Guía de Material Audiovisual — ML-MLOps Portfolio

Fecha: Noviembre 2025
Versión: 4.0 — Windows Edition
Plataforma: Windows 10/11
Herramientas: OBS Studio + Greenshot + DaVinci Resolve
Objetivo: Documentar, planificar y guiar la creación de TODO el material audiovisual para el portafolio

---

Resumen Ejecutivo

Categoría	Total Pendiente	Prioridad
GIFs Demostrativos	4	Alta
Screenshots	4	Media
Video Demo Principal	1	Alta
Videos por Proyecto	3	Media
Thumbnails	3	Baja

Total de elementos pendientes: 15

Mapa de Ubicaciones

Archivos a Crear → Dónde se Referencian

Archivo	Ubicación Física	Referenciado En
`media/gifs/portfolio-demo.gif`	Raíz/media	`README.md` (línea 41)
`media/gifs/bankchurn-preview.gif`	Raíz/media	`README.md` (L73), `BankChurn-Predictor/README.md` (L30), `docs/projects/bankchurn.md` (L8)
`media/gifs/carvision-preview.gif`	Raíz/media	`README.md` (L97), `CarVision-Market-Intelligence/README.md` (L30), `docs/projects/carvision.md` (L8)
`media/gifs/telecom-preview.gif`	Raíz/media	`README.md` (L121), `TelecomAI-Customer-Intelligence/README.md` (L29), `docs/projects/telecom.md` (L8)
`media/screenshots/grafana-dashboard.png`	Raíz/media	`media/README.md` , `docs/operations/`
`media/screenshots/mlflow-experiments.png`	Raíz/media	`media/README.md` , `docs/architecture/`
`media/screenshots/streamlit-carvision.png`	Raíz/media	`media/README.md` , `CarVision-Market-Intelligence/README.md`
`media/screenshots/swagger-ui.png`	Raíz/media	`media/README.md` , `docs/api/rest-apis.md`
Video YouTube/Drive	Externo	`README.md` (L43-44), links en cada proyecto

SECCIÓN 1: GIFs DEMOSTRATIVOS (Prioridad Alta)

Los GIFs son CRÍTICOS porque aparecen en la primera impresión del README principal y de cada proyecto.

1.1 GIF Principal del Portafolio

Especificaciones Técnicas

Campo	Valor
Nombre archivo	`portfolio-demo.gif`
Ruta destino	`media/gifs/portfolio-demo.gif`
Duración	10-15 segundos
Resolución	800x600 o 1280x720
FPS	10-15
Tamaño máximo	< 5 MB

Guion Paso a Paso

```
┌─────────────────────────────────────────────────────────────┐
│                    GUION: PORTFOLIO DEMO GIF                │
└─────────────────────────────────────────────────────────────┘

ESCENA 1: Terminal (0:00 - 0:03)
─────────────────────────────────────────────────────────────
• Terminal limpia con prompt
• Escribir (puede ser acelerado):
  docker-compose -f docker-compose.demo.yml up -d
• Ejecutar y mostrar "Creating bankchurn...", "Creating carvision..."

ESCENA 2: Split Screen - 3 APIs (0:03 - 0:08)
─────────────────────────────────────────────────────────────
• Mostrar 3 ventanas del navegador simultáneamente:
  - localhost:8001/docs (BankChurn)
  - localhost:8002/docs (CarVision)
  - localhost:8003/docs (TelecomAI)
• Pausar 2 segundos para que se vean los 3 Swagger UI

ESCENA 3: Predicción Rápida (0:08 - 0:12)
─────────────────────────────────────────────────────────────
• Click en uno de los endpoints /predict
• "Try it out" → Ejecutar
• Mostrar respuesta JSON exitosa

ESCENA 4: MLflow (0:12 - 0:15)
─────────────────────────────────────────────────────────────
• Cambiar a pestaña MLflow (localhost:5000)
• Mostrar lista de experimentos
• Resaltar métricas (AUC, F1, etc.)
```

Comandos de Preparación (PowerShell)

```powershell
# 1. Preparar entorno
cd "C:\Users\TU_USUARIO\projects\Projects Tripe Ten"

# 2. Levantar servicios con Docker Desktop
docker-compose -f docker-compose.demo.yml up -d

# 3. Esperar a que estén listos (30 segundos)
Start-Sleep -Seconds 30

# 4. Verificar servicios
Invoke-RestMethod -Uri "http://localhost:8001/health" -Method GET
Invoke-RestMethod -Uri "http://localhost:8002/health" -Method GET
Invoke-RestMethod -Uri "http://localhost:8003/health" -Method GET

# 5. Abrir pestañas del navegador
Start-Process "http://localhost:8001/docs"
Start-Process "http://localhost:8002/docs"
Start-Process "http://localhost:8003/docs"
Start-Process "http://localhost:5000"
```

Flujo de Grabación con OBS Studio

```
┌─────────────────────────────────────────────────────────────┐
│                  CONFIGURACIÓN OBS STUDIO PARA GIFs          │
├─────────────────────────────────────────────────────────────┤
│                                                             │
│  PASO 1: Configurar OBS                                     │
│  ─────────────────────────────────────────────────────────  │
│  • Settings → Output → Recording Path: C:\Videos\Portfolio  │
│  • Recording Format: mp4                                    │
│  • Encoder: x264 (o NVENC si tienes GPU NVIDIA)             │
│  • Resolution: 1280x720 (o 800x600 para GIFs más ligeros)   │
│                                                             │
│  PASO 2: Crear Escena                                       │
│  ─────────────────────────────────────────────────────────  │
│  • Sources → + → Window Capture → Seleccionar navegador     │
│  • O usar Display Capture para toda la pantalla             │
│                                                             │
│  PASO 3: Grabar                                             │
│  ─────────────────────────────────────────────────────────  │
│  • Click en "Start Recording"                               │
│  • Ejecutar la demo según el guion                          │
│  • Click en "Stop Recording"                                │
│                                                             │
│  PASO 4: Convertir MP4 → GIF (en DaVinci Resolve o ffmpeg)  │
│  ─────────────────────────────────────────────────────────  │
│  • Ver sección "Comandos Útiles" para conversión            │
│                                                             │
└─────────────────────────────────────────────────────────────┘
```

