

# GUÍA COMPLETA DE INSTALACIÓN Y EJECUCIÓN

## Analizador de Riesgo Emocional - Windows + Python 3.13

---

### REQUISITOS PREVIOS

#### ✓ Lo que ya tienes:

- ✓ Windows (cualquier versión reciente)
- ✓ Python 3.13 o superior

#### Lo que necesitas instalar:

1. Ollama (para el LLM local)
  2. Librerías Python específicas
- 

### PASO 1: INSTALAR OLLAMA

Ollama es el motor LLM que ejecuta los modelos de lenguaje localmente en tu PC.

#### Instalación:

##### 1. Descargar Ollama para Windows:

- Ve a: <https://ollama.com/download>
- Haz clic en "Download for Windows"
- Se descargará un archivo `.exe`

##### 2. Instalar Ollama:

- Ejecuta el archivo descargado `OllamaSetup.exe`
- Sigue el asistente de instalación (siguiente, siguiente, instalar)
- Espera a que termine la instalación

##### 3. Verificar instalación:

- Abre **PowerShell** o **CMD**
- Escribe:

```
bash
```

```
ollama --version
```

- Deberías ver algo como: `ollama version 0.x.x`

#### 4. Descargar el modelo llama3.2:1b:

```
bash

ollama pull llama3.2:1b
```

- Este comando descarga el modelo (~600MB)
- Espera a que termine (puede tardar 5-10 minutos)

#### 5. Verificar que el modelo funciona:

```
bash

ollama run llama3.2:1b "Hola, ¿cómo estás?"
```

- Deberías ver una respuesta del modelo
- Presiona `Ctrl+D` o escribe `/bye` para salir

---

## PASO 2: INSTALAR LIBRERÍAS PYTHON

### Opción A: Instalación con pip (RECOMENDADO)

Abre **PowerShell** o **CMD** y ejecuta:

```
bash

pip install ollama chromadb sentence-transformers
```

#### Estas librerías se instalan:

- `ollama`: Cliente Python para comunicarse con Ollama
- `chromadb`: Base de datos vectorial (para búsquedas semánticas)
- `sentence-transformers`: Modelos de embeddings (usado por ChromaDB)

### Opción B: Instalación desde archivo requirements.txt

1. Crea un archivo llamado `requirements.txt` con este contenido:

```
ollama>=0.3.0
chromadb>=0.4.0
sentence-transformers>=2.2.0
```

2. Instala todo de una vez:

```
bash

pip install -r requirements.txt
```

### Verificar instalación:

```
bash

python -c "import ollama; import chromadb; print('✅ Librerías instaladas correctamente')"
```

Si ves el mensaje de éxito, ¡todo está listo!

---

## PASO 3: DESCARGAR EL CÓDIGO

### Opción A: Copiar el código manualmente

1. Crea una carpeta para el proyecto:

```
bash

mkdir AnalizadorRiesgoEmocional
cd AnalizadorRiesgoEmocional
```

2. Crea un archivo llamado `emotional_risk_analyzer.py`

3. Copia todo el código del analizador en ese archivo

### Opción B: Desde un repositorio (si lo tienes en GitHub)

```
bash

git clone <URL_DEL_REPOSITORIO>
cd AnalizadorRiesgoEmocional
```

---

## PASO 4: EJECUTAR EL PROGRAMA

### 4.1 Verificar que Ollama está corriendo

Ollama debe estar ejecutándose en segundo plano. Para verificar:

```
bash

ollama list
```

Si ves la lista de modelos (incluido llama3.2:1b), está corriendo correctamente.

Si no está corriendo:

- En Windows, Ollama generalmente se inicia automáticamente
- Si no, busca "Ollama" en el menú inicio y ábrelo

4.2 Ejecutar en modo interactivo (RECOMENDADO)


Abre PowerShell o CMD en la carpeta del proyecto y ejecuta:

```
bash

python emotional_risk_analyzer.py
```

Verás algo como:

=====

 ANALIZADOR DE RIESGO EMOCIONAL

DÍA 2 - Desarrollo del Prototipo (25%)

=====

Comandos disponibles:

/analizar

- Analizar un nuevo mensaje

/caso

- Simular caso de estudio (María)

