentro de la

carpeta out y dentro de ella uno llamado make. La ventaja de esta forma que es altamente portable para instalar en otros equipos.

Para el despliegue no es un proceso complicado solo se necesita ejecutar los comandos específicos para la plataforma donde quieras ejecutar tu aplicativo. A continuación los ejemplos:

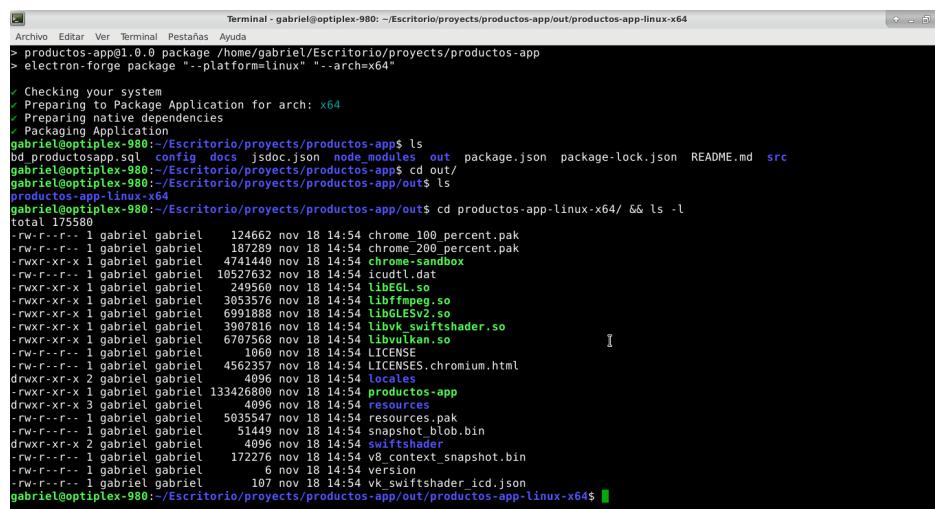
Despliegue multiplataforma: Linux:

Empaquetado:

Para este proceso usamos el comando

```
npm run package -- --platform=linux --arch=x64  
0  
npm run deploy-package-linux
```

Este comando comienza el volcado de la fuente en la plataforma Linux, puede funcionar en cualquier distribución Debian o Ubuntu descritos en los sistemas soportados. Los flag indican que empaqueta para la fuente Linux con cualquier arquitectura de chips que soporte procesamiento de 64bits.



```
Terminal - gabriel@optiplex-980: ~/Escritorio/projects/productos-app/out/productos-app-linux-x64
Archivo Editar Ver Terminal Pestañas Ayuda
> productos-app@1.0.0 package /home/gabriel/Escritorio/projects/productos-app
> electron-forge package "-platform=linux" "-arch=x64"
  Checking your system
  Preparing to Package Application for arch: x64
  Preparing native dependencies
  Packaging Application
gabriel@optiplex-980:~/Escritorio/projects/productos-app$ ls
bd_productos-app_src config docs fsdocs node_modules out package.json package-lock.json README.md src
gabriel@optiplex-980:~/Escritorio/projects/productos-app$ cd out/
gabriel@optiplex-980:~/Escritorio/projects/productos-app/out$ ls
productos-app-linux-x64
gabriel@optiplex-980:~/Escritorio/projects/productos-app/out$ cd productos-app-linux-x64/ && ls -l
total 175580
-rw-r--r-- 1 gabriel gabriel 124662 nov 18 14:54 chrome_100_percent.pak
-rw-r--r-- 1 gabriel gabriel 187289 nov 18 14:54 chrome_200_percent.pak
-rw-r--r-x 1 gabriel gabriel 4741440 nov 18 14:54 chrome-sandbox
-rw-r--r-- 1 gabriel gabriel 10527632 nov 18 14:54 icudtl.dat
-rw-r--r-x 1 gabriel gabriel 249560 nov 18 14:54 libEGL.so
-rw-r--r-x 1 gabriel gabriel 3053576 nov 18 14:54 libffmpeg.so
-rw-r--r-x 1 gabriel gabriel 6991888 nov 18 14:54 libglEsV2.so
-rw-r--r-x 1 gabriel gabriel 3997856 nov 18 14:54 libGLESv3.so
-rw-r--r-x 1 gabriel gabriel 6765568 nov 18 14:54 libuv.so
-rw-r--r-- 1 gabriel gabriel 1060 nov 18 14:54 LICENSE
-rw-r--r-- 1 gabriel gabriel 4562357 nov 18 14:54 LICENSES.chromium.html
drwxr--r-x 2 gabriel gabriel 4096 nov 18 14:54 locales
-rw-r--r-x 1 gabriel gabriel 133426800 nov 18 14:54 productos-app
drwxr--r- 3 gabriel gabriel 4096 nov 18 14:54 resources
-rw-r--r-- 1 gabriel gabriel 5035547 nov 18 14:54 resources.pak
-rw-r--r-- 1 gabriel gabriel 51449 nov 18 14:54 snapshot.blob.bin
drwxr--r-x 2 gabriel gabriel 4096 nov 18 14:54 swiftshader
-rw-r--r-- 1 gabriel gabriel 172276 nov 18 14:54 v8_context_snapshot.bin
-rw-r--r-- 1 gabriel gabriel 6 nov 18 14:54 version
-rw-r--r-- 1 gabriel gabriel 107 nov 18 14:54 vk_swiftshader_icd.json
gabriel@optiplex-980:~/Escritorio/projects/productos-app/out/productos-app-linux-x64$
```

