

PRD: Minimal Accounting & Document System (VAT 7% + WHT) สำหรับหจก. ใช้เอง

0) สรุปเป้าหมาย

สร้างเว็บแอป “บัญชีมินิมัล” สำหรับหจก. ที่

- ขายผ่าน เว็บไซต์ของตัวเอง
- มีงานบริการ ออกระบบ/เขียนโปรแกรม
- จด VAT 7%
- ลูกค้าส่วนใหญ่เป็น นิติบุคคล/สถานศึกษา → เจอ WHT ปอย
- ต้อง ออกรหัส PDF: ใบเสนอราคา, ใบแจ้งหนี้, ใบกำับภาษี+ใบเสร็จ
- ต้องสร้าง Project ได้ และมี หน่วยงานย่อยให้โรงเรียน/หน่วยงาน (CustomerUnit)

1) ขอบเขต v1 (Must Have)

ฟีเจอร์หลัก

1. Customer / CustomerUnit

- เก็บข้อมูลลูกค้า (บริษัท/สถานศึกษา)
- สร้างหน่วยงานย่อยให้ลูกค้า (เช่น งานพัสดุ/ฝ่าย IT/โครงการ...)

2. Orders (ขายผ่านเว็บ)

- Order + Items
- Shipment (Flash Express ฯลฯ)
- สถานะ: Draft → Invoiced → Paid → Closed / Void

3. Projects (งานบริการ)

- Project + Milestones (งวดงาน)
- ออกเอกสารรายงวดได้

4. Documents

- DocType: QUOTATION / INVOICE / TAX_INVOICE_RECEIPT
- เก็บ snapshot: subtotal / vat / total ให้ “นิ่ง” และถูกต้อง

5. Payments

- รองรับกรณีลูกค้าหัก WHT: รับเงินจริง = grandTotal - whtAmount

6. Expenses

- แบบหลักฐานหลายไฟล์
- แยก VAT Input ได้

7. VAT & WHT Reports

- รายงานภาษีขายรายเดือน (CSV)
- รายงานภาษีซื้อรายเดือน (CSV)
- รายงาน WHT (CSV)

8. Monthly Close (ล็อกเดือน)

- ปิดเดือนแล้วห้ามแก้รายการในเดือนนั้น (กัน VAT เพียง)

9. PDF Export

- สร้าง PDF A4 จาก HTML template ด้วย Puppeteer
- บุ่ม Print/Download PDF ในหน้า Document

2) ขอบเขต v1 (Not in v1)

- ระบบสต็อกคงเหลือ (ทำแค่ต้นทุนต่อออเดอร์/ค่าใช้จ่ายได้)
- e-Tax Invoice/Receipt ตามมาตรฐานเต็ม (ค่อยทำ v2)
- Multi-tenant / SaaS (เพราะ “ใช้อ่อง”)

3) User Roles (มินิมัล)

- Admin/Owner: ทำทุกอย่าง
- Staff: บันทึกรายจ่าย/แบบหลักฐาน/ออกเอกสาร (ถ้าจะทำ v1 ก็ role เดียวได้ก่อน)

4) UX Pages (6 หน้าหลัก)

- Dashboard
- Customers (รวม Units)

3. Orders
 4. Projects
 5. Documents
 6. Expenses + Reports + Close Month (รวมเป็นหน้าเดียวแบบ Tab)
-

5) Business Rules (คำนวณ)

- VAT rate ค่าเริ่มต้น = 7.00
- vatAmount = round(subTotal * vatRate/100, 2)
- grandTotal = subTotal + vatAmount
- WHT (บริการเจอบอย 3%):
$$\text{whtAmount} = \text{round}(\text{whtBase} * \text{whtRate}/100, 2)$$
$$\text{netReceived} = \text{grandTotal} - \text{whtAmount}$$
- เอกสาร **TAX_INVOICE_RECEIPT** ออกเมื่อ “รับเงินแล้ว”
- **MonthlyClose:** ถ้าเดือนถูกปิดแล้ว ห้ามแก้:
 - Document.issueDate ในเดือนนั้น
 - Expense.expenseDate ในเดือนนั้น
 - Payment.receivedAt ในเดือนนั้น
 - Order.orderDate/หรือ updatedAt ก็ได้ (แนะนำตรวจสอบเอกสารและวันที่รายการเงินจริง)