/status


- Ver estado del sistema

/test


- Ejecutar suite de pruebas


/salir


- Salir del sistema


 Componentes integrados: MCP | A2A | LLM | N8N | Tests


=====

 MCP Server inicializado (puerto 8080)

 A2A Agent inicializado (Ollama: llama3.2:1b)


 Agent ID: a3f5c912

 LLM Integration: Ollama (llama3.2:1b)


 N8N Flow Manager inicializado

• 3 flujos creados

=====

 AGENTE LISTO - Sesión: b7e4d9a1

=====

 Comando:

---

## 4.3 Comandos disponibles

Una vez que el programa esté corriendo, puedes usar estos comandos:

### **/analizar** - Analizar un mensaje personalizado

💙 Comando: /analizar  
👤 Nombre: María  
📄 Mensaje: Me siento muy triste y sola

El sistema analizará el mensaje y mostrará:

- Nivel de riesgo detectado
- Indicadores emocionales
- Respuesta empática generada
- Alertas activadas (si aplica)
- Workflows ejecutados

---

### **/caso** - Simular el caso de María

💙 Comando: /caso

Ejecuta automáticamente el caso de estudio del documento:

"Ya no sé qué sentido tiene seguir intentando. Todo está oscuro y no veo salida. Nadie me entendería de todas formas."

Muestra el procesamiento completo con todos los componentes.

---

### **/status** - Ver estado del sistema

💙 Comando: /status

Muestra:

- Estado de todos los componentes (MCP, A2A, LLM, N8N)
- Estadísticas de uso:
  - Total de evaluaciones realizadas

- Casos de alto riesgo detectados
- Alertas activadas
- Workflows ejecutados

---

### `/test` - Ejecutar pruebas

💙 Comando: `/test`

Ejecuta la suite completa de pruebas:

- ☒ MCP Server
- ☒ A2A Agent
- ☒ LLM Integration
- ☒ N8N Flows
- ☒ Risk Analyzer
- ☒ End-to-End

Muestra resultados y porcentaje de éxito.

---

### `/salir` - Cerrar el programa

💙 Comando: `/salir`

👋 ¡Hasta luego!

---

## 4.4 Ejecutar solo las pruebas (sin interfaz)

```
bash  
  
python emotional_risk_analyzer.py test
```

Ejecuta únicamente la suite de pruebas y muestra resultados.

---

## 4.5 Ejecutar solo el caso de María

```
bash
```

```
python emotional_risk_analyzer.py caso
```

Ejecuta únicamente el caso de estudio sin entrar al modo interactivo.

## PASO 5: VERIFICAR QUE TODO FUNCIONA

### Test completo paso a paso:

#### 1. Inicia el programa:

```
bash

python emotional_risk_analyzer.py
```

#### 2. Ejecuta las pruebas:

 Comando: /test

#### 3. Deberías ver:

```
=====
🧪 EJECUTANDO PRUEBAS
=====
```

- ✅ PASS - MCP Server
- ✅ PASS - A2A Agent
- ✅ PASS - LLM Integration
- ✅ PASS - N8N Flows
- ✅ PASS - Risk Analyzer
- ✅ PASS - End-to-End

```
=====
Total: 6/6 (100%)
=====
```

#### 4. Si ves 6/6 (100%): ¡Todo funciona perfectamente!

## SOLUCIÓN DE PROBLEMAS COMUNES

### Problema 1: "ollama: command not found"

**Causa:** Ollama no está instalado o no está en el PATH

## Solución:

1. Verifica que Ollama esté instalado:
    - Busca "Ollama" en el menú inicio de Windows
    - Si no está, vuelve al PASO 1
  2. Reinicia PowerShell/CMD después de instalar Ollama
  3. Si persiste, cierra sesión y vuelve a iniciar en Windows
- 

## Problema 2: "ModuleNotFoundError: No module named 'ollama'"

**Causa:** La librería ollama no está instalada

### Solución:

```
bash  
pip install ollama
```

Si usas Python 3.13 y da error, prueba:

```
bash  
python -m pip install ollama
```

