

Print Shop POS — Scaffold (Next.js + NestJS + MongoDB)

จุดเริ่มง่ายๆ ตั้งแต่ `create-next-app` + `nest new` พร้อม endpoint อัปโหลดไฟล์, คำนวณเบื้องต้น, และสร้าง QR พร้อมเพย์

Prerequisites

- Node.js ≥ 18
- MongoDB (local: `mongodb://localhost:27017`)
- Windows/macOS/Linux ได้หมด (ตัวอย่างใช้ `npm`)

0) สร้างโฟลเดอร์โปรเจกต์

```
mkdir print-pos && cd print-pos
```

1) สร้าง Next.js app (frontend)

```
npx create-next-app@latest web \
  --ts --eslint --app --src-dir --tailwind --use-npm --no-import-alias
```

ตัวเลือก: App Router + TypeScript + Tailwind

2) สร้าง NestJS app (backend)

```
npx @nestjs/cli new api --package-manager npm
```

หลังจากสร้างเสร็จ โครงจะเป็น:

```
print-pos/
  web/
  api/
```

3) ตั้งค่า ENV

ไฟล์: `api/.env`

```
PORT=3001
MONGODB_URI=mongodb://localhost:27017/printpos
PROMPTPAY_ID=0812345678 # หรือเลขนิติ 13 หลัก
```

ไฟล์: `web/.env.local`

```
NEXT_PUBLIC_API_BASE=http://localhost:3001
```

หมายเหตุ: `PROMPTPAY_ID` คือเบอร์/TaxID ที่จะใช้ generate QR แบบ Merchant-Presented

4) ติดตั้ง deps ฝั่ง backend (Nest)

```
cd api
npm i @nestjs/config @nestjs/mongoose mongoose @nestjs/serve-static
npm i promptpay-qr qrcode multer @nestjs/platform-express
npm i class-validator class-transformer sharp
npm i -D @types/multer
```

5) ติดตั้ง deps ฝั่ง frontend (Next)

```
cd ../web
npm i
```

(ตอนนี้ Next ถูกติดตั้งมาแล้วจาก create-next-app)

6) ปรับ Nest: main.ts, AppModule และโครงสร้าง

ไฟล์: `api/src/main.ts`

```
import { NestFactory } from '@nestjs/core';
import { AppModule } from './app.module';
import { ValidationPipe } from '@nestjs/common';
```

```

async function bootstrap() {
  const app = await NestFactory.create(AppModule);
  app.enableCors({ origin: ['http://localhost:3000'], credentials: false });
  app.useGlobalPipes(new ValidationPipe({ whitelist: true, transform: true }));
  const port = process.env.PORT || 3001;
  await app.listen(port);
  console.log(`API running on http://localhost:${port}`);
}
bootstrap();

```

📄: api/src/app.module.ts

```

import { Module } from '@nestjs/common';
import { ConfigModule } from '@nestjs/config';
import { MongooseModule } from '@nestjs/mongoose';
import { ServeStaticModule } from '@nestjs/serve-static';
import { join } from 'path';
import { PaymentsModule } from '../modules/payments/payments.module';
import { UploadsModule } from '../modules/uploads/uploads.module';
import { PricingModule } from '../modules/pricing/pricing.module';

@Module({
  imports: [
    ConfigModule.forRoot({ isGlobal: true }),
    MongooseModule.forRoot(process.env.MONGODB_URI),
    // เสร็จไฟล์ที่อัปโหลดเป็น static
    ServeStaticModule.forRoot({
      rootPath: join(__dirname, '..', 'uploads'),
      serveRoot: '/files',
    }),

    PaymentsModule,
    UploadsModule,
    PricingModule,
  ],
})
export class AppModule {}

```

โครงสร้างไดเรกทอรี:

```

api/src/modules/
  payments/

```

```
uploads/  
pricing/
```

7) Payments (PromptPay QR)

Ŧwá: `api/src/modules/payments/payments.module.ts`

```
import { Module } from '@nestjs/common';  
import { PaymentsController } from './payments.controller';  
import { PaymentsService } from './payments.service';  
  
@Module({  
  controllers: [PaymentsController],  
  providers: [PaymentsService],  
})  
export class PaymentsModule {}
```