โครงไฟล์ที่ให้ Cursor สร้าง

/app
(dashboard)/page.tsx
/customers/page.tsx
/orders/page.tsx
/projects/page.tsx
/documents/page.tsx

/finance/page.tsx (tabs: Expenses, Reports, Close Month)

/api

/customers/route.ts

/customers/[id]/route.ts

/customer-units/route.ts

/orders/route.ts

/orders/[id]/route.ts

/projects/route.ts

/documents/route.ts

/documents/[id]/route.ts

/documents/[id]/pdf/route.ts

/expenses/route.ts

/reports/vat-sales/route.ts

/reports/vat-purchase/route.ts

/reports/wht/route.ts

/monthly-close/route.ts

/src

/lib

prisma.ts

tax.ts

docNumber.ts

monthLock.ts

csv.ts

/docTemplates

base.ts

quotation.ts

invoice.ts

taxInvoiceReceipt.ts

render.ts

/prisma

schema.prisma

seed.ts

Prisma schema.prisma (MariaDB) — v1 พร้อม Unit + MonthlyClose + PDF support

(คัดลอกไปใช้ได้เลย)

```
generator client {
```

```
    provider = "prisma-client-js"
```

```
}
```

```
datasource db {
```

```
    provider = "mysql"
```

```
    url      = env("DATABASE_URL") // MariaDB ใช้ provider=mysql ได้
```

```
}
```

```
enum CustomerType { COMPANY SCHOOL INDIVIDUAL }
```

```
enum DocType { QUOTATION INVOICE TAX_INVOICE_RECEIPT }
```

```
enum OrderStatus { DRAFT INVOICED PAID CLOSED VOID }
```

```
enum PaymentMethod { CASH TRANSFER CARD OTHER }
```

```
enum ExpenseCategory {
```

```
    INVENTORY_PURCHASE SHIPPING_OUT PLATFORM_FEE MARKETING SOFTWARE_CLOUD
```

```
SALARY_FREELANCE TRAVEL_COMM OFFICE_SUPPLIES OTHER  
}
```

```
model Setting {  
    key      String @id  
    value    String  
    updatedAt DateTime @updatedAt  
}
```

```
model Customer {  
    id          String      @id @default(cuid())  
    name        String  
    type        CustomerType @default(COMPANY)  
    taxId       String?  
    address     String?  
    email       String?  
    phone       String?  
    createdAt   DateTime    @default(now())  
    updatedAt   DateTime    @updatedAt
```

```
units      CustomerUnit[]  
orders     Order[]  
projects   Project[]  
documents  Document[]
```

```
@@index([name])
```

```
@@index([type])  
}  
  
model CustomerUnit {  
    id      String  @id @default(cuid())  
    customerId  String  
    customer  Customer @relation(fields: [customerId], references: [id])  
  
    name      String  
    contactName  String?  
    contactPhone String?  
    contactEmail String?  
  
    projects  Project[]  
    createdAt  DateTime @default(now())  
    updatedAt  DateTime @updatedAt  
  
    @@index([customerId, name])  
}  
  
model Order {  
    id      String  @id @default(cuid())  
    code    String   @unique  
    customerId  String  
    customer  Customer @relation(fields: [customerId], references: [id])
```

```
status      OrderStatus @default(DRAFT)
orderDate   DateTime   @default(now())

subTotal    Decimal    @db.Decimal(12,2)
vatRate     Decimal    @db.Decimal(5,2)
vatAmount   Decimal    @db.Decimal(12,2)
grandTotal  Decimal    @db.Decimal(12,2)

items       OrderItem[]
shipment    Shipment?
payments   Payment[]
documents  Document[]
attachments Attachment[]

createdAt  DateTime   @default(now())
updatedAt  DateTime   @updatedAt

@@index([status, orderDate])
@@index([customerId, orderDate])
}

model OrderItem {
  id        String  @id @default(cuid())
  orderId   String
  order     Order   @relation(fields: [orderId], references: [id])
}
```

```
name      String  
qty       Int      @default(1)  
unitPrice Decimal  @db.Decimal(12,2)  
lineTotal Decimal  @db.Decimal(12,2)  
}
```

```
model Shipment {  
    id      String  @id @default(cuid())  
    orderId String  @unique  
    order   Order   @relation(fields: [orderId], references: [id])  
  
    carrier String  
    trackingNo String?  
    shippedAt DateTime?  
    shippingFee Decimal? @db.Decimal(12,2)  
}
```

```
model Project {  
    id      String  @id @default(cuid())  
    code   String  @unique  
    customerId String  
    customer Customer @relation(fields: [customerId], references: [id])  
  
    unitId String?  
    unit    CustomerUnit? @relation(fields: [unitId], references: [id])
```

```
title      String
startDate  DateTime?
endDate    DateTime?

milestones ProjectMilestone[]
documents   Document[]
attachments Attachment[]

createdAt  DateTime @default(now())
updatedAt   DateTime @updatedAt

@@index([customerId, unitId])

}

model ProjectMilestone {
    id      String  @id @default(cuid())
    projectId  String
    project   Project @relation(fields: [projectId], references: [id])
    title      String
    amount     Decimal @db.Decimal(12,2)
    dueDate    DateTime?
    isBilled   Boolean @default(false)
}

model Document {
```

```
id      String  @id @default(cuid())
type    DocType
number  String  @unique
issueDate DateTime @default(now())

customerId String
customer   Customer @relation(fields: [customerId], references: [id])

orderId   String?
order     Order?  @relation(fields: [orderId], references: [id])

projectId String?
project   Project? @relation(fields: [projectId], references: [id])

subTotal  Decimal @db.Decimal(12,2)
vatRate   Decimal @db.Decimal(5,2)
vatAmount Decimal @db.Decimal(12,2)
grandTotal Decimal @db.Decimal(12,2)

withholdingTax WithholdingTax?
attachments Attachment[]
payments   Payment[]

createdAt DateTime @default(now())
updatedAt  DateTime @updatedAt
```

```

@@index([type, issueDate])
@@index([customerId, issueDate])
}

model Payment {
    id      String      @id @default(cuid())
    receivedAt  DateTime   @default(now())
    method     PaymentMethod @default(TRANSFER)
    amount     Decimal     @db.Decimal(12,2)
    note      String?

    orderId   String?
    order     Order?     @relation(fields: [orderId], references: [id])

    documentId String?
    document   Document?  @relation(fields: [documentId], references: [id])

    attachments Attachment[]
}

@@index([receivedAt])
}

model WithholdingTax {
    id      String      @id @default(cuid())
    documentId String  @unique
    document   Document @relation(fields: [documentId], references: [id])
}