---

## Problema 3: "Cannot connect to Ollama"

**Causa:** El servicio Ollama no está corriendo

### Solución:

1. Verificar si está corriendo:

```
bash  
ollama list
```

2. Si no responde, iniciar Ollama:
  - Busca "Ollama" en el menú inicio
  - Ábrelo (se ejecutará en segundo plano)
3. Verificar que el servicio está activo:
  - Abre el navegador



- Ve a: <http://localhost:11434>
- Deberías ver: "Ollama is running"

#### 4. Si no inicia, reiniciar Ollama:

```
bash

# En CMD o PowerShell con permisos de administrador
net stop Ollama
net start Ollama
```

---

### Problema 4: "El modelo llama3.2:1b no está disponible"

**Causa:** No descargaste el modelo

**Solución:**

```
bash

ollama pull llama3.2:1b
```

Espera a que termine la descarga (~600MB).

---

### Problema 5: El programa funciona pero las respuestas son genéricas (no usa LLM)

**Síntomas:**

```
⚠ LLM no disponible - respuestas predefinidas
```

**Causa:** Ollama no está conectado correctamente

**Solución:**

#### 1. Verificar conexión:

```
bash

ollama run llama3.2:1b "Hola"
```

#### 2. Si funciona en terminal pero no en el programa:

- Cierra el programa Python
- Reinicia Ollama

- Vuelve a ejecutar el programa

---

## Problema 6: Error con ChromaDB

### Error:

ValueError: Chroma requires sqlite3 >= 3.35.0

### Solución:

**Opción A (RECOMENDADA):** Usar Python 3.11 o 3.12 en lugar de 3.13

Python 3.13 es muy nuevo y puede tener incompatibilidades. Descarga Python 3.12 desde:

<https://www.python.org/downloads/>

**Opción B:** Actualizar SQLite manualmente (avanzado)

1. Descarga SQLite DLL desde: <https://www.sqlite.org/download.html>
2. Reemplaza el archivo en la carpeta de Python

**Opción C:** Desactivar ChromaDB (el sistema seguirá funcionando sin búsqueda vectorial)

---

## Problema 7: "PermissionError: [WinError 5]"

**Causa:** Falta de permisos para crear archivos

### Solución:

1. Ejecuta PowerShell/CMD como **Administrador**:
  - Click derecho en PowerShell
  - "Ejecutar como administrador"
2. O cambia a una carpeta donde tengas permisos:

```
bash
cd C:\Users\TuUsuario\Documents
mkdir AnalizadorRiesgo
cd AnalizadorRiesgo
```

---

## ARCHIVOS GENERADOS AL EJECUTAR

El programa creará automáticamente estos archivos:

AnalizadorRiesgoEmocional/

├── emotional\_risk\_analyzer.py # Tu código principal

├── emotional\_risk.db # Base de datos SQLite (se crea al ejecutar)

└── emotional\_chroma\_db/ # Carpeta ChromaDB (se crea al ejecutar)

└── (archivos internos de ChromaDB)

**Estos archivos NO debes crearlos manualmente**, se generan automáticamente.

---



## EJEMPLO COMPLETO DE SESIÓN

powershell

# 1. Abrir PowerShell en la carpeta del proyecto

PS C:\AnalizadorRiesgo>

# 2. Ejecutar el programa

PS C:\AnalizadorRiesgo> python emotional\_risk\_analyzer.py

---

---

♥ ANALIZADOR DE RIESGO EMOCIONAL

[Sistema se inicializa...]

---

---

# 3. Ejecutar caso de prueba

♥ Comando: /caso

---

---

📄 CASO DE ESTUDIO: MARÍA (23 años)

---

---

💬 Mensaje de María:

"Ya no sé qué sentido tiene seguir intentando.

Todo está oscuro y no veo salida. Nadie me entendería de todas formas."

[Procesamiento completo...]

🚨 URGENTE Nivel: ALTO

📊 Indicadores detectados:

- Ideación suicida
- Desesperanza
- Aislamiento social

♥ RESPUESTA:

---

---

Hola María, entiendo que estás pasando por un momento muy difícil.

Lo que sientes es real e importante. No estás sola en esto. Es crucial que hables con alguien ahora mismo.

---

---

🆘 RECURSOS COMPARTIDOS:

🆘 Línea de Crisis (Colombia): 106 (24/7)

📞 Teléfono de la Esperanza: 57 (1) 323 24 25

...

