PART 2: DATABASE SCHEMA  
  
Create this Prisma schema in server/prisma/schema.prisma:  
  
generator client {

provider = "prisma-client-js"

}

datasource db {

provider = "postgresql"

url = env("DATABASE\_URL")

}

enum UserRole {

BOARD

MANAGEMENT

OWNER

ACCOUNTANT

ATTORNEY

}

enum LienStatus {

NONE

PRE\_LIEN

LIEN\_FILED

FORECLOSURE

}

enum InvoiceStatus {

PENDING\_APPROVAL

APPROVED

PAID

REJECTED

}

enum ViolationStatus {

OPEN

IN\_PROGRESS

RESOLVED

}

model User {

id String @id @default(uuid())

email String @unique

password String

firstName String

lastName String

phone String?

role UserRole

unitId String?

unit Unit? @relation(fields: [unitId], references: [id])

createdAt DateTime @default(now())

lastLogin DateTime?

transactions Transaction[]

auditLogs AuditLog[]

@@index([email])

}

model Unit {

id String @id @default(uuid())

unitNumber String @unique

ownerName String

squareFootage Int?

assessmentPercentage Float

monthlyMaintenance Float

monthlyReserve Float

monthlySpecial Float

balanceDue Float @default(0)

lastPaymentDate DateTime?

delinquentDays Int @default(0)

lateFees Float @default(0)

lienStatus LienStatus @default(NONE)

users User[]

transactions Transaction[]

ownerAccount OwnerAccount?

@@index([unitNumber])

}

model OwnerAccount {

id String @id @default(uuid())

unitId String @unique

unit Unit @relation(fields: [unitId], references: [id])

balanceDue Float @default(0)

lateFees Float @default(0)

attorneyFees Float @default(0)

lastPayment Float?

lastPaymentDate DateTime?

autoPayEnabled Boolean @default(false)

paymentPlan Boolean @default(false)

updatedAt DateTime @updatedAt

}

model Transaction {

id String @id @default(uuid())

unitId String

unit Unit @relation(fields: [unitId], references: [id])

type String // 'assessment', 'payment', 'late\_fee', 'adjustment'

amount Float

date DateTime

description String

createdBy String

user User @relation(fields: [createdBy], references: [id])

createdAt DateTime @default(now())

@@index([unitId, date])

}

model Vendor {

id String @id @default(uuid())

name String

serviceType String

contactEmail String?

contactPhone String?

monthlyCost Float?

contractExpiration DateTime?

invoices Invoice[]

createdAt DateTime @default(now())

}

model Invoice {

id String @id @default(uuid())

vendorId String

vendor Vendor @relation(fields: [vendorId], references: [id])

invoiceNumber String?

amount Float

dueDate DateTime

status InvoiceStatus @default(PENDING\_APPROVAL)

approvedBy String?

paidDate DateTime?

checkNumber String?

pdfUrl String?

description String?

createdAt DateTime @default(now())

@@index([vendorId, status])

}

model Document {

id String @id @default(uuid())

title String

category String // 'insurance', 'violation', 'financial', 'contract', 'meeting'

fileUrl String

uploadedBy String

expirationDate DateTime?

accessLevel String @default("board-only") // 'board-only', 'all-owners', 'public'

createdAt DateTime @default(now())

}

model Violation {

id String @id @default(uuid())

caseNumber String?

agency String

issueDescription String

fineAmount Float?

finePerDay Float?

status ViolationStatus @default(OPEN)

filedDate DateTime

dueDate DateTime?

resolvedDate DateTime?

createdAt DateTime @default(now())

}

model ManagementPerformance {

id String @id @default(uuid())

managementCompany String

metricType String // 'financial-report', 'response-time', 'vendor-payment'

targetValue Float

actualValue Float

measurementDate DateTime

status String // 'passing', 'failing', 'warning'

createdAt DateTime @default(now())

@@index([measurementDate])

}

model AuditLog {

id String @id @default(uuid())

userId String

user User @relation(fields: [userId], references: [id])

action String

tableName String?

recordId String?

oldValue Json?

newValue Json?

ipAddress String?

timestamp DateTime @default(now())

@@index([userId, timestamp])

}

model SystemSettings {

id String @id @default(uuid())

key String @unique

value String

}