```

```
rate      Decimal @db.Decimal(5,2)

baseAmount  Decimal @db.Decimal(12,2)

taxAmount   Decimal @db.Decimal(12,2)

certificateNo String?

certificateDate DateTime?

}
```

```
model Expense {

    id      String      @id @default(cuid())

    expenseDate  DateTime     @default(now())

    vendorName   String

    category     ExpenseCategory

    description  String?

    hasVat      Boolean     @default(false)

    subTotal    Decimal     @db.Decimal(12,2)

    vatRate     Decimal     @db.Decimal(5,2)

    vatAmount   Decimal     @db.Decimal(12,2)

    grandTotal  Decimal     @db.Decimal(12,2)

    paymentMethod PaymentMethod @default(TRANSFER)

    paidAmount   Decimal?    @db.Decimal(12,2)

    relatedOrderId String?

    relatedProjectId String?


}
```

```
attachments Attachment[]
createdAt DateTime @default(now())
updatedAt DateTime @updatedAt

@@index([expenseDate])
@@index([category, expenseDate])
}
```

```
model Attachment {
    id String @id @default(cuid())
    fileName String
    mimeType String
    url String
    uploadedAt DateTime @default(now())
```

```
orderId String?
documentId String?
expenseld String?
paymentId String?
projectId String?
}
```

```
model MonthlyClose {
    id String @id @default(cuid())
    year Int
```

```
month    Int  
closedAt DateTime @default(now())  
note     String?
```

```
@@unique([year, month])
```

```
@@index([year, month])
```

```
}
```

Seed ข้อมูลพื้นฐาน (prisma/seed.ts) — “ข้อมูลจริงที่ใช้คำนวณล่วงหน้า”

จุดสำคัญ: seed ต้องสร้าง “Setting/Tax/Prefix” + หมวดพื้นฐาน + ลูกค้าตัวอย่าง 1 โรงเรียน + units 3 หน่วย + project 1 งาน