✅ Caso procesado exitosamente con todos los componentes

---

---

# 4. Ver estadísticas

Comando: /status

ESTADO DEL SISTEMA

Componentes:

MCP Server - Puerto 8080

A2A Agent - ID a3f5c912

LLM - Ollama

N8N Flows - 3 activos

Estadísticas:

Evaluaciones: 1

Alto riesgo: 1

Alertas: 1

Flows ejecutados: 3

# 5. Analizar mensaje propio

Comando: /analizar

Nombre: Juan

Mensaje: Estoy un poco estresado por el trabajo

[Procesamiento...]

MONITOREO Nivel: BAJO

[Respuesta empática generada...]

# 6. Salir

Comando: /salir

¡Hasta luego!

Recuerda: siempre hay ayuda disponible

PS C:\AnalizadorRiesgo>



# CHECKLIST DE INSTALACIÓN

Usa esta lista para verificar que completaste todos los pasos:

- ☐ Python 3.13 instalado y funcionando
- ☐ Ollama descargado e instalado
- ☐ Modelo llama3.2:1b descargado ((ollama pull llama3.2:1b))

- ☐ Ollama corriendo ((ollama list) funciona)
  - ☐ Librerías Python instaladas ((pip install ollama chromadb sentence-transformers))
  - ☐ Código descargado en una carpeta
  - ☐ Programa ejecutado ((python emotional\_risk\_analyzer.py))
  - ☐ Pruebas pasadas ((/test) muestra 6/6)
  - ☐ Caso de María funciona ((/caso))
- 

## COMANDOS ÚTILES DE REFERENCIA

### Comandos Ollama:

```
bash

ollama --version      # Ver versión instalada
ollama list           # Ver modelos descargados
ollama pull llama3.2:1b # Descargar modelo
ollama run llama3.2:1b # Probar modelo interactivo
ollama ps             # Ver modelos en ejecución
```

### Comandos Python:

```
bash

python --version      # Ver versión de Python
pip list              # Ver paquetes instalados
pip install --upgrade <paquete> # Actualizar paquete
python -m pip install <paquete> # Instalar con módulo pip
```

### Comandos del programa:

```
bash

python emotional_risk_analyzer.py      # Modo interactivo
python emotional_risk_analyzer.py test # Solo pruebas
python emotional_risk_analyzer.py caso  # Solo caso María
```

---

## PRÓXIMOS PASOS

Una vez que todo funcione:

#### 1. Experimenta con diferentes mensajes:

- Prueba mensajes de alto, moderado y bajo riesgo

- Observa cómo cambian las respuestas y alertas

## 2. Revisa la base de datos:

- Los archivos se guardan en `emotional_risk.db`
- Puedes abrirlo con DB Browser for SQLite (gratis)

## 3. Analiza las estadísticas:

- Usa `/status` regularmente
- Observa cómo crecen los números

## 4. Modifica el código:

- Añade nuevas palabras clave al analizador
- Personaliza las respuestas empáticas
- Crea nuevos workflows

---

## SOPORTE ADICIONAL

### Si algo no funciona:

#### 1. Revisa la sección de Solución de Problemas (arriba)

#### 2. Verifica versiones:

```
bash

python --version    # Debe ser 3.11+
ollama --version    # Debe ser 0.3+
pip list | grep ollama
```

#### 3. Busca el error específico en la documentación de:

- Ollama: <https://github.com/ollama/ollama/issues>
- ChromaDB: <https://docs.trychroma.com/troubleshooting>

#### 4. Ejecuta el programa con más detalles:

```
bash

python -u emotional_risk_analyzer.py
```

---

## VERIFICACIÓN FINAL

Para confirmar que TODO está funcionando correctamente, ejecuta:

```
bash

python emotional_risk_analyzer.py test
```

Resultado esperado:

RESULTADOS

PASS - MCP Server

PASS - A2A Agent

PASS - LLM Integration

PASS - N8N Flows

PASS - Risk Analyzer

PASS - End-to-End

Total: 6/6 (100%)

Si ves esto, ¡felicidades! Tu sistema está 100% funcional. 🎉

Documento generado para: Windows + Python 3.13  
Última actualización: Octubre 2025  
Sistema: Analizador de Riesgo Emocional - Día 2