Synapse CRM: 2025 Tech Stack Updates & Key Changes

Executive Summary

This document highlights the critical updates and changes when upgrading from older versions to **Next.js 16**, **NestJS 11**, **Prisma 6.18+**, and **Supabase (2025)**.

---

1. Next.js 16 (October 2025) - Major Breaking Changes

Key Updates

Feature	Old (v15)	New (v16)	Action Required
Bundler	Webpack (default, optional Turbopack)	Turbopack (stable, always-on)	No action; enjoy 2-5x faster builds
Async Params	`params.id` (sync)	`await params.id` (async)	Update ALL page components
Route Cache	`revalidateTag(tag)`	`revalidateTag(tag, profile?)`	Add optional profile parameter
Middleware File	`middleware.ts` (Edge)	`proxy.ts` (Node)	Rename file; update function exports
Parallel Slots	Optional `default.tsx`	Required `default.tsx`	Add default.tsx to all parallel route slots
React Version	React 19.0	React 19.2	Minor; no breaking changes for basic apps

Critical Migration Steps

1. Update Page Components for Async Params

OLD (Next.js 15):

```
export default function Page({ params }) {
  return <div>{params.id}</div>;
}
```

NEW (Next.js 16):

```
export default async function Page({ params }) {
  const id = (await params).id;  // MUST use await
  return <div>{id}</div>;
}
```

2. Rename `middleware.ts` to `proxy.ts` (if you have it)

- Change filename: `middleware.ts` → `proxy.ts`
- Update structure to use `Proxy` type instead of `Middleware`
- Proxy runs in Node runtime (not Edge), so you can use Node APIs

3. Cache Revalidation Changes

OLD:

```
revalidateTag('posts');
```

NEW:

```
revalidateTag('posts', { revalidate: 60 }); // Optional: add cache profil
```

---

2. NestJS 11 (January 2025) - HTTP Adapter & Logging

Key Updates

Feature	Old (v10)	New (v11)	Action Required
Express	v4 (default)	v5 (default)	No action; automatic upgrade
Fastify	v4	v5	Update version, test endpoints
Wildcard Routes	`@Get('/*')`	`@Get('/*splat')`	Rename wildcard routes
JSON Logging	Manual setup	Built-in `ConsoleLogger`	Use new logger config

Critical Migration Steps

1. Update Wildcard Routes

OLD:

```
@Get('/*')
serveStatic() {
  return 'catch-all';
}
```

NEW:

```
@Get('/*splat')  // Must name the wildcard
serveStatic() {
  return 'catch-all';
}
```

2. Use Built-in JSON Logging

NEW (NestJS 11):

```
import { Logger } from '@nestjs/common';

const app = await NestFactory.create(AppModule, {
  logger: new Logger({
    json: true,  // Enable JSON logging
    colors: false, // No colors in JSON
  }),
});
```

---

3. Prisma 6.18+ (October 2025) - Config File & Runtime

Key Updates

Feature	Old (v5)	New (v6.18+)	Action Required
Config	`prisma` section in `package.json`	`prisma.config.ts` (TypeScript)	Create `prisma.config.ts`

Feature	Old (v5)	New (v6.18+)	Action Required
Direct URL	Not required	REQUIRED for migrations	Add `DIRECT_URL` env var
Runtime Options	`node`, `deno-deploy`, `vercel`	`nodejs`, `deno`, `bun`, `workerd`, `vercel-edge`	Update runtime names
Seeding	CLI flag `-s`	Config file `seed` property	Move seed command to config
Client Output	Fixed at `node_modules`	Configurable in config	Optional; default is fine

Critical Migration Steps

1. Create `prisma.config.ts` (NEW)

```
import path from "node:path";
import { defineConfig, env } from "prisma/config";

export default defineConfig({
  schema: path.join("prisma", "schema.prisma"),
  migrations: {
    path: path.join("prisma", "migrations"),
    seed: "tsx prisma/seed.ts",  // Define seed here
  },
  engine: "classic",
  datasource: {
    url: env("DATABASE_URL"),
    directUrl: env("DIRECT_URL"),  // REQUIRED for migrations
  },
});
```

2. Update Environment Variables

REQUIRED in `.env`:

```
# Two database URLs required for connection pooling
DATABASE_URL="postgresql://pooler-url"   # Use pooler for app
DIRECT_URL="postgresql://direct-url"     # Use direct for migrations
```

3. Update Runtime in `prisma/schema.prisma`

OLD:

```
generator client {
  provider = "prisma-client-js"
  engine   = "node"  # Deprecated
}
```

NEW:

```
generator client {
  provider = "prisma-client-js"
  engine   = "classic"  # Use "classic"
  output   = "./generated/client"  # Optional
}
```

4. Move Seed Command from `package.json` to Config

OLD (package.json):

```
{
  "prisma": {
    "seed": "node prisma/seed.js"
  }
}
```

NEW (prisma.config.ts):

```
export default defineConfig({
  migrations: {
    seed: "tsx prisma/seed.ts",  // Moved here
  },
});
```

---

4. Supabase (2025 Updates)

Key Updates

Feature	Old	New 2025	Action

Feature	Old	New 2025	Action
Clerk Integration	Manual OAuth setup	Native Clerk provider (GA)	Update Auth config
Connection Pooling	PgBouncer optional	Recommended for production	Configure Direct + Pooler URLs
Realtime Auth	Supabase Auth only	Supabase Auth OR Clerk	Use Clerk for faster setup
Edge Functions	CLI + deploy	Dashboard + CLI + API	Use Dashboard for quick deploys
Postgres Version	15.1+	15.4+ or 16+	Upgrade available in settings