สิ่งที่ต้อง seed อย่างน้อย

- Setting:
 - VAT_RATE = 7.00
 - DEFAULT_WHT_SERVICE_RATE = 3.00
 - DOC_PREFIX_QUOT = QT
 - DOC_PREFIX_INV = INV
 - DOC_PREFIX_TAX = TX
 - COMPANY_NAME / COMPANY_TAXID / COMPANY_ADDRESS / COMPANY_BRANCH
 - CARRIER_DEFAULT = Flash Express

(ให้ Cursor ทำ seed.ts ตามนี้)

PDF Export Spec (Puppeteer)

วิธีทำ

- สร้าง HTML template ต่อ DocType

- API: GET /api/documents/:id/pdf
→ ดึง Document + relations → render HTML → puppeteer page.pdf(A4) → ส่งเป็น application/pdf

Templates ที่ต้องมี

- src/docTemplates/quotation.ts
 - src/docTemplates/invoice.ts
 - src/docTemplates/taxInvoiceReceipt.ts
 - src/docTemplates/render.ts (เลือก template ตาม type)
-

Guard: Month Lock (สำคัญมาก)

- src/lib/monthLock.ts
 - ฟังก์ชัน assertNotClosed(date: Date)
 - เช็คใน MonthlyClose ว่า year/month ถูกปิดหรือยัง
 - ทุก API ที่ “create/update/delete” ของ Document/Expense/Payment ให้เรียก guard ก่อนเสมอ
-

รายงาน CSV (Reports)

API:

- /api/reports/vat-sales?year=2026&month=2
- /api/reports/vat-purchase?year=2026&month=2
- /api/reports/wht?year=2026&month=2

ผลลัพธ์:

- Content-Type: text/csv
- ใช้ src/lib/csv.ts สร้าง CSV
- VAT Sales = รวม Document ที่ type = TAX_INVOICE_RECEIPT ในเดือนนั้น
- VAT Purchase = รวม Expense ที่ hasVat = true ในเดือนนั้น

- WHT = รวม WithholdingTax ในเดือนนั้น
-

ชุด Prompt ส่งให้ Cursor AI (คัดลอกใช้ได้ทันที)

You are a senior full-stack engineer + accounting-aware system designer.

Build a minimal accounting web app for a Thai registered partnership (หจก.) used internally (single company).

Business: sales via own website + software/system development services.

Tax: VAT registered 7%. Customers are mostly legal entities/schools, so withholding tax (WHT) is common.

DB: MariaDB via Prisma (provider=mysql).

Tech:

- Next.js App Router + TypeScript
- Prisma + MariaDB
- Tailwind + shadcn/ui
- Zod validation
- File attachments stored on disk (or local volume), DB stores only URL.

Must-haves:

- 1) Customer + CustomerUnit (units under a school/organization).
- 2) Orders (items, shipment, statuses).
- 3) Projects with milestones (service work by phases).
- 4) Documents: QUOTATION, INVOICE, TAX_INVOICE_RECEIPT.
 - Save snapshot totals: subTotal, vatRate, vatAmount, grandTotal.
- 5) Payments: support WHT net received = grandTotal - whtAmount.

- 6) Expenses: VAT input support, attachments.
- 7) Reports (CSV): VAT sales, VAT purchase, WHT monthly.
- 8) MonthlyClose: lock month to prevent edits in closed months.
- 9) PDF export A4 using Puppeteer:
 - GET /api/documents/:id/pdf returns PDF
 - HTML templates per DocType.

Deliverables:

- File tree exactly as specified
- prisma/schema.prisma updated (use the provided schema)
- prisma/seed.ts creating settings (VAT_RATE=7, WHT default=3, doc prefixes, company profile), sample customer, 3 units, 1 project, 1 order
- API routes + services + guards (month lock)
- UI pages (6 pages) with search/filter and print PDF button
- CSV export utilities
- Basic auth can be stubbed or simple (single-user) for v1

“งานที่ Cursor ต้องทำ” (Checklist ให้ทำตามลำดับ)