En este punto el técnico puede comprimirlo en zip o mover esa contenido de la carpeta /opt del sistema Linux y generar un lanzador al aplicativo.

Distribuible:

Para crear distribuibles en Linux son archivos con la extensión .deb, para crearlos utilizaremos un compresor dentro de electron-forge llamado @electron-forge/maker-deb, lo creamos con el siguiente comando:

```
npm run make -- --platform=linux --arch=x64 --targets=@electron-forge/maker-deb
```

El flag adicional de targets indica cual es el compresor utilizado por Electron Forge para el sistema operativo seleccionado, estos son los 3 mas utilizados:

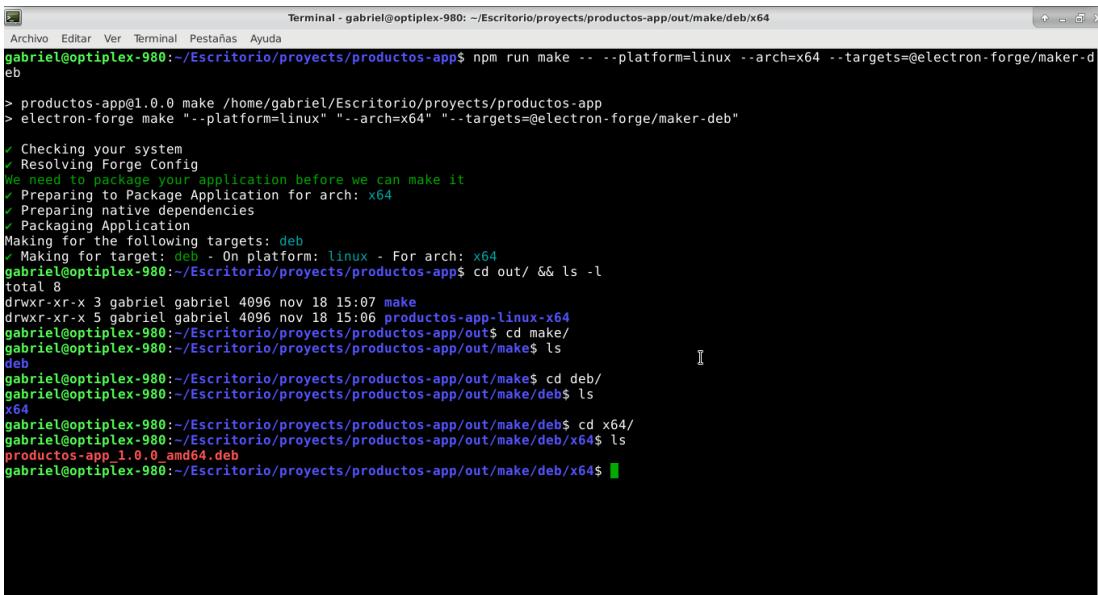
targets:

@electron-forge/maker-deb: Cualquier distribución Linux basada en debian

@electron-forge/maker-rpm: Red-Hat-CentOS

@electron-forge/maker-squirrel: Windows 8, 8.1, 10 de 64bits

Nota: En Windows puedes empaquetar para Linux, pero no crea distribuibles para Linux debido a que el propio kernel no posee extensiones de instrucciones de compilación para otros sistemas. Si lo haces, mostrará un mensaje de error diciendo “*Windows no puede compilar para la plataforma Linux*”. Adicionalmente Electron no posee compilaciones para arquitectura ia32, para Linux.



```
Terminal - gabriel@optiplex-980: ~/Escritorio/proycts/productos-app/out/make/deb/x64
Archivo Editar Ver Terminal Pestañas Ayuda
gabriel@optiplex-980:~/Escritorio/proycts/productos-app$ npm run make -- --platform=linux --arch=x64 --targets=@electron-forge/maker-deb
> productos-app@1.0.0 make /home/gabriel/Escritorio/proycts/productos-app
> electron-forge make "--platform=linux" "--arch=x64" "--targets=@electron-forge/maker-deb"
✓ Checking your system
✓ Resolving Forge Config
We need to package your application before we can make it
✓ Preparing to Package Application for arch: x64
✓ Preparing native dependencies
✓ Packaging Application
Making for the following targets: deb
✓ Making for target: deb - On platform: linux - For arch: x64
gabriel@optiplex-980:~/Escritorio/proycts/productos-app$ cd out/ && ls -l
total 8
drwxr-xr-x 3 gabriel gabriel 4096 nov 18 15:07 make
drwxr-xr-x 5 gabriel gabriel 4096 nov 18 15:06 productos-app-linux-x64
gabriel@optiplex-980:~/Escritorio/proycts/productos-app/out$ cd make/
gabriel@optiplex-980:~/Escritorio/proycts/productos-app/out/make$ ls
deb
gabriel@optiplex-980:~/Escritorio/proycts/productos-app/out/make$ cd deb/
gabriel@optiplex-980:~/Escritorio/proycts/productos-app/out/make/deb$ ls
x64
gabriel@optiplex-980:~/Escritorio/proycts/productos-app/out/make/debs$ cd x64/
gabriel@optiplex-980:~/Escritorio/proycts/productos-app/out/make/deb/x64$ ls
productos-app_1.0.0_amd64.deb
gabriel@optiplex-980:~/Escritorio/proycts/productos-app/out/make/deb/x64$
```

Una vez finalizado obtendremos un ejecutable para instalación de la aplicación a través de la herramienta apt o dpkg de Linux. Se instala a través del comando como superusuario:

```
apt install ./productos-app_1.0.0_amd64.deb
```

o

```
dpkg -i productos-app_1.0.0_amd64.deb
```

Con eso apt o dpkg procede a la instalación del paquete en el Sistema Linux.

Despliegue multiplataforma Windows:

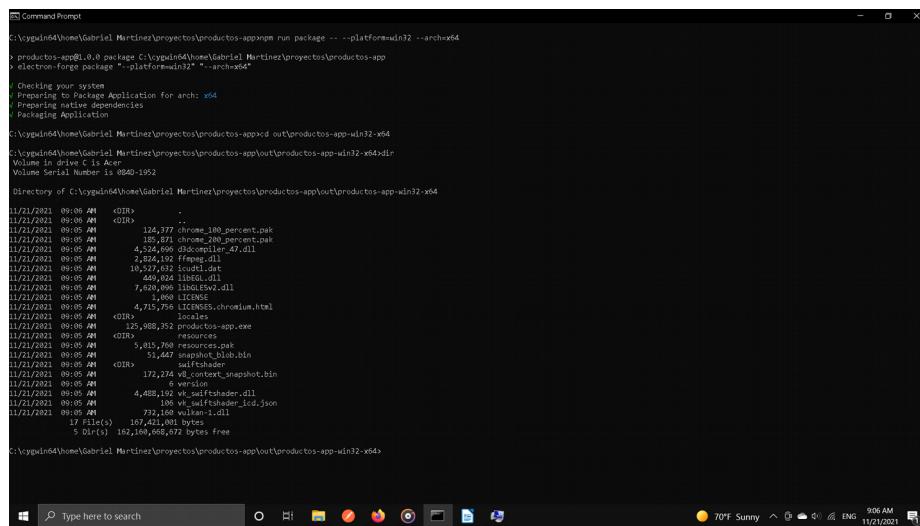
Empaquetado:

Para este proceso usamos el comando para sistemas de 64bits.

```
npm run package -- --platform=win32 --arch=x64
```

o

```
npm run deploy-package-win64
```



The screenshot shows a Windows Command Prompt window with the following output:

```
Command Prompt
C:\cygwin64\home\Gabriel.Martinez\proyectos\productos-app>npm run package -- --platform=win32 --arch=x64
> productos-app@1.0.0 package C:\cygwin64\home\Gabriel.Martinez\proyectos\productos-app
> electron-forge package "--platform=win32" "--arch=x64"
  Checking your system
  Preparing to Package Application for arch: x64
  Preparing native dependencies
  Packaging Application
C:\cygwin64\home\Gabriel.Martinez\proyectos\productos-app>out\productos-app-win32-x64
C:\cygwin64\home\Gabriel.Martinez\proyectos\productos-app>out\productos-app-win32-x64\dir
Volume in drive C is Acer
Volume Serial Number is 0400-1952
Directory of C:\cygwin64\home\Gabriel.Martinez\proyectos\productos-app\out\productos-app-win32-x64

11/21/2021 09:06 AM <DIR> .
11/21/2021 09:06 AM <DIR> ..chromium
11/21/2021 09:05 AM 114,377 chromium_js_percent.pak
11/21/2021 09:05 AM 185,871 chrome_200_percent.pak
11/21/2021 09:05 AM 4,526,698 discopplier_47.dll
11/21/2021 09:05 AM 1,024,000 electron-forge.js
11/21/2021 09:05 AM 10,527,632 iconfile.dat
11/21/2021 09:05 AM 449,024 libEGL.dll
11/21/2021 09:05 AM 7,920,000 libGLESv2.dll
11/21/2021 09:05 AM 1,600 LICENSE
11/21/2021 09:05 AM 4,715,758 LICENSES_chromium.html
11/21/2021 09:05 AM <DIR> node_modules
11/21/2021 09:05 AM 25,988,352 productos-app.exe
11/21/2021 09:05 AM <DIR> resources
11/21/2021 09:05 AM 5,015,768 resources.pak
11/21/2021 09:05 AM 51,441 resources-pak-310-0.bln
11/21/2021 09:05 AM <DIR> swiftshader
11/21/2021 09:05 AM 172,274 v8_context_snapshot.bin
11/21/2021 09:05 AM 1,024 v8_context_snapshot.pak
11/21/2021 09:05 AM 4,458,192 v8_swiftshader.dll
11/21/2021 09:05 AM 108 v8_swiftshader.lcid.json
11/21/2021 09:05 AM 732,552 v8_swiftshader.pak
17 File(s) 167,421,091 bytes
5 Dir(s) 162,169,668,072 bytes free
C:\cygwin64\home\Gabriel.Martinez\proyectos\productos-app\out\productos-app-win32-x64>
```

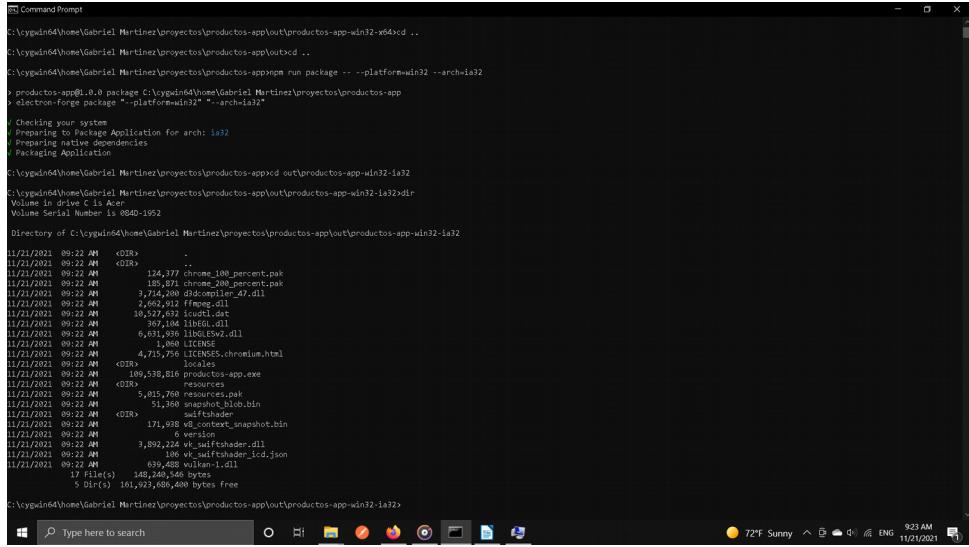
Empaquetacion para arquitectura de 64bits

