

# Pose & Poise — Product Roadmap

**Version:** 1.0

**Last Updated:** December 2025

**Domain:** poseandpoise.studio

---

## Product Vision

A portfolio platform where models can showcase their work, generate professional comp cards, and get discovered by agencies and brands — all in one place.

---

## Tech Stack

Layer	Choice	Rationale
Framework	Next.js 15 (App Router)	Best-in-class React framework, Vercel-native
Language	TypeScript	Type safety, better DX
Styling	Design tokens + CSS	Custom brand aesthetic, no framework bloat
Database	Supabase (Postgres)	Realtime, RLS, generous free tier
Storage	Supabase Storage	Unified platform, simple API
Auth	Supabase Auth	Native RLS integration, cost-effective
Payments	Stripe	Industry standard, excellent docs
Email	Resend	Modern, developer-friendly
Hosting	Vercel	Seamless Next.js deployment
Analytics	PostHog	Product analytics + session replay
Image CDN	Supabase + imgix (later)	Transform/optimize on the fly

---

## Phase 1: Foundation

# Phase 1: Foundation

**Timeline:** Week 1-2

**Goal:** Users can sign up, log in, and access an authenticated dashboard.

## Tasks

### Project setup

- Initialize Next.js 15 with TypeScript
- Configure Biome (linting + formatting)
- Set up project structure (`src/` directory)
- Configure path aliases (`[@/*]`)
- Enable React Compiler

### Supabase setup

- Create Supabase project
- Define initial database schema (profiles, photos)
- Configure Row Level Security (RLS) policies
- Set up storage buckets (avatars, portfolio-photos)
- Configure storage policies

### Authentication

- Integrate Supabase Auth with Next.js
- Create auth UI (sign up, sign in, forgot password)
- Protected route middleware
- Auth context/hooks for client components
- Email verification flow

### Dashboard shell

- Authenticated layout with navigation
- Empty state for new users
- Basic responsive sidebar/header
- Match landing page design tokens

### Email capture (landing page)

- Connect existing form to Supabase
- Store subscribers in `waitlist` table
- Send welcome email via Resend

## Deliverables

1. Working auth flow (sign up → verify email → dashboard)
2. Protected dashboard route
3. Database with initial schema
4. Landing page form connected to backend

## Schema: Phase 1

```
sql  
  
-- Waitlist (from landing page)  
create table waitlist (  
    id uuid default gen_random_uuid() primary key,  
    email text unique not null,  
    created_at timestamp with time zone default now()  
);  
  
-- Profiles (extends Supabase auth.users)  
create table profiles (  
    id uuid primary key references auth.users(id) on delete cascade,  
    username text unique,  
    email text,  
    display_name text,  
    avatar_url text,  
    created_at timestamp with time zone default now(),  
    updated_at timestamp with time zone default now()  
);  
  
-- Auto-create profile on signup  
create or replace function handle_new_user()  
returns trigger as $$  
begin  
    insert into public.profiles (id, email)  
    values (new.id, new.email);  
    return new;  
end;  
$$ language plpgsql security definer;  
  
create trigger on_auth_user_created  
    after insert on auth.users  
    for each row execute function handle_new_user();
```

---

## Phase 2: Core Product

**Timeline:** Week 3-5

**Goal:** A model can create and share a complete portfolio.

### Tasks

#### Profile builder

- Edit display name, bio, avatar
- Model stats (height, bust, waist, hips, shoe size)
- Physical attributes (hair color, eye color)
- Location, agency affiliation
- Social links (Instagram, TikTok)
- Username selection (for public URL)

#### Photo management

- Drag-and-drop upload (multiple files)
- Image preview before upload
- Progress indicators
- Automatic thumbnail generation
- Reorder photos (drag to sort)
- Add captions
- Delete photos
- Image compression/optimization

#### Public portfolio page

- Route: `poseandpoise.studio/[username]`
- Display profile info + stats
- Photo gallery (grid or masonry)
- Contact/inquiry button
- Social links
- SEO optimization (meta tags, OG images)

#### Comp card generator

- Select photos for comp card (typically 4-6)
- Auto-populate stats from profile
- Single template (MVP)
- Preview comp card
- Share link to comp card view

## Deliverables

1. Complete profile editing UI
2. Photo upload with drag-drop and reordering
3. Public portfolio page (`/username`)
4. Basic comp card generation