1. ตั้งโปรเจกต์ Next.js + Tailwind + shadcn + Prisma
2. הג schema ตามที่ให้ + migrate MariaDB
3. ทำ seed.ts (Setting/ลูกค้า/Units/ตัวอย่าง Order/Project)
4. สร้าง CRUD API: customers, units, orders, projects, documents, expenses, payments
5. เพิ่ม monthLock guard ให้ทุก mutation
6. ทำ Documents UI + ปุ่ม “Generate/Print PDF”
7. ทำ Puppeteer PDF route + templates 3 แบบ
8. ทำ Reports (CSV) 3 ชุด + UI download

9. ทำ Close Month UI + API

ถ้าครูใหญ่ต้องการให้ “ข้อมูล seed เป็นข้อมูลจริงของหจก.” ตอนนี้เลย

ส่งมาแค่ 4 บรรทัดก็พอครับ:

1. ชื่อหจก.
2. เลขผู้เสียภาษี
3. ที่อยู่สำหรับออกเอกสาร
4. สาขา (เช่น 00000)

ผมจะปรับ seed.ts ให้เป็น “ของจริง” ทันที (ส่วนลูกค้าตัวอย่าง/หน่วยย่อຍມทำเป็นแม่แบบให้เลย)

ให้มีฟังก์ชันเพื่อสรุประยารายจ่ายเป็นรายเดือนสรุปการนำส่งภาษีทั้งหมดที่จ่ายรายเดือนสรุปกำไรขาดทุน

ประจำเดือน

ได้ครับครูใหญ่ — เพิ่ม 3 ฟังก์ชันนี้ “จำเป็นมาก” และยังคงมินิมัลได้ โดยทำเป็น **Monthly Summary**

Service ที่ดึงจากข้อมูลจริงในระบบ (Documents/Expenses/WHT/Payments) แล้วสรุปเป็นรายเดือน พร้อม export/แสดงบน Dashboard

ด้านล่างคือส่วนที่ต้องเพิ่มเข้า PRD + วิธีคำนวณ + API ที่ Cursor ทำต่อได้เลย

เพิ่มฟังก์ชันสรุประยารายเดือน (**Monthly Finance Summary**)

A) ฟังก์ชันที่ต้องมี (รายเดือน/รายปี)

1. สรุประยารับ-รายจ่ายรายเดือน
2. สรุปภาษีทั้งหมดที่จ่าย (WHT) รายเดือน (เพื่อเตรียมนำส่ง)
3. สรุปกำไรขาดทุนประจำเดือน (P&L)

จุดสำคัญ: ใช้ “เอกสารภาษีจริง” เป็นฐาน

- รายรับ: จากเอกสาร TAX_INVOICE_RECEIPT (ภาษีขายเกิดแล้ว)
- รายจ่าย: จาก Expense
- WHT: จาก WithholdingTax ที่ผูกกับเอกสารขาย/บริการ

วิธีนี้ทำให้สรุป “ตรงกับการยื่นภาษี” มากรีดสุด

1) นิยามตัวเลข (คำนวนแบบขั้ดเจน)

1.1 รายรับ (Revenue)

Revenue (ไม่รวม VAT) = ผลรวม Document.subTotal

เฉพาะ Document.type = TAX_INVOICE_RECEIPT ภายในเดือนนั้น

1.2 VAT

- VAT Output = ผลรวม Document.vatAmount (TAX_INVOICE_RECEIPT)
- VAT Input = ผลรวม Expense.vatAmount เฉพาะ hasVat=true
- VAT Payable = VAT Output - VAT Input

1.3 รายจ่าย (Expense)

- Expense (ไม่รวม VAT) = ผลรวม Expense.subTotal
- Total cash out (รวม VAT) = ผลรวม Expense.grandTotal (ถ้าต้องการดูการเงินสด)

หมายเหตุ: ค่าใช้จ่ายบางรายการอาจ “ไม่มี VAT” ก็ยังนับเป็นรายจ่ายได้

1.4 WHT (ภาษีหัก ณ ที่จ่าย)

สำหรับการ “เตรียมนำส่ง/ติดตามเอกสาร”

- WHT Withheld = ผลรวม WithholdingTax.taxAmount ของเอกสารในเดือนนั้น
- WHT Base = ผลรวม WithholdingTax.baseAmount
- รายการนำส่งรายเดือน = list เอกสาร + ลูกค้า + เลขรับรอง + จำนวนเงินถูกหัก

เชิงบัญชี: WHT ที่ลูกค้าหักไว้คือ “ภาษีรอเครดิต” ของหจก.

