Perfecto, Chema. Aquí tienes un **paso a paso profesional y completo** para montar y probar el frontend y backend con **firma digital integrada** desde tu **PC local**, sin necesidad de servidor aún.

**🧰 ENTORNO DE TRABAJO LOCAL – FIRMA DIGITAL + AUTENTICACIÓN JWT**

**✅ REQUISITOS**

Instala en tu PC:

* [Node.js](https://nodejs.org/) (v18+)
* [Python 3.10+](https://www.python.org/)
* [Docker](https://www.docker.com/) (opcional, si prefieres usar contenedores)
* Un editor como VSCode

**🔧 1. CREAR EL BACKEND (FastAPI + Firma)**

**📁 Estructura**

backend\_firma/

├── app/

│ ├── main.py

│ ├── configuracion.py

│ ├── dependencias.py

│ └── rutas/

│ ├── autenticacion.py

│ └── firma.py ✅ nuevo

├── requirements.txt

├── .env

**🔢 Paso a paso**

1. Crea el directorio backend\_firma y su contenido:

**🔹 main.py**

from fastapi import FastAPI

from app.rutas import autenticacion, firma

from fastapi.middleware.cors import CORSMiddleware

app = FastAPI()

app.include\_router(autenticacion.router)

app.include\_router(firma.router)

app.add\_middleware(

CORSMiddleware,

allow\_origins=["\*"],

allow\_credentials=True,

allow\_methods=["\*"],

allow\_headers=["\*"],

)

**🔹 configuracion.py**

from pydantic import BaseSettings

class Settings(BaseSettings):

clave\_jwt: str = "clave\_ultra\_secreta"

algoritmo\_jwt: str = "HS256"

duracion\_token: int = 60

class Config:

env\_file = ".env"

settings = Settings()

**🔹 dependencias.py**

from fastapi import Depends, HTTPException, status

from fastapi.security import OAuth2PasswordBearer

from jose import jwt, JWTError

from app.configuracion import settings

oauth2\_scheme = OAuth2PasswordBearer(tokenUrl="/auth/login")

def obtener\_usuario\_actual(token: str = Depends(oauth2\_scheme)):

try:

payload = jwt.decode(token, settings.clave\_jwt, algorithms=[settings.algoritmo\_jwt])

return payload

except JWTError:

raise HTTPException(

status\_code=status.HTTP\_401\_UNAUTHORIZED,

detail="Token inválido o expirado"

)

**🔹 rutas/autenticacion.py**

from fastapi import APIRouter, Depends, HTTPException

from fastapi.security import OAuth2PasswordRequestForm

from jose import jwt

from datetime import datetime, timedelta

from app.configuracion import settings

router = APIRouter(prefix="/auth", tags=["Autenticación"])

usuarios = {

"admin": {"username": "admin", "password": "segura123"},

}

@router.post("/login")

def login(form\_data: OAuth2PasswordRequestForm = Depends()):

user = usuarios.get(form\_data.username)

if not user or user["password"] != form\_data.password:

raise HTTPException(status\_code=401, detail="Credenciales inválidas")

payload = {

"sub": form\_data.username,

"exp": datetime.utcnow() + timedelta(minutes=settings.duracion\_token)

}

token = jwt.encode(payload, settings.clave\_jwt, algorithm=settings.algoritmo\_jwt)

return {"access\_token": token, "token\_type": "bearer"}

**🔹 rutas/firma.py**

from fastapi import APIRouter, Depends

from app.dependencias import obtener\_usuario\_actual

import requests

router = APIRouter(prefix="/firma", tags=["Firma Digital"])

@router.post("/solicitar")

def firmar\_documento(payload: dict, usuario=Depends(obtener\_usuario\_actual)):

archivo = payload.get("archivo")

print(f"Usuario {usuario['sub']} solicitó firmar: {archivo}")

# Simular llamada a SealSign (puedes sustituir esto por una llamada real):

return {"mensaje": f"Firma simulada de '{archivo}' enviada a SealSign."}

**🔹 .env**

CLAVE\_JWT=clave\_ultra\_secreta

ALGORITMO\_JWT=HS256

DURACION\_TOKEN=60

**🔹 requirements.txt**

fastapi

uvicorn

python-jose[cryptography]

python-multipart

requests

**▶️ Ejecutar backend**

cd backend\_firma

uvicorn app.main:app --reload

Accede a http://localhost:8000/docs para probar login y firma.

**🖥️ 2. CREAR EL FRONTEND (Vue.js)**

**📁 Estructura mínima**

frontend\_firma/

├── src/

│ ├── main.js

│ ├── App.vue

│ ├── router/index.js

│ └── componentes/

│ └── FirmaDigital.vue ✅ nuevo

├── public/

│ └── index.html

├── package.json

**🔹 Componente FirmaDigital.vue**

<template>

<div>

<h2>🖊️ Firmar Documento</h2>

<select v-model="documento">

<option v-for="doc in documentos" :key="doc" :value="doc">{{ doc }}</option>

</select>

<button @click="firmar">Firmar con SealSign</button>

<p v-if="mensaje">{{ mensaje }}</p>

</div>

</template>

<script>

export default {

data() {

return {

documentos: ["informe\_cliente\_001.pdf", "evaluacion\_riesgos\_2025.pdf"],

documento: "",

mensaje: ""

};

},

methods: {

async firmar() {

const res = await fetch("http://localhost:8000/firma/solicitar", {

method: "POST",

headers: {

"Authorization": `Bearer ${localStorage.getItem("token")}`,

"Content-Type": "application/json"

},

body: JSON.stringify({ archivo: this.documento })

});

const data = await res.json();

this.mensaje = data.mensaje;

}

}

};

</script>

**🛠️ Iniciar frontend**

npm install

npm run dev

Accede a: http://localhost:3000

**🧪 PRUEBA COMPLETA**

1. Ve a http://localhost:8000/docs
2. Haz login con admin / segura123 y copia el token.
3. Pega el token en el frontend en localStorage o modifica el login Vue.
4. Elige un documento y pulsa “Firmar”.
5. Verifica la consola del backend que muestra "Firma simulada de archivo XYZ"

¿Quieres que te prepare el archivo .docx con todo esto para entrega, presentación o documentación del proyecto?