Ŧwá: `api/src/modules/payments/payments.service.ts`

```
import { Injectable } from '@nestjs/common';  
import generatePayload from 'promptpay-qr';  
import * as QR from 'qrcode';  
  
@Injectable()  
export class PaymentsService {  
  async createPromptPayQR(ref: string, amount: number) {  
    const id = process.env.PROMPTPAY_ID;  
    if (!id) throw new Error('PROMPTPAY_ID is not set');  
    const payload = generatePayload(id, { amount });  
    const dataURL = await QR.toDataURL(payload, { margin: 1, scale: 6 });  
    return { ref, amount, payload, dataURL };  
  }  
}
```

Ŧwá: `api/src/modules/payments/payments.controller.ts`

```
import { Body, Controller, Post } from '@nestjs/common';  
import { IsNumber, IsString, Min } from 'class-validator';  
import { PaymentsService } from './payments.service';  
  
class PromptPayDto {  
  @IsString()
```

```

    ref!: string; // orderId หรือรหัสบิล

    @IsNumber()
    @Min(1)
    amount!: number; // จำนวนเงิน (THB)
}

@Controller('payments')
export class PaymentsController {
    constructor(private readonly svc: PaymentsService) {}

    @Post('promptpay')
    createQR(@Body() body: PromptPayDto) {
        return this.svc.createPromptPayQR(body.ref, body.amount);
    }
}

```

8) Uploads (อัปโหลดไฟล์ + คัด URL)

📄: api/src/modules/uploads/uploads.module.ts

```

import { Module } from '@nestjs/common';
import { UploadsController } from './uploads.controller';

@Module({ controllers: [UploadsController] })
export class UploadsModule {}

```

📄: api/src/modules/uploads/uploads.controller.ts

```

import { Controller, Post, UploadedFile, UseInterceptors } from '@nestjs/common';
import { FileInterceptor } from '@nestjs/platform-express';
import { diskStorage } from 'multer';
import { extname } from 'path';

@Controller('uploads')
export class UploadsController {
    @Post()
    @UseInterceptors(
        FileInterceptor('file', {
            storage: diskStorage({
                destination: './uploads',
                filename: (req, file, cb) => {

```

```

        const unique = Date.now() + '-' + Math.round(Math.random() * 1e9);
        cb(null, unique + extname(file.originalname));
    },
    }),
    limits: { fileSize: 50 * 1024 * 1024 }, // 50MB
  )),
)
upload(@UploadedFile() file: Express.Multer.File) {
  return {
    filename: file.filename,
    originalname: file.originalname,
    mimetype: file.mimetype,
    size: file.size,
    url: `/files/${file.filename}`, // ใช้ร่วมกับ ServeStaticModule
  };
}
}

```

ผลลัพธ์ url จะเข้าถึงได้ที่ `http://localhost:3001/files/<filename>`

9) Pricing (กติกาคิดราคาอย่างง่าย)

ไฟล์: `api/src/modules/pricing/pricing.module.ts`

```

import { Module } from '@nestjs/common';
import { PricingController } from './pricing.controller';

@Module({ controllers: [PricingController] })
export class PricingModule {}

```

ไฟล์: `api/src/modules/pricing/pricing.controller.ts`

```

import { Body, Controller, Post } from '@nestjs/common';
import { IsBoolean, IsInt, Min } from 'class-validator';

class EvaluatedDto {
  @IsInt() @Min(1)
  pages!: number;

  @IsBoolean()
  duplex!: boolean; // ตัวอย่าง: ยังไม่คิดผลต่างจริงในสูตรง่าย

  @IsBoolean()

```

```

    grayscale!: boolean; // ตรวจสอบราคา

    @IsInt() @Min(1)
    qty!: number;
  }

  @Controller('pricing')
  export class PricingController {
    @Post('evaluate')
    evaluate(@Body() b: EvaluateDto) {
      const rateColor = 2.5; // THB/หน้า (ตัวอย่าง)
      const rateBW = 1.0;    // THB/หน้า (ตัวอย่าง)
      const rate = b.grayscale ? rateBW : rateColor;
      const unit = b.pages * rate; // สูตรง่าย ๆ: หน้า x rate
      const subtotal = unit * b.qty;
      return { unit, subtotal, currency: 'THB' };
    }
  }
}