แต่เชิงงานจริง: เราต้องสรุปไว้เพื่อ “รวมรวมใบรับรอง” และตรวจยอดถูกต้อง

2) กำไรขาดทุนรายเดือน (Monthly P&L) แบบมินิมัล

เราทำ P&L แบบ “ใช้งานจริง” โดยไม่ต้องมีสมุดรายวันให้กลับ

สูตร P&L v1

- Revenue = \sum Document.subTotal (TAX_INVOICE_RECEIPT)
- COGS = \sum Expense.subTotal เฉพาะหมวด

- INVENTORY_PURCHASE
- SHIPPING_OUT
- Gross Profit = Revenue - COGS
- OPEX = \sum Expense.subTotal หมวดอื่น ๆ
- Operating Profit = Gross Profit - OPEX

ไม่เอา VAT มาคิดเป็นรายได้/รายจ่ายใน P&L (VAT เป็นภาษีผ่านทาง)

3) API + UI (ให้ Cursor ทำต่อ)

3.1 เพิ่ม API: Monthly Summary

Endpoint

GET /api/reports/monthly-summary?year=2026&month=2

Response JSON (ตัวอย่าง)

```
{
  "period": { "year": 2026, "month": 2 },
  "revenue": { "subTotal": 250000.00, "vat": 17500.00, "grandTotal": 267500.00 },
  "expenses": { "subTotal": 120000.00, "vat": 3500.00, "grandTotal": 123500.00 },
  "vat": { "output": 17500.00, "input": 3500.00, "payable": 14000.00 },
  "wht": { "baseAmount": 200000.00, "taxAmount": 6000.00, "count": 8 },
  "pnl": {
    "revenue": 250000.00,
    "cogs": 45000.00,
    "grossProfit": 205000.00,
    "opex": 75000.00,
    "operatingProfit": 130000.00
  }
}
```

3.2 API: รายการ WHT เพื่อนำส่ง (รายละเอียด)

GET /api/reports/wht-detail?year=2026&month=2

ส่งกลับเป็นรายการ:

- customerName, customerTaxId
- documentNumber, issueDate
- baseAmount, taxAmount, rate
- certificateNo, certificateDate
- status (missing certificate? / completed)

UI จะโชว์เป็น “to-do list” ได้เลย: ให้รับรองหายกี่ใบ

4) เพิ่มตารางสรุป (Optional แต่ต้องมาก)

ถ้าอยากรีวิวและ “ล็อกตัวเลขตอนปิดเดือน” ให้เพิ่ม MonthlySummary เก็บ snapshot ตอนกด Close Month

```
model MonthlySummary {  
    id      String  @id @default(cuid())  
    year    Int  
    month   Int  
  
    revenueSubTotal  Decimal @db.Decimal(12,2)  
    revenueVat      Decimal @db.Decimal(12,2)  
  
    expenseSubTotal Decimal @db.Decimal(12,2)  
    expenseVat      Decimal @db.Decimal(12,2)  
  
    vatOutput       Decimal @db.Decimal(12,2)  
    vatInput        Decimal @db.Decimal(12,2)
```

```

vatPayable      Decimal @db.Decimal(12,2)

whtBaseAmount   Decimal @db.Decimal(12,2)
whtTaxAmount    Decimal @db.Decimal(12,2)

pnlRevenue      Decimal @db.Decimal(12,2)
pnlCogs         Decimal @db.Decimal(12,2)
pnlGrossProfit  Decimal @db.Decimal(12,2)
pnlOpex          Decimal @db.Decimal(12,2)
pnlOperatingProfit Decimal @db.Decimal(12,2)

generatedAt DateTime @default(now())

@@unique([year, month])
}

```

กติกา

- ถ้าเดือนถูกปิด (MonthlyClose) → สร้าง/อัปเดต MonthlySummary เป็น snapshot
- Dashboard เดือนปิดแล้วอ่านจาก Summary จะเร็วและนิ่ง

v1 ทำแบบ “คำนวณสุดจาก query” กีได้ก่อน แล้วค่อยทำ summary table ใน v1.1

5) UI เพิ่ม (แบบมินิมัล)

ในหน้า Dashboard/Finance เพิ่ม Tab:

- Monthly Summary**
 - รายรับ (ฐานภาษี, VAT)
 - รายจ่าย (ฐานภาษี, VAT)
 - VAT ต้องชำระ