Para este proceso usamos el comando para sistemas de 32bits.

```
npm run package -- --platform=win32 --arch=ia32
```

0

npm run deploy-package



```
C:\cygwin64\home\Gabriel Martinez\proyectos\productos-app>cd ..  
C:\cygwin64\home\Gabriel Martinez\proyectos\productos-app>cd ..  
C:\cygwin64\home\Gabriel Martinez\proyectos\productos-app>npm run package -- --platform=win32 --arch=ia32  
produtos-app@0.0.0 package C:\cygwin64\home\Gabriel Martinez\proyectos\productos-app  
+ electron-forge package --platform=win32 --arch=ia32  
  Checking your system  
  Preparing to Package Application for arch: ia32  
  Preparing native Dependencies  
  Packaging Application  
C:\cygwin64\home\Gabriel Martinez\proyectos\productos-app>cd out\productos-app\win32-ia32  
C:\cygwin64\home\Gabriel Martinez\proyectos\productos-app>cd ..  
Volume in drive C is A-  
Volume Serial Number is 0840-1992  
Directory of C:\cygwin64\home\Gabriel Martinez\proyectos\productos-app\out\productos-app\win32-ia32  
11/21/2021 09:22 AM <DIR> .  
11/21/2021 09:22 AM <DIR> ..  
11/21/2021 09:22 AM 124,177 chrome_100_percent.pak  
11/21/2021 09:22 AM 139,271 chrome_200_percent.pak  
11/21/2021 09:22 AM 3,114,200 diskcontroller_47.dll  
11/21/2021 09:22 AM 2,652,912 ffmpeg.dll  
11/21/2021 09:22 AM 18,527,632 icudtl.dat  
11/21/2021 09:22 AM 1,080 LICENSE  
11/21/2021 09:22 AM 4,715,756 libchromium.html  
11/21/2021 09:22 AM <DIR> locales  
11/21/2021 09:22 AM 108,538,816 productos-app.exe  
11/21/2021 09:22 AM <DIR> resources  
11/21/2021 09:22 AM 5,015,760 snapshot_blob.pak  
11/21/2021 09:22 AM 51,360 snapshot_blob.bin  
11/21/2021 09:22 AM <DIR> swiftshader  
11/21/2021 09:22 AM 171,938 swiftshader_snapshot.bin  
11/21/2021 09:22 AM 6 versions  
11/21/2021 09:22 AM 3,892,224 vk_swiftshader.dll  
11/21/2021 09:22 AM 186 vk_swiftshader_lcid.json  
11/21/2021 09:22 AM 639,441 vk_swiftshader_lcid.dll  
17 File(s) 148,240,546 bytes  
5 Dir(s) 161,923,689,400 bytes free  
C:\cygwin64\home\Gabriel Martinez\proyectos\productos-app>
```

Empaquetacion para arquitectura de 32bits

Una vez empaquetado puedes moverlo a archivos del programa, y crear un acceso directo al ejecutable del aplicación.