Conversión MP4 a GIF (PowerShell + ffmpeg)

```powershell
# Instalar ffmpeg con winget (si no lo tienes)
winget install ffmpeg

# Convertir video a GIF
ffmpeg -i "C:\Videos\Portfolio\portfolio-demo.mp4" `
  -vf "fps=12,scale=800:-1:flags=lanczos" `
  -loop 0 `
  "media\gifs\portfolio-demo.gif"

# Optimizar tamaño (opcional, requiere gifsicle)
# Descargar de: https://www.lcdf.org/gifsicle/
gifsicle -O3 --colors 128 "media\gifs\portfolio-demo.gif" -o "media\gifs\portfolio-demo.gif"
```

1.2 GIF BankChurn API

Especificaciones Técnicas

Campo	Valor
Nombre archivo	`bankchurn-preview.gif`
Ruta destino	`media/gifs/bankchurn-preview.gif`
Duración	6-8 segundos
Resolución	800x600
FPS	12

Guion Paso a Paso

```
┌──────────────────────────────────────────────────────────────┐
│                    GUION: BANKCHURN API GIF                  │
└──────────────────────────────────────────────────────────────┘

ESCENA 1: Swagger UI (0:00 - 0:02)
──────────────────────────────────────────────────────────────
• Navegador abierto en http://localhost:8001/docs
• Scroll suave hasta endpoint POST /predict

ESCENA 2: Expandir Endpoint (0:02 - 0:03)
──────────────────────────────────────────────────────────────
• Click en POST /predict para expandir
• Click en "Try it out"

ESCENA 3: Ejecutar Request (0:03 - 0:05)
──────────────────────────────────────────────────────────────
• Datos de ejemplo ya pre-cargados:
  {
    "CreditScore": 650,
    "Geography": "France",
    "Gender": "Female",
    "Age": 40,
    "Tenure": 3,
    "Balance": 60000,
    "NumOfProducts": 2,
    "HasCrCard": 1,
    "IsActiveMember": 1,
    "EstimatedSalary": 50000
  }
• Click en "Execute"

ESCENA 4: Mostrar Respuesta (0:05 - 0:08)
──────────────────────────────────────────────────────────────
• Scroll a Response Body
• Resaltar visualmente:
  {
    "prediction": 0,
    "probability": 0.23,
    "risk_level": "low"
  }
• Pausar 2 segundos en respuesta
```

Texto para Narración (si se convierte en video)

> *"Aquí vemos la API de BankChurn en Swagger UI. Enviamos los datos de un cliente bancario — score de crédito, geografía, edad, balance — y recibimos una predicción de churn con probabilidad del 23%, clasificada como riesgo bajo."*

1.3 GIF CarVision Dashboard

Especificaciones Técnicas

Campo	Valor
Nombre archivo	`carvision-preview.gif`
Ruta destino	`media/gifs/carvision-preview.gif`
Duración	8-10 segundos
Resolución	1280x720
FPS	12

Guion Paso a Paso

```
┌─────────────────────────────────────────────────────────────┐
│                    GUION: CARVISION DASHBOARD GIF           │
├─────────────────────────────────────────────────────────────┤
│                                                             │
│  ESCENA 1: Dashboard Overview (0:00 - 0:02)                 │
│  ─────────────────────────────────────────────────────────  │
│  • Navegador en http://localhost:8501                       │
│  • Mostrar tab "Overview" con KPIs:                         │
│    - Total Vehicles                                         │
│    - Average Price                                          │
│    - Top Brands                                             │
│                                                             │
│  ESCENA 2: Navegar a Price Predictor (0:02 - 0:03)          │
│  ─────────────────────────────────────────────────────────  │
│  • Click en sidebar → "Price Predictor"                     │
│  • Esperar a que cargue el formulario                       │
│                                                             │
│  ESCENA 3: Llenar Formulario (0:03 - 0:06)                  │
│  ─────────────────────────────────────────────────────────  │
│  • Seleccionar campos visualmente (movimientos lentos):     │
│    - Model: "ford f-150"                                    │
│    - Year: 2020                                             │
│    - Odometer: 45000                                        │
│    - Transmission: "automatic"                              │
│    - Fuel: "gas"                                            │
│                                                             │
│  ESCENA 4: Predicción (0:06 - 0:08)                         │
│  ─────────────────────────────────────────────────────────  │
│  • Click en botón "Predict Price"                           │
│  • Mostrar resultado con gauge:                             │
│    - Predicted Price: $32,450                               │
│    - Market Percentile: 72nd                                │
│                                                             │
│  ESCENA 5: Gauge Animation (0:08 - 0:10)                    │
│  ─────────────────────────────────────────────────────────  │
│  • Pausar en el gauge de percentil                          │
│  • Mostrar contexto de mercado                              │
│                                                             │
└─────────────────────────────────────────────────────────────┘
```

Texto para Narración

> *"Este es el dashboard de CarVision. En la sección Overview vemos las métricas del mercado. Navegamos a Price Predictor, ingresamos los datos de un Ford F-150 2020 con 45 mil millas, y obtenemos una predicción de precio de $32,450, ubicándolo en el percentil 72 del mercado."*

Tips de Grabación (Windows)