- WHT ถูกหัก (จำนวนรายการ/ยอดรวม)
- P&L (Gross/Operating)

มีปุ่ม:

- Download CSV (สรุป)
- Download WHT detail CSV
- Close Month (ถ้ายังไม่ปิด)

6) Prompt เพิ่มให้ Cursor (เฉพาะฟังก์ชันสรุป)

เอาไปแปะต่อท้าย prompt เดิมได้เลย:

Add monthly finance summary features:

- Implement GET /api/reports/monthly-summary?year=&month=

Compute:

- revenue from Document where type=TAX_INVOICE_RECEIPT (sum subTotal, vatAmount)

- expenses from Expense (sum subTotal, vatAmount; hasVat=true for vatInput)

- vatPayable = vatOutput - vatInput

- wht from WithholdingTax joined to Document (sum baseAmount, taxAmount)

- P&L:

revenue = sum revenue subTotal

cogs = sum Expense.subTotal where category IN (INVENTORY_PURCHASE, SHIPPING_OUT)

opex = sum Expense.subTotal for other categories

operatingProfit = revenue - cogs - opex

- Implement GET /api/reports/wht-detail?year=&month=

Return list grouped by customer with certificate tracking (missing certificate flagged).

- Add Dashboard/Finance UI section to show Monthly Summary cards + P&L table and CSV downloads.

ถ้าครูใหญ่ต้องการให้ “สรุปกำไรขาดทุน” แยก ขายสินค้า vs งานบริการ ด้วย (จะเห็นชัดมากว่ากำไรจริงมาจากไหน)

ผู้ทำให้เพิ่มแบบมินิมัลได้โดยใส่ revenueStream ใน Document (GOODS/SERVICE) หรือแยกตาม Order/Project ที่ผูกเอกสารอยู่แล้วครับ
สรุปกำไรขาดทุน” แยก ขายสินค้า vs งานบริการ ด้วย

จัดให้ครับครูใหญ่  แบบมินิมัลแต่ “แยกกำไรชัด” ระหว่าง ขายสินค้า vs งานบริการ โดยไม่ต้องเพิ่มความซับซ้อนเยอะ

แนวทางที่ดีที่สุดสำหรับระบบเราคือ **ไม่ต้องเพิ่มฟิลด์ revenueStream ใหม่** เพราะเรามีความสัมพันธ์อยู่แล้ว:

- เอกสารที่ผูกกับ orderId ⇒ **ขายสินค้า/ออเดอร์เว็บ**
- เอกสารที่ผูกกับ projectId ⇒ **งานบริการ/โครงการ**

ถ้าบางเอกสารผูกได้ทั้งสองอย่าง (แทบไม่ควร) ให้บังคับในกฎว่า 1 เอกสารผูกได้อย่างใดอย่างหนึ่งเท่านั้น เพื่อความชัด

1) กติกาการจัดกลุ่มรายได้ (แยก Goods vs Service)

Revenue – Goods

Document.type = TAX_INVOICE_RECEIPT

และ document.orderId IS NOT NULL

Revenue – Service

Document.type = TAX_INVOICE_RECEIPT

และ document.projectId IS NOT NULL

2) การแยกต้นทุน/ค่าใช้จ่ายให้สองกลุ่ม (มินิมัล)

เพื่อให้ P&L แยกสองส่วนได้จริง เราใช้ “การผูกค่าใช้จ่าย” ที่มีอยู่แล้วใน Expense:

- Expense.relatedOrderId \Rightarrow เข้ากลุ่ม Goods
- Expense.relatedProjectId \Rightarrow เข้ากลุ่ม Service
- ถ้าไม่ผูกอะไรเลย \Rightarrow ถือเป็น Shared OPEX (ค่าใช้จ่ายร่วม)

UX ต้องมีตัวเลือก “ผูกกับอเดอร์/โครงการ” แบบ optional แต่แนะนำให้ผูกเมื่อทำได้ จะทำให้กำไรแยกแม่นขึ้นทุกเดือน

3) สูตรกำไรขาดทุนแยก 2 สาย + ส่วนร่วม

3.1 Goods P&L

- Goods Revenue = \sum doc.subTotal (orderId != null)
- Goods COGS = \sum expense.subTotal ที่ category \in (INVENTORY_PURCHASE, SHIPPING_OUT) และ relatedOrderId != null
- Goods OPEX = \sum expense.subTotal หมวดอื่น ๆ และ relatedOrderId != null
- Goods Operating Profit = Goods Revenue - Goods COGS - Goods OPEX