```

10) Frontend helper: fetcher

ไฟล์: web/src/lib/api.ts

```

export const API_BASE = process.env.NEXT_PUBLIC_API_BASE!;

export async function api<T>(path: string, init?: RequestInit): Promise<T> {
  const res = await fetch(`${API_BASE}${path}`, init);
  if (!res.ok) throw new Error(await res.text());
  return res.json();
}

```

11) หน้าอัปโหลด + รีวิว (ใส่กรองขาวดำแบบฟรีวีว)

ไฟล์: web/src/app/upload/page.tsx

```

'use client';
import { useState, useMemo } from 'react';
import { API_BASE } from '@lib/api';

export default function UploadPage() {
  const [file, setFile] = useState<File | null>(null);

```

```

const [uploaded, setUploaded] = useState<any>(null);
const [grayscale, setGrayscale] = useState(false);

const previewURL = useMemo(() => (file ? URL.createObjectURL(file) : ''),
[file]);
const uploadedURL = uploaded ? `${API_BASE}${uploaded.url}` : '';

const onUpload = async () => {
  if (!file) return;
  const fd = new FormData();
  fd.append('file', file);
  const res = await fetch(`${API_BASE}/uploads`, { method: 'POST', body:
fd });
  const data = await res.json();
  setUploaded(data);
};

return (
  <div className="max-w-3xl mx-auto p-6 space-y-4">
    <h1 className="text-2xl font-semibold">อัปโหลดไฟล์เพื่อพิมพ์</h1>

    <input type="file" accept="image/*,.pdf" onChange={e =>
setFile(e.target.files?.[0] || null)} />

    {file && (
      <div className="border rounded p-4">
        <div className="flex items-center gap-2 mb-3">
          <label className="flex items-center gap-2">
            <input type="checkbox" checked={grayscale} onChange={e =>
setGrayscale(e.target.checked)} />
            ปร้วิวแบบขาวดำ (เฉพาะปร้วิว)
          </label>
        </div>

        {file.type === 'application/pdf' ? (
          <p className="text-sm text-gray-500">* ปร้วิว PDF แบบย่อ (กรุณาดาวนัโหลดหรือ
อัปโหลดเพื่อใช้งานจริง)</p>
        ) : (
          <img
            src={previewURL}
            alt="preview"
            className="max-h-96"
            style={{ filter: grayscale ? 'grayscale(100%)' : 'none' }}
          />
        )}
      </div>
    )}
  </div>
)

```



```

    <div className="flex gap-3">
      <button onClick={onUpload} className="px-4 py-2 rounded bg-black text-
white disabled:opacity-50" disabled={!file}>
        อัปโหลด
      </button>
      {uploaded && (
        <a href={uploadedURL} target="_blank" className="px-4 py-2 rounded
border">ดูไฟล์ที่อัปโหลด</a>
      )}
    </div>
  </div>
);
}

```

12) หน้า POS (คำนวณราคาย่างๆ แล้วไปหน้า Pay)

ไฟล์: `web/src/app/pos/page.tsx`

```

'use client';
import { useState } from 'react';
import { api } from '@lib/api';
import { useRouter } from 'next/navigation';

export default function PosPage() {
  const [pages, setPages] = useState(1);
  const [qty, setQty] = useState(1);
  const [grayscale, setGrayscale] = useState(false);
  const router = useRouter();


  const [subtotal, setSubtotal] = useState<number | null>(null);

  const calc = async () => {
    const data = await api<{ unit: number; subtotal: number; currency: string }>(
      '/pricing/evaluate',
      {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify({ pages, duplex: false, grayscale, qty }),
      },
    );
    setSubtotal(data.subtotal);
  };
}