## Schema: Phase 2

```
sql
-- Extend profiles with model stats
alter table profiles add column if not exists
  bio text,
  height_cm integer,
  bust_cm integer,
  waist_cm integer,
  hips_cm integer,
  shoe_size text,
  hair_color text,
  eye_color text,
  location text,
  agency text,
  instagram text,
  tiktok text,
  website text,
  is_public boolean default false;
```

```
-- Portfolio photos
create table photos (
  id uuid default gen_random_uuid() primary key,
  profile_id uuid references profiles(id) on delete cascade not null,
  url text not null,
  thumbnail_url text,
  caption text,
```

```
sort_order integer default 0,  
  
width integer,  
height integer,  
size_bytes integer,  
created_at timestamp with time zone default now()  
);  
  
create index photos_profile_id_idx on photos(profile_id);  
create index photos_sort_order_idx on photos(profile_id, sort_order);  
  
-- Comp cards  
create table comp_cards (  
    id uuid default gen_random_uuid() primary key,  
    profile_id uuid references profiles(id) on delete cascade not null,  
    name text default 'My Comp Card',  
    template text default 'classic',  
    photo_ids uuid[] not null,  
    is_primary boolean default false,  
    created_at timestamp with time zone default now(),  
    updated_at timestamp with time zone default now()  
);  
  
create index comp_cards_profile_id_idx on comp_cards(profile_id);
```

## RLS Policies: Phase 2

sql

-- Profiles: users can read public profiles, edit own  
alter table profiles enable row level security;

```
create policy "Public profiles viewable by everyone"  
on profiles for select  
using (is_public = true);
```

```
create policy "Users can view own profile"  
on profiles for select  
using (auth.uid() = id);
```

```
create policy "Users can update own profile"  
on profiles for update  
using (auth.uid() = id);
```

-- Photos: users can manage own, public profiles show photos  
alter table photos enable row level security;

```
create policy "Photos viewable on public profiles"  
on photos for select  
using (  
exists (  
select 1 from profiles  
where profiles.id = photos.profile_id  
and profiles.is_public = true  
)  
);
```

```
create policy "Users can view own photos"  
on photos for select  
using (auth.uid() = profile_id);
```

```
create policy "Users can insert own photos"  
on photos for insert  
with check (auth.uid() = profile_id);
```

```
create policy "Users can update own photos"  
on photos for update  
using (auth.uid() = profile_id);
```

```
create policy "Users can delete own photos"  
on photos for delete
```

```
using (auth.uid() == profile_id);
```

## Phase 3: Growth Features

**Timeline:** Week 6-8

**Goal:** Features that increase engagement, sharing, and stickiness.

### Tasks

#### Analytics dashboard

- Portfolio view count
- Unique visitors
- Views over time (chart)
- Top referrers
- Geographic breakdown
- Comp card views/downloads

#### Multiple comp card templates

- Classic (2x2 grid)
- Editorial (asymmetric)
- Minimal (single focus image)
- Agency (traditional format)
- Template preview/selection UI

#### PDF export

- Generate print-ready PDF from comp card
- Standard sizes (8.5x11, A4, 5x7)
- High-resolution export
- Download button

#### Custom domains

- Users can connect `portfolio.modelname.com`
- Vercel domain configuration API
- SSL provisioning
- Domain verification flow

#### Social optimization

- Dynamic OG images (personalized preview)
- Twitter card optimization
- Copy-to-clipboard share links
- QR code generation

## Inquiry/contact form

- Allow visitors to send messages
- Email notifications to model
- Inquiry management in dashboard
- Spam protection (rate limiting, captcha)

## Deliverables

1. Analytics dashboard with charts
2. 4+ comp card templates
3. PDF download functionality
4. Custom domain support
5. Enhanced social sharing

## Schema: Phase 3

```
sql

-- Portfolio analytics
create table portfolio_views (
    id uuid default gen_random_uuid() primary key,
    profile_id uuid references profiles(id) on delete cascade not null,
    visitor_id text, -- anonymous identifier
    referrer text,
    country text,
    city text,
    device_type text,
    viewed_at timestamp with time zone default now()
);

create index portfolio_views_profile_id_idx on portfolio_views(profile_id);
create index portfolio_views_viewed_at_idx on portfolio_views(viewed_at);

-- Comp card templates
create table comp_card_templates (
```

```

create table comp_card_templates (
    id text primary key, -- 'classic', 'editorial', etc,
    name text not null,
    description text,
    thumbnail_url text,
    is_premium boolean default false,
    sort_order integer default 0
);

-- Custom domains
create table custom_domains (
    id uuid default gen_random_uuid() primary key,
    profile_id uuid references profiles(id) on delete cascade not null,
    domain text unique not null,
    verified boolean default false,
    verification_token text,
    created_at timestamp with time zone default now()
);

-- Inquiries
create table inquiries (
    id uuid default gen_random_uuid() primary key,
    profile_id uuid references profiles(id) on delete cascade not null,
    sender_name text not null,
    sender_email text not null,
    message text not null,
    is_read boolean default false,
    created_at timestamp with time zone default now()
);

create index inquiries_profile_id_idx on inquiries(profile_id);