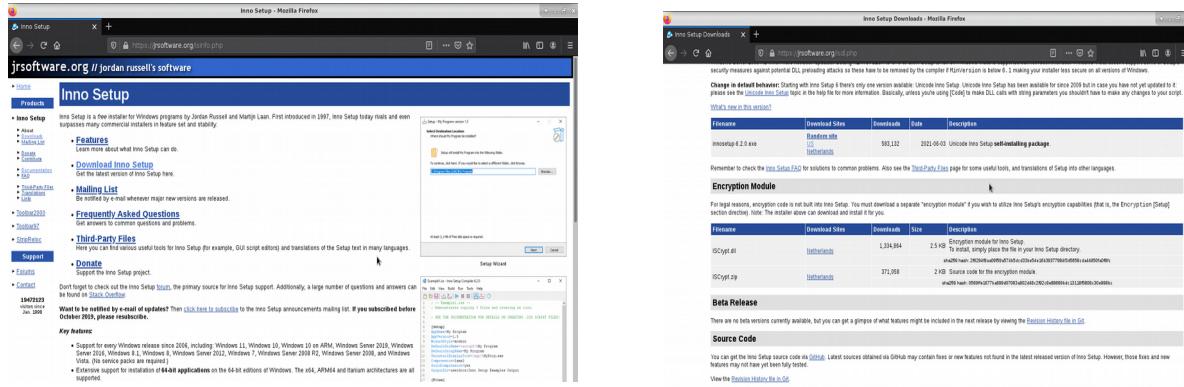
ATENCIÓN: Si posees antivirus, con la finalidad para evitar bloqueos en el sistema, debes permitir que el directorio Productos-App ejecute operaciones sobre el sistema de archivos, debido a que corre un script interpretado al momento de generar los reportes y muchos antivirus lo consideran un programa malicioso.

Distribuible:

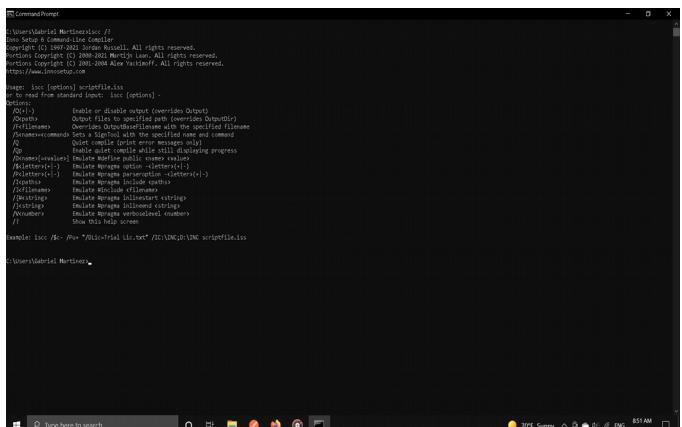
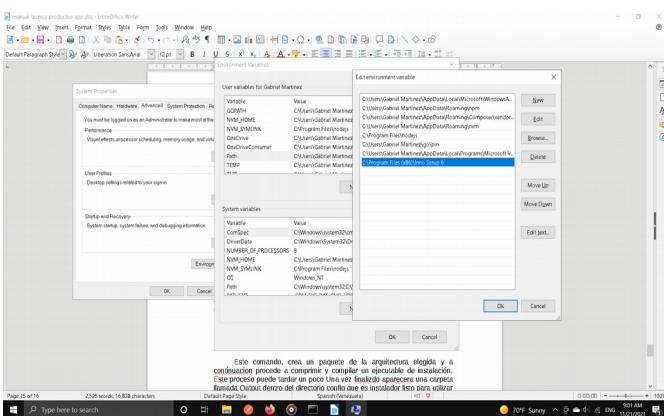
Para crear distribuibles en Windows, los programas son archivos con la extensión .exe existen 2 formas de crear distribuibles. Cabe destacar que el equipo selecciona la primera opción debido a que la forma mas compatible de instalar la aplicación en Windows en sus diferentes versiones.

Forma 1: InnoSetup

InnoSetup: es un programa que facilita a los desarrolladores la creación de instaladores para sus aplicaciones en Microsoft Windows, sin necesidad de estar configurando manualmente el aplicativo, comprime y configura para entornos de producción, listo para utilizar por los técnicos y usuarios. Disponible únicamente para Windows, para más información consultar su sitio: <https://jrsoftware.org/isinfo.php>



Una vez descargado el programa desde el sitio e instalado en el equipo de desarrollo, debemos configurar en las variables de entorno el compilador iscc pueda ser utilizado por la consola de comandos de Windows, y ser ejecutado por un proceso dentro NodeJS para enviar los archivos empaquetados por Electron Forge a InnoSetup para generar el ejecutable a través del compilador.



Configuración de InnoSetup por la consola

Una vez instalado el compilador y configurada en la variable de entorno en el equipo de desarrollo de Windows. Lo siguiente es modificar la ruta dentro del archivo manifiesto con la arquitectura seleccionada dentro de la carpeta config, con la dirección de los archivos

donde se encuentre el mismo. Dentro de la sección files cambiar el valor del campo Source, con la ruta completa donde se encuentran los archivos empaquetados del proyecto.