- **Usar cursor grande**:
  - Windows 10/11: `Configuración → Accesibilidad → Puntero del mouse → Tamaño: Grande`

- O: `Configuración → Dispositivos → Mouse → Opciones adicionales → Punteros → Esquema: Windows Estándar (extra grande)`

- **Resaltar cursor**: En OBS, puedes añadir un efecto de resaltado de cursor
- **Movimientos lentos y deliberados**: El espectador debe poder seguir cada acción
- **Evitar parpadeos**: No cambiar de ventana rápidamente
- **Limpiar escritorio**: Ocultar iconos y usar fondo neutro

1.4 GIF TelecomAI API

Especificaciones Técnicas

Campo	Valor
Nombre archivo	`telecom-preview.gif`
Ruta destino	`media/gifs/telecom-preview.gif`
Duración	6-8 segundos
Resolución	800x600
FPS	12

Guion Paso a Paso

```
┌─────────────────────────────────────────────────────────────────┐
│                      GUION: TELECOMAI API GIF                   │
└─────────────────────────────────────────────────────────────────┘

OPCIÓN A: Terminal (más técnico)
═══════════════════════════════════════════════════════════════

ESCENA 1: Terminal Limpia (0:00 - 0:01)
─────────────────────────────────────────────────────────────────
• Terminal con tema oscuro
• Prompt visible: ~/projects $

ESCENA 2: Escribir Comando (0:01 - 0:04)
─────────────────────────────────────────────────────────────────
• Escribir (puede ser acelerado 2x):
  curl -X POST "http://localhost:8003/predict" \
    -H "Content-Type: application/json" \
    -d '{
      "calls": 50,
      "minutes": 500,
      "messages": 100,
      "mb_used": 20000
    }'

ESCENA 3: Ejecutar (0:04 - 0:05)
─────────────────────────────────────────────────────────────────
• Presionar Enter
• Ver respuesta JSON:
  {
    "prediction": 1,
    "probability": 0.78,
    "recommendation": "upgrade_recommended"
  }

ESCENA 4: Resaltar Resultado (0:05 - 0:08)
─────────────────────────────────────────────────────────────────
• Si usas asciinema: añadir marcador visual
• Pausar en "recommendation": "upgrade_recommended"

─────────────────────────────────────────────────────────────────

OPCIÓN B: Swagger UI (más visual)
═══════════════════════════════════════════════════════════════
• Similar a BankChurn: Swagger → Try it out → Execute → Response
```

Texto para Narración

> *"TelecomAI analiza el comportamiento del usuario: 50 llamadas, 500 minutos, 100 mensajes y 20GB de datos. El modelo recomienda upgrade al plan Ultra con 78% de confianza."*

Grabación con OBS Studio (Windows)

```
┌─────────────────────────────────────────────────────────────────┐
│                   GRABACIÓN TELECOMAI EN WINDOWS                │
├─────────────────────────────────────────────────────────────────┤
│                                                                 │
│  OPCIÓN 1: PowerShell/Terminal con OBS                          │
│  ─────────────────────────────────────                          │
│  • Abrir Windows Terminal o PowerShell                          │
│  • En OBS: Sources → Window Capture → Seleccionar terminal      │
│  • Usar tema oscuro en terminal (mejor contraste)               │
│  • Ejecutar curl con Invoke-RestMethod                          │
│                                                                 │
│  OPCIÓN 2: Swagger UI (recomendado para demos visuales)         │
│  ─────────────────────────────────────                          │
│  • Más visual y fácil de seguir para el espectador              │
│  • Similar a BankChurn: Swagger → Try it out → Execute → Response │
│                                                                 │
└─────────────────────────────────────────────────────────────────┘
```

```powershell
# Comando curl en PowerShell para TelecomAI
$body = @{
    calls = 50
    minutes = 500
    messages = 100
    mb_used = 20000
} | ConvertTo-Json

Invoke-RestMethod -Uri "http://localhost:8003/predict" `
    -Method POST `
    -ContentType "application/json" `
    -Body $body
```

SECCIÓN 2: SCREENSHOTS (Prioridad Media)

2.1 Screenshot Grafana Dashboard

Campo	Valor
Archivo	`media/screenshots/grafana-dashboard.png`
Resolución	1280x800 px
Formato	PNG

Contenido a capturar: - Dashboard de Grafana mostrando: - Request rate de las 3 APIs - Latencia P95 - Error rate - Us de memoria/CPU

Preparación (PowerShell):

```powershell
# 1. Iniciar stack con monitoreo
docker-compose -f docker-compose.demo.yml --profile monitoring up -d

# 2. Generar tráfico para métricas
$body =
'{"CreditScore":650,"Geography":"France","Gender":"Female","Age":40,"Tenure":3,"Balance":60000,"NumOfProducts":2,"HasCrCard":1,"IsActiveM
ember":1,"EstimatedSalary":50000}'
for ($i = 1; $i -le 50; $i++) {
    Invoke-RestMethod -Uri "http://localhost:8001/predict" -Method POST -ContentType "application/json" -Body $body
    Start-Sleep -Milliseconds 500
}

# 3. Acceder a Grafana: http://localhost:3000 (admin/admin)
Start-Process "http://localhost:3000"
```

Captura con Greenshot:

```
┌─────────────────────────────────────────────────────────────┐
│                    CAPTURA CON GREENSHOT                    │
├─────────────────────────────────────────────────────────────┤
│                                                             │
│  1. Presionar Print Screen (o atajo configurado en Greenshot)
│  2. Seleccionar área del dashboard de Grafana               │
│  3. En el editor de Greenshot:                              │
│     • Añadir anotaciones si es necesario                    │
│     • Recortar bordes innecesarios                          │
│  4. Guardar como: media\screenshots\grafana-dashboard.png   │
│                                                             │
└─────────────────────────────────────────────────────────────┘
```