Critical Setup Steps

1. Use Direct URL + Pooler (for Supabase + Prisma)

In Supabase Dashboard:

```
Database > Connection String > Session pooler (application)
Database > Connection String > Direct connection
```

`.env`:

```
DATABASE_URL="postgresql://user:pass@aws-0-us-east-1.pooler.supabase.com:
DIRECT_URL="postgresql://user:pass@aws-0-us-east-1.db.supabase.com:5432/p
```

2. Configure Clerk as Third-Party Auth (if using Supabase Auth)

In Supabase > Auth > Providers > Clerk:

- Add Clerk API Key
- Configure JWT template if needed

---

5. Integrated Stack Compatibility Matrix

Component	Version	Status	Node Version	Notes
Next.js	16	GA	20.9+	Turbopack default
NestJS	11	GA	20.9+	Express v5 default
Prisma	6.18+	GA	20.9+	TypeScript-first config

Component	Version	Status	Node Version	Notes
Supabase	2025	GA	20.9+	Clerk integration native
Clerk	Latest	GA	Any	Works with Next.js 16 & NestJS 11
TypeScript	5.1+	GA	Any	Required

---

6. Breaking Changes Summary (What Will Break Without Updates)

Frontend (Next.js 16)

❌ **Will break:**

```
// Page using sync params (WRONG in Next.js 16)
export default function Page({ params }) {
  return params.id;  // ERROR: params is now Promise
}
```

✅ **Fix:**

```
export default async function Page({ params }) {
  const id = (await params).id;
  return id;
}
```

Backend (NestJS 11)

❌ **Will break:**

```
@Get('/*')  // ERROR: Wildcard must be named
catchAll() {}
```

✅ **Fix:**

```
@Get('/*splat')  // Correct
catchAll() {}
```

Database (Prisma 6.18+)

❌ **Will break:**

```
# Using old prisma.schema config
npx prisma db push  # ERROR: No config file found
```

✅ **Fix:**

```
# Create prisma.config.ts first
npx prisma db push  # Works
```

---

7. Development Environment Setup (2025)

Required Versions

```
node --version        # v20.9.0+
npm --version         # v10.0.0+
npx -v                # v10.0.0+
git --version         # Any recent version
```

Recommended Global Packages

```
npm install -g @nestjs/cli@latest
npm install -g create-next-app@latest
npm install -g prisma@latest
```

---

8. Common Migration Errors & Solutions

Error 1: "Cannot use 'await' in non-async function" (Next.js 16)

Cause: Page component not marked as async

Solution:

```
// Add async keyword
export default async function Page({ params }) {
  const id = (await params).id;
  return id;
}
```

Error 2: "Prisma schema requires directUrl when using connection pooler"

Cause: Missing `DIRECT_URL` in `.env`

Solution:

```
DATABASE_URL="postgresql://...pooler..."  # Pooler for app
DIRECT_URL="postgresql://...direct..."    # Direct for migrations
```

Error 3: "Express 5 wildcard routes must be named"

Cause: Using `@Get('/*')` instead of `@Get('/*splat')`

Solution:

```
@Get('/*splat')  // Name the wildcard
catchAll(@Param('*splat') splat: string) {}
```

Error 4: "Cannot find prisma config"

Cause: Missing `prisma.config.ts`

Solution:

```
cd synapse-crm-backend
# Create prisma.config.ts (see section 3.1)
npx prisma generate
npx prisma db push
```

---

9. Performance Improvements (2025 Stack)

Metric	Improvement	Impact
Build Speed	2-5x faster (Turbopack)	Dev iteration 5-10 mins → 1-2 mins
API Response	NestJS 11 + Prisma 6.18+	10-15% improvement over v10
Database	Direct + Pooler setup	50-100ms latency reduction
Bundle Size	React 19.2 optimizations	5-10% smaller JS bundles

---

10. Deployment Checklist (2025 Stack)

Backend (NestJS 11 + Prisma 6.18+ on Vercel/Railway)

- [ ] Set `DATABASE_URL` (pooler) and `DIRECT_URL` (direct)
- [ ] Set `CLERK_SECRET_KEY`
- [ ] Set `NODE_ENV=production`
- [ ] Run migrations: `npx prisma db push`
- [ ] Enable CORS for frontend domain
- [ ] Test endpoints with Clerk token

Frontend (Next.js 16 on Vercel)

- [ ] Set `NEXT_PUBLIC_CLERK_PUBLISHABLE_KEY`
- [ ] Set `NEXT_PUBLIC_API_BASE_URL` (production API)
- [ ] Configure Clerk redirect URIs for production domain
- [ ] Test async params in production build
- [ ] Verify Turbopack build completes

Resources & Links

- **Next.js 16 Docs:** https://nextjs.org/docs
- **Next.js 16 Breaking Changes:** https://nextjs.org/docs/app/guides/upgrading/version-16
- **NestJS 11 Guide:** https://trilon.io/blog/announcing-nestjs-11-whats-new
- **Prisma 6.18 Changelog:** https://www.prisma.io/changelog
- **Supabase Docs:** https://supabase.com/docs
- **Clerk Docs:** https://clerk.com/docs

Migration Timeline (Recommended)

Week	Task	Owner
Week 1	Study breaking changes, prepare environment	Dev Team
Week 2	Migrate backend (NestJS 11 + Prisma 6.18+)	Backend
Week 3	Migrate frontend (Next.js 16)	Frontend
Week 4	Integration testing, deployment	Full Team

Last Updated: October 31, 2025

Version: 1.0

Compatibility: Next.js 16, NestJS 11, Prisma 6.18+, Supabase 2025