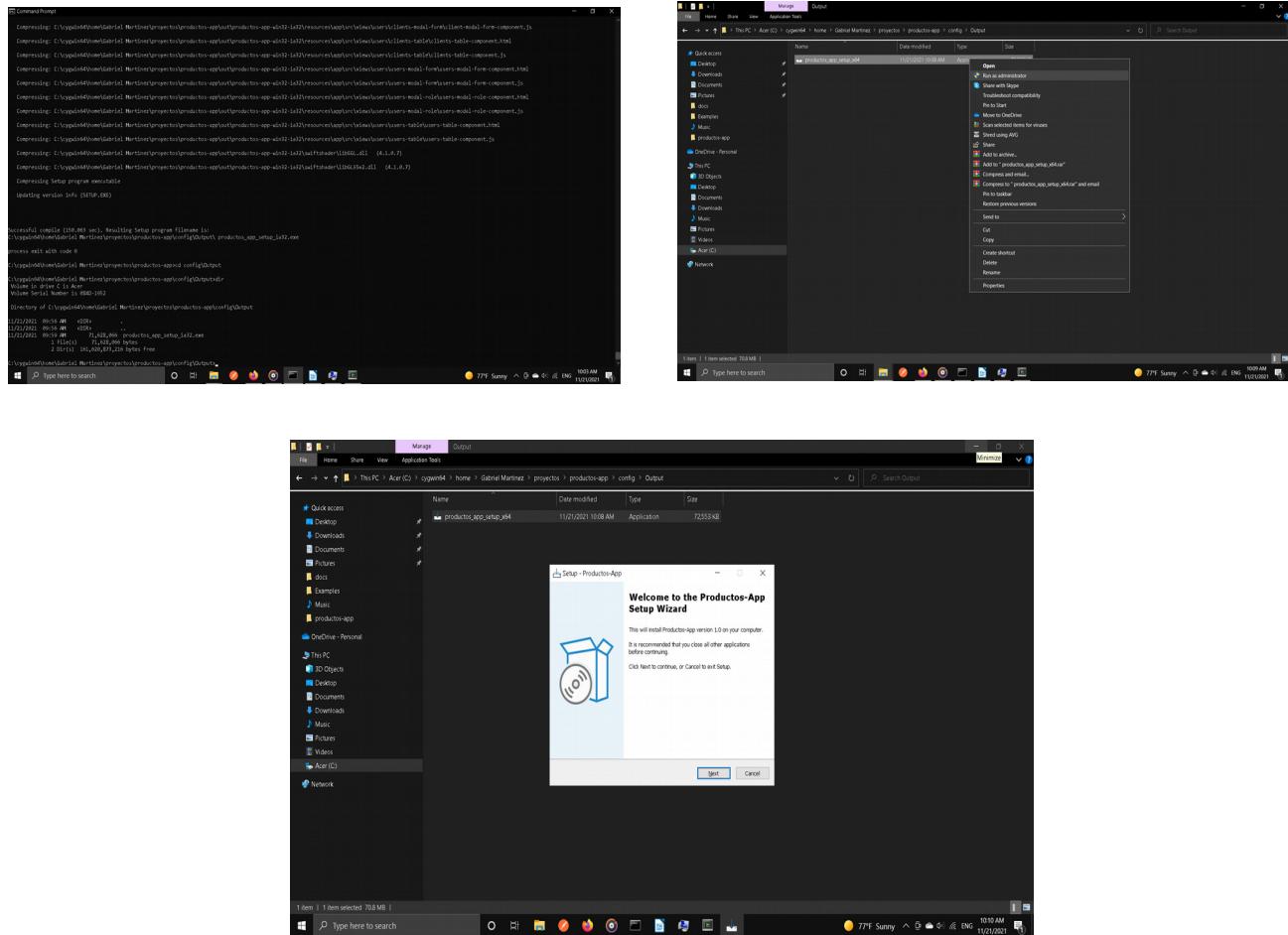
manifiesto_productos-app.iss (32bits)
manifiesto_productos-app_x64.iss (64bits)

Luego ejecutamos en el proyecto el siguiente comando:

npm run deploy-make (32 bits)
o
npm run deploy-make-win64 (64bits)

Este comando, crea un paquete de la arquitectura elegida, seleccionando el archivo por la arquitectura. A continuación procede a comprimir los archivos fuente y compilar el

ejecutable de instalación. Este proceso puede tardar un poco. Una vez finalizado aparecerá una carpeta llamada Output dentro del directorio config del proyecto que es instalador listo para el sistema operativo. El usuario o técnico puede ejecutarlo directamente sobre el equipo con los permisos de administrador.