3.2 Service P&L

- Service Revenue = \sum doc.subTotal (projectId != null)
- Service Direct Cost (ถ้ามี) = \sum expense.subTotal ที่ relatedProjectId != null (รวมได้ทั้ง cost/opex)
- Service Operating Profit = Service Revenue - Service Direct Cost

3.3 Shared OPEX (ค่าใช้จ่ายร่วม)

- Shared OPEX = \sum expense.subTotal ที่ relatedOrderId is null AND relatedProjectId is null

แล้วกระจาย Shared OPEX เข้าสองฝ่ายแบบมินิมัล:

- วิธีแนะนำ v1: กระจายตาม “สัดส่วนรายได้” ของเดือนนั้น
 - goodsShare = goodsRevenue / (goodsRevenue + serviceRevenue)
 - serviceShare = 1 - goodsShare
 - goodsAllocatedOPEX = sharedOPEX * goodsShare

- o $\text{serviceAllocatedOPEX} = \text{sharedOPEX} * \text{serviceShare}$

สุดท้ายได้:

- Goods Net Operating = $\text{goodsOperatingProfit} - \text{goodsAllocatedOPEX}$
- Service Net Operating = $\text{serviceOperatingProfit} - \text{serviceAllocatedOPEX}$

ถ้าเดือนใดรายได้ผิ้งหนึ่งเป็น 0 ให้ยืน sharedOPEX ไปอีกผิ้งทั้งหมด หรือแสดงว่า “ยังไม่สามารถกระจายได้”
(แต่ผมแนะนำอยู่ไปผิ้งที่มีรายได้)

4) ปรับ Response ของ monthly-summary ให้รองรับแยกสาย

เพิ่ม `pnlByStream` ใน JSON

```
{
  "pnlByStream": {
    "goods": {
      "revenue": 120000.00,
      "cogs": 30000.00,
      "directOpex": 5000.00,
      "allocatedSharedOpex": 8000.00,
      "netOperatingProfit": 77000.00
    },
    "service": {
      "revenue": 130000.00,
      "directCost": 20000.00,
      "allocatedSharedOpex": 9000.00,
      "netOperatingProfit": 101000.00
    },
    "sharedOpex": 17000.00,
    "allocationMethod": "REVENUE_RATIO"
  }
}
```

```
 }  
 }
```

5) ปรับ UX (มินิมัล)

ในหน้า Monthly Summary เพิ่ม 1 ส่วน:

- Toggle: รวม / แยกสินค้า-บริการ
- ตาราง 2 คอลัมน์ (Goods | Service) + แฉ Shared OPEX + แฉ Net Profit

และปุ่ม:

- Export CSV (P&L by stream)
-

6) Prompt เพิ่มให้ Cursor (คัดลอกไปแปะได้เลย)

Enhance monthly P&L to split Goods vs Service:

Rules:

- Goods revenue: Document.type=TAX_INVOICE_RECEIPT and orderId != null
- Service revenue: Document.type=TAX_INVOICE_RECEIPT and projectId != null
- Goods costs/opex: Expense where relatedOrderId != null (split COGS categories

INVENTORY_PURCHASE, SHIPPING_OUT vs other categories)

- Service direct cost: Expense where relatedProjectId != null
- Shared OPEX: Expense where relatedOrderId is null and relatedProjectId is null

Allocate shared OPEX by monthly revenue ratio:

goodsShare = goodsRevenue / (goodsRevenue + serviceRevenue)

serviceShare = 1 - goodsShare

Handle divide-by-zero safely.

Update /api/reports/monthly-summary response to include pnlByStream as described.

Add UI toggle and table to display the split P&L and allocation method.

Add CSV export for P&L by stream.

จุดเล็ก ๆ ที่ทำให้ “แยกกำไร” แม่นขึ้นแบบไม่เพิ่มตาราง

เวลาเพิ่ม Expense ให้มีช่อง:

- “ผูกกับอโอดอร์” (dropdown)
- “ผูกกับโครงการ” (dropdown)
- ถ้าไม่เลือกทั้งคู่ = shared

แค่นี้กำไรแยกจะคอมปิวเตอร์