2.2 Screenshot MLflow Experiments

Campo	Valor
Archivo	`media/screenshots/mlflow-experiments.png`
Resolución	1280x800 px

Contenido a capturar: - Vista de experimentos en MLflow - Al menos 2-3 runs visibles - Métricas: AUC, F1, Accuracy Parámetros del modelo

Preparación (PowerShell):

```powershell
# 1. Ejecutar entrenamiento para generar runs
cd BankChurn-Predictor
python main.py --mode train

# 2. Abrir MLflow UI
mlflow ui --port 5000

# 3. Abrir en navegador
Start-Process "http://localhost:5000"
```

Captura con Greenshot: Presionar `Print Screen` → Seleccionar área → Guardar en `media\screenshots\`

2.3 Screenshot Streamlit CarVision

Campo	Valor
Archivo	`media/screenshots/streamlit-carvision.png`
Resolución	1280x900 px

Contenido: - Dashboard Streamlit con la tab “Overview” visible - KPIs en la parte superior - Gráfico de distribución d precios - Sidebar visible

Preparación (PowerShell):

```powershell
cd CarVision-Market-Intelligence
streamlit run app/streamlit_app.py

# Se abrirá automáticamente en http://localhost:8501
```

Captura con Greenshot: `Print Screen` → Seleccionar dashboard completo → Guardar

---

2.4 Screenshot Swagger UI

Campo	Valor
Archivo	`media/screenshots/swagger-ui.png`
Resolución	1280x800 px

Contenido: - Swagger UI de cualquiera de las 3 APIs - Mostrar todos los endpoints expandidos - Badge de versión visible Esquemas de request/response visibles

---

SECCIÓN 3: VIDEO DEMO PRINCIPAL (Prioridad Alta)

3.1 Especificaciones

Campo	Valor
Duración	2-4 minutos
Resolución	1920x1080 (1080p) mínimo
Audio	Narración clara, sin música de fondo
Formato	MP4 (H.264)
Hosting	YouTube (unlisted) o Google Drive

3.2 Guion Completo con Tiempos

Tiempo	Sección	Contenido
0:00-0:15	Intro	Título, nombre, “ML-MLOps Portfolio”
0:15-0:45	Overview	Estructura del repo, 3 proyectos, CI verde
0:45-1:15	Docker Compose	`docker-compose up --build`, servicios iniciando
1:15-1:45	BankChurn Demo	API request, respuesta, explicación breve
1:45-2:15	CarVision Demo	Dashboard Streamlit, predicción interactiva
2:15-2:45	TelecomAI Demo	API request, recomendación de plan
2:45-3:15	MLflow	Experimentos, métricas, comparación de runs
3:15-3:30	Monitoring	Prometheus/Grafana (si aplica)
3:30-4:00	Cierre	CI/CD pipeline, badges, contacto

3.3 Guion Detallado

```
┌──────────────────────────────────────────────────────────────┐
│                  GUION: VIDEO DEMO PRINCIPAL                 │
```

```
                        Duración: ~3 minutos
═══════════════════════════════════════════════════════════════════════

INTRO (0:00 - 0:15)
═══════════════════════════════════════════════════════════════════════

[VISUAL]: Pantalla de título con tu nombre y "ML-MLOps Portfolio"

[NARRACIÓN]:
"Hola, soy [Tu Nombre]. Este es mi portafolio de Machine Learning
y MLOps, con tres proyectos production-ready que demuestran
el ciclo completo de ML: desde entrenamiento hasta deployment."

═══════════════════════════════════════════════════════════════════════
OVERVIEW DEL REPO (0:15 - 0:45)
═══════════════════════════════════════════════════════════════════════

[VISUAL]: GitHub/IDE mostrando estructura de carpetas

[NARRACIÓN]:
"El repositorio está organizado como un monorepo con tres proyectos:
BankChurn Predictor para clasificación de churn bancario,
CarVision para predicción de precios de vehículos,
y TelecomAI para recomendación de planes telefónicos.

Cada proyecto tiene su propia estructura modular con src/, tests/,
configs/, y Dockerfile. Todo integrado con un pipeline CI/CD unificado."

[VISUAL]: Mostrar badge de CI verde en README

═══════════════════════════════════════════════════════════════════════
DOCKER COMPOSE (0:45 - 1:15)
═══════════════════════════════════════════════════════════════════════

[VISUAL]: Terminal ejecutando docker-compose

[NARRACIÓN]:
"Para la demo, usamos Docker Compose que levanta los tres servicios
simultáneamente. Un solo comando:"

[COMANDO]: docker-compose -f docker-compose.demo.yml up -d

[NARRACIÓN]:
"En segundos tenemos BankChurn en el puerto 8001, CarVision en 8002,
TelecomAI en 8003, y MLflow en el 5000."

═══════════════════════════════════════════════════════════════════════
DEMO BANKCHURN (1:15 - 1:45)
═══════════════════════════════════════════════════════════════════════

[VISUAL]: Swagger UI de BankChurn

[NARRACIÓN]:
"Empezamos con BankChurn. Esta API predice la probabilidad de que
un cliente abandone el banco. Enviamos datos como credit score,
geografía, edad y balance..."

[ACCIÓN]: Ejecutar predicción en Swagger

[NARRACIÓN]:
"...y recibimos la predicción con probabilidad y nivel de riesgo.
Este cliente tiene 23% de probabilidad de churn, clasificado como
riesgo bajo. El modelo tiene un AUC de 0.85."

═══════════════════════════════════════════════════════════════════════
DEMO CARVISION (1:45 - 2:15)
═══════════════════════════════════════════════════════════════════════

[VISUAL]: Dashboard Streamlit

[NARRACIÓN]:
"CarVision tiene dos interfaces: una API REST y este dashboard
interactivo en Streamlit. En la sección Overview vemos métricas
```

```
del mercado de vehículos."

[ACCIÓN]: Navegar a Price Predictor, llenar formulario

[NARRACIÓN]:
"En el Price Predictor ingresamos las características del vehículo:
un Ford F-150 2020 con 45 mil millas... y obtenemos una predicción
de $32,450, posicionándolo en el percentil 72 del mercado.
El modelo tiene un R² de 0.77."

═══════════════════════════════════════════════════════════════
DEMO TELECOMAI (2:15 - 2:45)
═══════════════════════════════════════════════════════════════

[VISUAL]: Terminal o Swagger

[NARRACIÓN]:
"TelecomAI analiza el comportamiento del usuario para recomendar
el plan óptimo. Enviamos métricas de uso: llamadas, minutos,
mensajes y datos móviles..."

[ACCIÓN]: Ejecutar predicción

[NARRACIÓN]:
"...y el modelo recomienda upgrade al plan Ultra con 78% de confianza.
Usa un VotingClassifier ensemble con AUC de 0.84."

═══════════════════════════════════════════════════════════════
MLFLOW & EXPERIMENTOS (2:45 - 3:15)
═══════════════════════════════════════════════════════════════

[VISUAL]: MLflow UI

[NARRACIÓN]:
"Todos los experimentos se trackean con MLflow. Aquí vemos los runs
de entrenamiento con sus métricas, parámetros y artefactos.
Esto facilita la comparación y reproducibilidad."

[ACCIÓN]: Mostrar comparación de runs, métricas

═══════════════════════════════════════════════════════════════
CIERRE (3:15 - 3:30)
═══════════════════════════════════════════════════════════════

[VISUAL]: GitHub Actions con CI verde

[NARRACIÓN]:
"Todo está integrado con CI/CD en GitHub Actions: tests automáticos,
cobertura de código, security scanning con Trivy y Bandit.

Gracias por ver. El código está disponible en mi GitHub.
Los links están en la descripción."

[VISUAL]: Pantalla final con:
- GitHub: github.com/DuqueOM/ML-MLOps-Portfolio
- LinkedIn: linkedin.com/in/duqueom
```