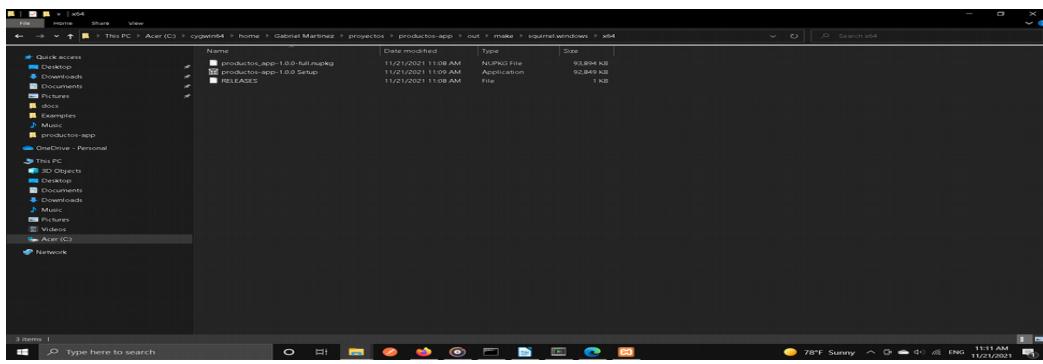


Forma 2: maker-squirrel (Opcional)

Existe otra forma de compilar la aplicación en arquitectura en Windows 64bits, para ello utilizaremos un compresor dentro de electron-forge llamado @electron-forge/maker-squirrel, que crea una salida como en los proyectos en .NET lo creamos con el siguiente comando:

```
npm run make -- --platform=win32 --arch=x64 --targets=@electron-forge/maker-squirrel
```

```
C:\ Command Prompt
C:\cygwind\Home\Gabriel.Martinez\proyectos\productos-app\npm run make -- -platform=win32 -arch=x64 --target=@electron-forge/maker-squirrel
+ products-app@0.0.0 make C:\cygwind\Home\Gabriel.Martinez\proyectos\productos-app
+ electron-forge make -platform=win32" "-arch=x64" "--targets=@electron-forge/maker-squirrel"
  Checking for updates...
    Resolving Forge Config
      Preparing to Package Application for arch x64
  Packaging Application
    Making target: squirrel - On platform: win32 - For arch: x64
  Making target: squirrel - On platform: win32 - For arch: x64
  Volume Serial Number is 084D-1952
  Directory of C:\cygwind\Home\Gabriel.Martinez\proyectos\productos-app\out\make\squirrel.windows\x64
11/21/2021 11:09 AM 404KB squirrel.exe
11/21/2021 11:09 AM 404KB squirrel.exe.manifest
11/21/2021 11:09 AM 99,157,362 products-app@1.0.0-full.mspq
11/21/2021 11:09 AM 3 files(s) 101,224,314 bytes
1 file(s) 101,224,314 bytes free
C:\cygwind\Home\Gabriel.Martinez\proyectos\productos-app\out\make\squirrel.windows\x64
```



Una vez finalizado obtendremos un ejecutable simplemente lo colocamos en una carpeta dentro de los Archivos del Programa y lo Ejecutamos. Un detalle a considerar es que este compresor no funciona en Windows de 32 bits.

CONCLUSIÓN

Para concluir los comandos descritos en el manual permiten al desarrollador o técnico montar con éxito el entorno de desarrollo y despliegue de la aplicación en diferentes sistemas operativos que se utilizan actualmente en el mercado. Cumpliendo con diferentes estándares tecnológicos que permiten la distribución fácil y rápida de la aplicación, facilidad da uso, diseño minimalista, almacenamiento de datos y salida de los mismos, todo con la finalidad de ofrecer una aplicación al cliente pueda gestionar rápidamente sus datos de operaciones sin necesidad de tener servicios de terceros. “Requisito muy importante hoy en día, porque no toda la información de una organización no debe estar en la nube“ mas si son datos sensibles.