```

```

const goPay = async () => {
  const amount = subtotal || 0;
  const orderId = 'ORD-' + Date.now();
  router.push(`/pay?ref=${orderId}&amount=${amount}`);
};

return (
  <div className="max-w-xl mx-auto p-6 space-y-4">
    <h1 className="text-2xl font-semibold">POS  ตัวอย่างคำนวณราคา</h1>
    <label className="block">จำนวนหน้า: <input type="number" min={1}
value={pages} onChange={e=>setPages(+e.target.value)} className="border px-2
py-1 ml-2 w-24"/></label>
    <label className="block">จำนวนชุด: <input type="number" min={1} value={qty}
onChange={e=>setQty(+e.target.value)} className="border px-2 py-1 ml-2 w-24"/></
label>
    <label className="flex items-center gap-2"><input type="checkbox"
checked={grayscale} onChange={e=>setGrayscale(e.target.checked)}> ขาดค่า</label>

    <div className="flex gap-3">
      <button onClick={calc} className="px-4 py-2 rounded bg-black text-
white">คำนวณ</button>
      {subtotal !== null && <div className="px-3 py-2 border rounded">ยอดชำระ:
{subtotal.toFixed(2)} THB</div>}
    </div>

    <button disabled={!subtotal} onClick={goPay} className="px-4 py-2 rounded
bg-emerald-600 text-white disabled:opacity-50">ไปหน้าชำระเงิน</button>
  </div>
);
}

```

13) หน้า Pay (สร้าง QR พร้อมเพย์จาก API)

ไฟล์: `web/src/app/pay/page.tsx`

```

'use client';
import { useEffect, useState } from 'react';
import { API_BASE } from '@lib/api';

export default function PayPage({ searchParams }: any) {
  const ref = searchParams.ref as string;
  const amount = Number(searchParams.amount || 0);
  const [qr, setQr] = useState<string | null>(null);

```

```

useEffect(() => {
  (async () => {
    const res = await fetch(`${API_BASE}/payments/promptpay`, {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify({ ref, amount }),
    });
    const data = await res.json();
    setQr(data.dataURL);
  })();
}, [ref, amount]);

return (
  <div className="min-h-[60vh] grid place-items-center p-6">
    <div className="text-center space-y-3">
      <h1 className="text-2xl font-semibold">สแกนจ่ายด้วยพร้อมเพย์</h1>
      <p>จำนวนเงิน: {amount.toFixed(2)} THB</p>
      {qr ? <img src={qr} alt="PromptPay QR" className="w-64 h-64 mx-auto"/>
      : <div>กำลังสร้าง QR...</div>}
      <p className="text-sm text-gray-500">* ตัวอย่างง่ายๆ ยังไม่ได้ผูก webhook/ยืนยัน
      อัตโนมัติ</p>
    </div>
  </div>
);
}

```

14) รับโปรเจกต์

เปิด 2 Terminal:

Terminal A (API):

```

cd api
npm run start:dev

```

จะได้ `http://localhost:3001`

Terminal B (Web):

```

cd web
npm run dev

```

จะได้ `http://localhost:3000`

ทดลองโฟลว์:

1. ไปที่ `http://localhost:3000/upload` เพื่ออัปโหลดและพรีวิว
2. ไปที่ `http://localhost:3000/pos` คำนวณราคา เลือกขนาด/จำนวนหน้า/จำนวนชุด
3. กด "ไปหน้าชำระเงิน" → `/pay` สร้าง QR พร้อมเพย์

15) ต่อไปนี้แนะนำทำเพิ่ม

- เก็บ `Order` / `Job` ลง MongoDB จริง (ตอนนี้เป็นเพียงตัวอย่างหน้า)
- เพิ่ม webhook หรือ polling เพื่อยืนยันการชำระเงินอัตโนมัติจาก PSP/ธนาคาร
- แปลงไฟล์เป็นขาวดำจริงฝั่ง API (ภาพ: `sharp` / PDF: Ghostscript) ก่อนส่งเข้าคิวพิมพ์
- แยก RBAC, รายงานรายวัน/กะ, การพิมพ์ใบเสร็จ ESC/POS

16) ทดสอบเร็วด้วย curl (ออปชัน)

```
# อัปโหลดไฟล์ตัวอย่าง
curl -F "file=@/path/to/sample.jpg" http://localhost:3001/uploads

# สร้าง QR พร้อมเพย์ 99 บาท
curl -H "Content-Type: application/json" -d '{"ref":"TEST-1","amount":99}'
http://localhost:3001/payments/promptpay

# คำนวณราคา
curl -H "Content-Type: application/json" -d '{"pages":
10,"duplex":false,"grayscale":true,"qty":2}' http://localhost:3001/pricing/
evaluate
```

ถ้าติดอะไร แจ้ง error ขึ้นมาได้เลย เดี๋ยวช่วยไล่ให้ครับ 🎯