3.4 Tips de Producción

1. **Audio**: Usar micrófono USB, grabar en ambiente silencioso
2. **Terminal**: Tema oscuro, fuente 16-18pt
3. **Navegador**: Perfil limpio sin bookmarks personales
4. **Practicar**: 2-3 ensayos antes de grabar
5. **Edición**: Cortar pausas largas, mantener ritmo fluido

SECCIÓN 4: VIDEOS INDIVIDUALES (Opcional)

4.1 Video BankChurn (45-60 segundos)

```
┌─────────────────────────────────────────────────────────────┐
│                    GUION: VIDEO BANKCHURN                   │
├─────────────────────────────────────────────────────────────┤
│                                                             │
│  (0:00 - 0:10) INTRO                                        │
│  "BankChurn Predictor: sistema de predicción de churn para banca.
│  Modelo ensemble con VotingClassifier, AUC de 0.85."        │
│                                                             │
│  (0:10 - 0:40) DEMO                                         │
│  • Mostrar Swagger UI                                       │
│  • Ejecutar predicción con datos de ejemplo                 │
│  • Explicar respuesta: prediction, probability, risk_level  │
│                                                             │
│  (0:40 - 0:60) FEATURES                                     │
│  • Mencionar: "Docker ready, MLflow tracking, 77% test coverage"
│  • Mostrar badge de CI verde brevemente                     │
│                                                             │
└─────────────────────────────────────────────────────────────┘
```

4.2 Video CarVision (60-90 segundos)

```
┌─────────────────────────────────────────────────────────────┐
│                    GUION: VIDEO CARVISION                   │
├─────────────────────────────────────────────────────────────┤
│                                                             │
│  (0:00 - 0:15) INTRO                                        │
│  "CarVision Market Intelligence: plataforma de valuación de vehículos
│  con dashboard interactivo y API REST."                     │
│                                                             │
│  (0:15 - 0:45) DASHBOARD TOUR                               │
│  • Overview: KPIs, distribución de precios                  │
│  • Market Analysis: insights por marca                      │
│  • Model Metrics: RMSE, R², MAPE                            │
│                                                             │
│  (0:45 - 1:15) PRICE PREDICTOR                              │
│  • Llenar formulario con Ford F-150                         │
│  • Mostrar predicción y gauge de percentil                  │
│  • Explicar interpretación de resultados                    │
│                                                             │
│  (1:15 - 1:30) CIERRE                                       │
│  "96% test coverage, dual interface: API + Dashboard"       │
│                                                             │
└─────────────────────────────────────────────────────────────┘
```

4.3 Video TelecomAI (45-60 segundos)

```
┌─────────────────────────────────────────────────────────────┐
│                    GUION: VIDEO TELECOMAI                   │
├─────────────────────────────────────────────────────────────┤
│                                                             │
│  (0:00 - 0:10) INTRO                                        │
│  "TelecomAI: inteligencia de cliente para telecomunicaciones.
│  Predice qué usuarios deben recibir upgrade de plan."       │
│                                                             │
│  (0:10 - 0:40) DEMO API                                     │
│  • Mostrar Swagger o curl en terminal                       │
│  • Enviar métricas de uso: calls, minutes, messages, mb_used
│  • Mostrar respuesta con recommendation                     │
│                                                             │
│  (0:40 - 0:60) ARQUITECTURA                                 │
│  • "VotingClassifier: LogReg + RF + XGBoost"                │
│  • "AUC 0.84, precision 81.7%"                              │
│  • "Docker ready, 96% test coverage"                        │
│                                                             │
└─────────────────────────────────────────────────────────────┘
```

🖼️ SECCIÓN 5: THUMBNAILS (Prioridad Baja)

Si subes videos a YouTube, necesitarás thumbnails atractivos.