```

## Phase 4: Monetization

**Timeline:** Week 8-10

**Goal:** Sustainable revenue through subscriptions.

### Pricing Tiers

Tier	Price	Features
Free	\$0/mo	10 photos, 1 comp card, subdomain only, basic stats

**Pro**      \$12/mo    Unlimited photos, all templates, custom domain, full analytics, PDF export, priority support

---

**Agency**    \$49/mo    Everything in Pro + 10 model profiles, team management, white-label option, API access

---

## Tasks

### Stripe integration

- Products and prices in Stripe dashboard
- Checkout session creation
- Customer portal (manage subscription)
- Webhook handling (subscription events)

### Subscription management

- Current plan display
- Upgrade/downgrade flow
- Billing history
- Cancel subscription

### Feature gating

- Check subscription tier for features
- Graceful upgrade prompts
- Usage limits (photo count)
- Feature flags system

### Agency features

- Multi-model dashboard
- Team invitations
- Role permissions (admin, editor, viewer)
- Centralized billing

### Onboarding optimization

- Free trial (14 days Pro)
- Onboarding checklist
- Email drip campaign

- In-app upgrade prompts

## Deliverables

1. Stripe checkout integration
2. Customer billing portal
3. Feature gating by tier
4. Agency multi-model support
5. Trial and conversion flow

## Schema: Phase 4

```
sql

-- Subscriptions (synced from Stripe)
create table subscriptions (
    id uuid default gen_random_uuid() primary key,
    profile_id uuid references profiles(id) on delete cascade not null,
    stripe_customer_id text unique,
    stripe_subscription_id text unique,
    plan_id text not null default 'free', -- 'free', 'pro', 'agency'
    status text not null default 'active', -- 'active', 'canceled', 'past_due'
    current_period_start timestamp with time zone,
    current_period_end timestamp with time zone,
    cancel_at_period_end boolean default false,
    created_at timestamp with time zone default now(),
    updated_at timestamp with time zone default now()
);

create index subscriptions_profile_id_idx on subscriptions(profile_id);
create index subscriptions_stripe_customer_id_idx on subscriptions(stripe_customer_id);

-- Usage tracking (for limits)
create table usage (
    id uuid default gen_random_uuid() primary key,
    profile_id uuid references profiles(id) on delete cascade not null,
    photo_count integer default 0,
    comp_card_count integer default 0,
    updated_at timestamp with time zone default now()
);

-- Plan limits reference
```

```
-- Plan limits reference
create table plan_limits (
    plan_id text primary key,
    max_photos integer,
    max_comp_cards integer,
    custom_domain boolean,
    pdf_export boolean,
    analytics_full boolean,
    api_access boolean
);

insert into plan_limits values
('free', 10, 1, false, false, false, false),
('pro', -1, -1, true, true, true, false),
('agency', -1, -1, true, true, true, true);

-- Agency: team members
create table team_members (
    id uuid default gen_random_uuid() primary key,
    agency_profile_id uuid references profiles(id) on delete cascade not null,
    member_profile_id uuid references profiles(id) on delete cascade not null,
    role text default 'model', -- 'admin', 'editor', 'model'
    invited_at timestamp with time zone default now(),
    accepted_at timestamp with time zone,
    unique(agency_profile_id, member_profile_id)
);
```

## Stripe Webhook Events to Handle

```
typescript

// src/app/api/webhooks/stripe/route.ts

const relevantEvents = [
    'checkout.session.completed', // New subscription
    'customer.subscription.updated', // Plan change
    'customer.subscription.deleted', // Cancellation
    'invoice.payment_failed', // Payment issue
    'invoice.payment_succeeded', // Renewal success
];
```

---

## Future Considerations (Post-Launch)

## Phase 5+

- **Mobile app** — React Native for iOS/Android
- **AI features** — Auto-tagging, background removal, enhancement
- **Marketplace** — Agencies browse/discover models
- **Booking system** — Calendar, availability, job management
- **Contracts** — Digital signing for model releases
- **Invoicing** — Built-in payment requests

## Technical Debt to Address

- Move inline styles to CSS modules or Tailwind
- Implement proper error boundaries
- Add comprehensive testing (Playwright, Vitest)
- Set up CI/CD pipeline
- Monitoring and alerting (Sentry)

---

## Key Metrics to Track

Metric	Target (Month 3)
Signups	500
Active portfolios	200
Conversion to Pro	5%
MRR	\$500
Churn rate	<5%

---

## Resources

- [Supabase Docs](#)

Supabase Does

- [Stripe Subscriptions](#)
  - [Next.js App Router](#)
  - [Vercel Domains API](#)
  - [Resend](#)
- 

## Changelog

Date	Version	Changes
Dec 2025	1.0	Initial roadmap