Archivo	Contenido sugerido
`media/thumbnails/portfolio-thumb.png`	Logo + “ML-MLOps Portfolio” + badges
`media/thumbnails/bankchurn-thumb.png`	Icono de banco + métricas AUC
`media/thumbnails/carvision-thumb.png`	Icono de auto + dashboard preview

Herramientas gratuitas: Canva, Figma, GIMP

🪟 SECCIÓN 6: HERRAMIENTAS WINDOWS (Stack Recomendado)

Este es el stack optimizado para Windows que usaremos:

OBS Studio — Grabación de Video

Campo	Valor
Descarga	obsproject.com
Costo	Gratis y Open Source
Uso principal	Grabar pantalla para videos y GIFs
Formato salida	MP4 (H.264)

```
┌─────────────────────────────────────────────────────────────┐
│                  CONFIGURACIÓN INICIAL OBS STUDIO           │
├─────────────────────────────────────────────────────────────┤
│                                                             │
│  1. INSTALACIÓN                                             │
│     • Descargar desde obsproject.com                        │
│     • Ejecutar instalador → Next → Next → Install           │
│                                                             │
│  2. PRIMERA CONFIGURACIÓN                                   │
│     • Al abrir, ejecutar "Auto-Configuration Wizard"        │
│     • Seleccionar: "Optimize just for recording"            │
│     • Base Resolution: 1920x1080                            │
│     • FPS: 30 (o 60 si tu PC lo soporta)                    │
│                                                             │
│  3. AJUSTES PARA DEMOS                                      │
│     • Settings → Output → Recording:                        │
│       - Recording Path: C:\Videos\Portfolio                 │
│       - Recording Format: mp4                               │
│       - Encoder: x264 (o NVENC si tienes NVIDIA)            │
│       - Recording Quality: High Quality                     │
│                                                             │
│  4. CREAR ESCENAS                                           │
│     • Scenes → + → "Demo Portfolio"                         │
│     • Sources → + → Window Capture → Navegador              │
│     • Sources → + → Audio Output Capture (para sonido del sistema)
│                                                             │
│  5. HOTKEYS (Recomendados)                                  │
│     • Settings → Hotkeys:                                   │
│       - Start Recording: Ctrl+Shift+R                       │
│       - Stop Recording: Ctrl+Shift+S                        │
│                                                             │
└─────────────────────────────────────────────────────────────┘
```

Greenshot — Capturas de Pantalla

Campo	Valor
Descarga	getgreenshot.org
Costo	Gratis (donación opcional)
Uso principal	Screenshots rápidos y anotados
Formato salida	PNG, JPG

```
┌──────────────────────────────────────────────────────────────────┐
│                   CONFIGURACIÓN GREENSHOT                        │
├──────────────────────────────────────────────────────────────────┤
│                                                                  │
│  1. INSTALACIÓN                                                  │
│     • Descargar desde getgreenshot.org                           │
│     • Instalar y permitir que inicie con Windows                 │
│                                                                  │
│  2. ATAJOS DE TECLADO (por defecto)                              │
│     • Print Screen: Capturar región                              │
│     • Alt + Print Screen: Capturar ventana activa                │
│     • Ctrl + Print Screen: Capturar pantalla completa            │
│                                                                  │
│  3. CONFIGURACIÓN RECOMENDADA                                    │
│     • Click derecho en icono de bandeja → Preferences            │
│     • Output → Storage location: C:\Users\TU_USUARIO\projects\...\media   │
│     • Output → Preferred file format: PNG                        │
│     • Capture → Capture mouse pointer: Sí (para demos)           │
│                                                                  │
│  4. FLUJO DE TRABAJO                                             │
│     • Print Screen → Seleccionar área → Se abre editor           │
│     • En editor: añadir flechas, recuadros, texto si es necesario│
│     • Guardar en carpeta media\screenshots\                      │
│                                                                  │
└──────────────────────────────────────────────────────────────────┘
```

DaVinci Resolve — Edición de Video

Campo	Valor
Descarga	blackmagicdesign.com/products/davinciresolve
Costo	Gratis (versión Studio: $295, NO necesaria)
Uso principal	Edición profesional, exportar a MP4 y GIF
Versión recomendada	DaVinci Resolve 20 (gratis)

```
┌─────────────────────────────────────────────────────────────┐
│                  FLUJO EN DAVINCI RESOLVE                   │
├─────────────────────────────────────────────────────────────┤
│                                                             │
│  1. CREAR PROYECTO                                          │
│     • File → New Project → "Portfolio Demo"                 │
│     • Project Settings: 1920x1080, 30fps                    │
│                                                             │
│  2. IMPORTAR CLIPS                                          │
│     • Media Pool → Import Media → Seleccionar MP4s de OBS   │
│     • Arrastrar clips al Timeline                           │
│                                                             │
│  3. EDICIÓN BÁSICA (pestaña "Edit")                         │
│     • Cortar: Ctrl+B en el punto de corte                   │
│     • Eliminar: Seleccionar clip → Delete                   │
│     • Transiciones: Effects → Video Transitions → Arrastrar │
│     • Texto: Effects → Titles → Text+ → Arrastrar al timeline│
│                                                             │
│  4. AÑADIR AUDIO/NARRACIÓN                                  │
│     • Importar archivo de audio con narración               │
│     • Arrastrar a pista de audio debajo del video           │
│     • Ajustar volumen en el Inspector                       │
│                                                             │
│  5. EXPORTAR VIDEO (pestaña "Deliver")                      │
│     • Preset: YouTube 1080p                                 │
│     • Format: MP4                                           │
│     • Codec: H.264                                          │
│     • Location: C:\Videos\Portfolio\Final                   │
│     • Click "Add to Render Queue" → "Render All"            │
│                                                             │
│  6. EXPORTAR COMO GIF (método indirecto)                    │
│     • Exportar segmento corto como MP4 (10-15 seg)          │
│     • Usar ffmpeg para convertir a GIF (ver Comandos Útiles)│
│                                                             │
└─────────────────────────────────────────────────────────────┘
```

Herramientas Auxiliares (Windows)

Herramienta	Uso	Instalación
ffmpeg	Conversión video→GIF	`winget install ffmpeg`
Docker Desktop	Ejecutar servicios	docker.com
Windows Terminal	Terminal moderna	Microsoft Store
Git for Windows	Control de versiones	`winget install Git.Git`

⚡ SECCIÓN 7: COMANDOS ÚTILES (Windows PowerShell)

Instalar ffmpeg (si no lo tienes)

```
# Opción 1: Con winget (recomendado)
winget install ffmpeg

# Opción 2: Con Chocolatey
choco install ffmpeg

# Verificar instalación
ffmpeg -version
```

Convertir MP4 a GIF (método básico)

```
# Conversión simple
ffmpeg -i "C:\Videos\Portfolio\demo.mp4" `
  -vf "fps=12,scale=800:-1:flags=lanczos" `
  -loop 0 `
  "media\gifs\demo.gif"
```

Convertir MP4 a GIF (alta calidad con paleta)

```
# Paso 1: Crear paleta de colores
ffmpeg -i "C:\Videos\Portfolio\demo.mp4" `
  -vf "fps=12,scale=800:-1:flags=lanczos,palettegen" `
  palette.png

# Paso 2: Generar GIF usando la paleta
ffmpeg -i "C:\Videos\Portfolio\demo.mp4" -i palette.png `
  -filter_complex "fps=12,scale=800:-1:flags=lanczos[x];[x][1:v]paletteuse" `
  "media\gifs\demo.gif"

# Paso 3: Limpiar paleta temporal
Remove-Item palette.png
```

Optimizar tamaño de GIF (requiere gifsicle)

```
# Descargar gifsicle de: https://www.lcdf.org/gifsicle/
# O instalar con Chocolatey: choco install gifsicle

# Reducir colores y optimizar
gifsicle -O3 --colors 128 "media\gifs\demo.gif" -o "media\gifs\demo-optimized.gif"
```

Recortar video (extraer segmento para GIF)

```
# Extraer desde segundo 5, duración 10 segundos
ffmpeg -ss 00:00:05 -i "demo-full.mp4" -t 00:00:10 -c copy "demo-clip.mp4"
```

Comprimir imágenes PNG

```
# Instalar pngquant (alternativa a optipng para Windows)
winget install pngquant

# Comprimir manteniendo calidad
pngquant --quality=65-80 "media\screenshots\*.png" --ext .png --force
```

Script completo: Grabar → Convertir → Optimizar

```powershell
# === SCRIPT COMPLETO PARA CREAR GIF ===
# Ejecutar después de grabar con OBS

$videoPath = "C:\Videos\Portfolio\portfolio-demo.mp4"
$gifOutput = "media\gifs\portfolio-demo.gif"

# Convertir con paleta (mejor calidad)
ffmpeg -i $videoPath -vf "fps=12,scale=800:-1:flags=lanczos,palettegen" palette.png
ffmpeg -i $videoPath -i palette.png -filter_complex "fps=12,scale=800:-1:flags=lanczos[x];[x][1:v]paletteuse" $gifOutput
Remove-Item palette.png

# Verificar tamaño
$size = (Get-Item $gifOutput).Length / 1MB
Write-Host "GIF creado: $gifOutput ($([math]::Round($size, 2)) MB)"

if ($size -gt 5) {
    Write-Host "⚠ GIF muy grande. Considera reducir duración o resolución."
}
```

SECCIÓN 8: MAPEO DE REFERENCIAS

8.1 Actualización de README.md (Raíz)

```
# Línea ~41: Descomentar GIF del portfolio
ANTES:  <!-- ![Portfolio Demo](media/gifs/portfolio-demo.gif) -->
DESPUÉS: ![Portfolio Demo](media/gifs/portfolio-demo.gif)

# Líneas 43-44: Reemplazar link del video
ANTES:  **[ WATCH FULL DEMO VIDEO — CLICK HERE](#)**
DESPUÉS: **[ WATCH FULL DEMO VIDEO — CLICK HERE](https://youtube.com/watch?v=YOUR_VIDEO_ID)**

# Línea ~73: Descomentar GIF de BankChurn
ANTES:  <!-- ![BankChurn Demo](media/gifs/bankchurn-preview.gif) -->
DESPUÉS: ![BankChurn Demo](media/gifs/bankchurn-preview.gif)

# Línea ~97: Descomentar GIF de CarVision
ANTES:  <!-- ![CarVision Demo](media/gifs/carvision-preview.gif) -->
DESPUÉS: ![CarVision Demo](media/gifs/carvision-preview.gif)

# Línea ~121: Descomentar GIF de TelecomAI
ANTES:  <!-- ![TelecomAI Demo](media/gifs/telecom-preview.gif) -->
DESPUÉS: ![TelecomAI Demo](media/gifs/telecom-preview.gif)
```

8.2 Actualización de READMEs de Proyectos

BankChurn-Predictor/README.md:

```
# Línea ~30: Descomentar GIF y eliminar placeholder
# Línea ~33: Actualizar link de video con timestamp
```

CarVision-Market-Intelligence/README.md:

```
# Línea ~30: Descomentar GIF y eliminar placeholder
# Línea ~33: Actualizar link de video con timestamp
```

TelecomAI-Customer-Intelligence/README.md:

```
# Línea ~29: Descomentar GIF y eliminar placeholder
# Línea ~32: Actualizar link de video con timestamp
```

8.3 Actualización de docs/projects/*.md

```
# docs/projects/bankchurn.md línea ~8
# docs/projects/carvision.md línea ~8
# docs/projects/telecom.md línea ~8
# Descomentar las líneas de GIF
```

SECCIÓN 9: CHECKLIST FINAL

9.1 Material Crítico (Prioridad Alta)

- [ ] `media/gifs/portfolio-demo.gif` — Crear y colocar
- [ ] `media/gifs/bankchurn-preview.gif` — Crear y colocar
- [ ] `media/gifs/carvision-preview.gif` — Crear y colocar
- [ ] `media/gifs/telecom-preview.gif` — Crear y colocar
- [ ] Video principal subido a YouTube/Drive
- [ ] Actualizar `README.md` (raíz) con links
- [ ] Actualizar READMEs de proyectos con links

9.2 Material Secundario (Prioridad Media)

- [ ] `media/screenshots/grafana-dashboard.png`
- [ ] `media/screenshots/mlflow-experiments.png`
- [ ] `media/screenshots/streamlit-carvision.png`
- [ ] `media/screenshots/swagger-ui.png`
- [ ] Actualizar `media/README.md` con checklist

9.3 Material Opcional (Prioridad Baja)

- [ ] Videos individuales por proyecto
- [ ] Thumbnails para YouTube
- [ ] `docs/projects/*.md` actualizados con GIFs

SECCIÓN 10: PLAN DE ACCIÓN (Windows)

Día 1: Preparación del Entorno

```
# 1. Instalar herramientas con winget
winget install OBSProject.OBSStudio
winget install Greenshot.Greenshot
winget install ffmpeg
# DaVinci Resolve: descargar manualmente de blackmagicdesign.com

# 2. Verificar Docker Desktop está corriendo
docker --version

# 3. Levantar stack completo
cd "C:\Users\TU_USUARIO\projects\Projects Tripe Ten"
docker-compose -f docker-compose.demo.yml up -d

# 4. Verificar servicios
Start-Sleep -Seconds 30
Invoke-RestMethod -Uri "http://localhost:8001/health"
Invoke-RestMethod -Uri "http://localhost:8002/health"
Invoke-RestMethod -Uri "http://localhost:8003/health"
```

Día 2: Grabar y Crear GIFs

1. **Configurar OBS Studio** (ver Sección 6)
2. **Grabar cada demo** según los guiones:
   - `portfolio-demo.mp4` (15 seg)
   - `bankchurn-demo.mp4` (8 seg)
   - `carvision-demo.mp4` (10 seg)

- `telecom-demo.mp4` (8 seg)

3. **Convertir a GIF** con ffmpeg (ver Comandos Útiles)
4. **Colocar en** `media\gifs\`
5. **Actualizar READMEs** (descomentar líneas de imagen)

Día 3: Screenshots con Greenshot

1. **Abrir cada servicio** en el navegador
2. **Capturar con Print Screen**:
   - Grafana Dashboard
   - MLflow Experiments
   - Streamlit CarVision
   - Swagger UI
3. **Editar en Greenshot** si es necesario (anotaciones, recortes)
4. **Guardar en** `media\screenshots\`

Día 4: Video Principal en DaVinci Resolve

1. **Practicar guion** 2-3 veces
2. **Grabar en OBS** con narración de voz (o grabar audio aparte)
3. **Importar en DaVinci Resolve**
4. **Editar**:
   - Cortar pausas largas
   - Añadir títulos/texto
   - Ajustar audio
5. **Exportar** como MP4 1080p
6. **Subir a YouTube** (unlisted) o Google Drive
7. **Actualizar links** en READMEs

Día 5: Verificación Final

```
# 1. Verificar archivos creados
Get-ChildItem -Path "media\gifs\*.gif" | Select-Object Name, @{N='Size(MB)';E={[math]::Round($_.Length/1MB,2)}}
Get-ChildItem -Path "media\screenshots\*.png" | Select-Object Name, @{N='Size(KB)';E={[math]::Round($_.Length/1KB,2)}}

# 2. Commit y push
git add media/
git commit -m "docs: add all demo media assets (GIFs, screenshots)"
git push

# 3. Verificar en GitHub que las imágenes se muestran correctamente
Start-Process "https://github.com/DuqueOM/ML-MLOps-Portfolio"
```

RESUMEN DE ARCHIVOS A CREAR

```
media/
├── gifs/
│   ├── portfolio-demo.gif      ← PENDIENTE (Alta prioridad)
│   ├── bankchurn-preview.gif   ← PENDIENTE (Alta prioridad)
│   ├── carvision-preview.gif   ← PENDIENTE (Alta prioridad)
│   └── telecom-preview.gif     ← PENDIENTE (Alta prioridad)
├── screenshots/
│   ├── grafana-dashboard.png   ← PENDIENTE (Media prioridad)
│   ├── mlflow-experiments.png  ← PENDIENTE (Media prioridad)
│   ├── streamlit-carvision.png ← PENDIENTE (Media prioridad)
│   └── swagger-ui.png          ← PENDIENTE (Media prioridad)
├── thumbnails/
│   ├── portfolio-thumb.png     ← PENDIENTE (Baja prioridad)
│   ├── bankchurn-thumb.png     ← PENDIENTE (Baja prioridad)
│   └── carvision-thumb.png     ← PENDIENTE (Baja prioridad)
└── videos/
    └── (hosted externally)     ← PENDIENTE en YouTube/Drive
```

Autor: Guía MLOps Portfolio
Última actualización: Noviembre 2025
Estado del portafolio después de completar: 100%