Neurose Virtual Fitting Room (VFR) – Technical Overview & System Blueprint

---

1. Project Vision

The Neurose Virtual Fitting Room (VFR) enables users to realistically visualize garments on themselves or chosen models using a hybrid AI pipeline that combines **geometry-aware 3D simulation**, **AI diffusion rendering**, and **photo-realistic garment synthesis**. The system aims to achieve **Kling AI-level realism** while maintaining scalability and cost-efficiency.

---

2. Core Objectives

1. Deliver ultra-realistic garment try-on experiences without CAD models.
2. Integrate Kling AI as a premium rendering backend for ultimate realism.
3. Maintain a full local fallback pipeline achieving comparable quality.
4. Support variable pricing through pay-as-you-go and tiered quality levels.
5. Provide API and SDK access for retailers and e-commerce platforms.

---

3. Key Functional Modules

3.1 User Interface (Front-End)

- Web and mobile app (React + Flutter)
- User photo upload & privacy consent
- Garment selection from retailer catalog
- Real-time preview and rendering queue
- User account management, subscription, and history

3.2 Backend API (FastAPI / Node.js)

- User management & authentication (Firebase Auth)
- Image upload & preprocessing service
- Rendering job queue (Redis + Celery)
- Result delivery & caching layer (PostgreSQL + S3)
- REST/GraphQL endpoints for partners

3.3 AI Rendering Pipeline

Hybrid Architecture: Local + Kling AI integration.

Stage	Model	Purpose	Cost
Person Parsing	SCHP / CIHP / LIP	Segment body & clothes	Free
Pose Extraction	OpenPose / YOLOv8-Pose / ControlNet	Extract joints	Free
Garment Warping	StableVITON / GP-VTON / ReclothVITON	Align garment to body	GPU cost only
Geometry Fitting	SMPL-X + Cloth Simulation	Physically accurate drape	GPU cost only
Final Rendering	Kling AI API	Photo-realistic result	Pay-per-image
Local Fallback	Flux / SDXL + IP-Adapter + InstantID	Kling-grade rendering	GPU cost only

4. Advanced Stack (Tier-S Hybrid System)

4.1 Person Canonicalization

- SMPL-X or GHUM model fit from user photo(s)
- Neural texture extraction for user identity
- Segmentation-based hair/skin protection masks

4.2 Garment Assetization

- Multi-photo garment reconstruction (front, side, detail)
- UV unwrapping and neural texture mapping
- Fabric classification for draping presets

4.3 Differentiable Draping

- Coarse cloth simulation with stiffness presets
- Collision & contact correction refinement
- Real-time pose-based garment fitting

4.4 Photoreal Finisher

- Gaussian-splat rendering for lighting & shading
- SDXL/Flux img2img polish (denoise 0.10–0.22)
- Multi-ControlNet (Pose + Depth + Normal + Seg + Edge)
- IP-Adapter (Garment) + InstantID/FaceID-Plus for identity and fabric detail

4.5 Post-Processing

- SDXL Refiner pass

- Real-ESRGAN upscale → downsample
- CodeFormer (face restoration)
- BGMv2 matting + relight blending

4.6 Quality Assurance & Retry Logic

- CLIP & LPIPS garment similarity
- Identity retention score
- Auto-retry with modified parameters
- Kling escalation on double-failure

5. Infrastructure Requirements

5.1 Hardware

Component	Minimum	Recommended
GPU	RTX 3090 / A6000	RTX 4090 / A100
VRAM	24 GB	48 GB+
CPU	i9 / Ryzen 9	Threadripper / Xeon
RAM	64 GB	128 GB
Storage	2 TB NVMe	4 TB NVMe SSD
Cloud Option	GCP / Vast.ai	AWS EC2 G6e / Lambda Labs

5.2 Software Stack

- OS: Ubuntu 22.04 LTS
- Backend: FastAPI, Celery, Redis, PostgreSQL
- AI: PyTorch 2.2+, Diffusers, ControlNet, IP-Adapter, InstantID, SMPL-X
- Cloud Integration: Firebase, S3, Kling AI API
- Frontend: React.js / Flutter
- Deployment: Docker Compose / Kubernetes (multi-GPU scaling)

6. Data Flow Overview

1. User uploads image(s) and selects garment.
2. Preprocessing: segmentation, pose, depth, and garment parsing.
3. If Kling quota available → send to Kling API.
4. If not → run local 2.5D/3D hybrid fallback pipeline.
5. Post-process output → store in cache → deliver to user.
6. Training data (Kling pairs) periodically used for LoRA distillation.

7. Functional Highlights

- Variable pricing & quality tiers (Basic, Pro, Ultra)
- Real-time rendering queue monitoring
- Retailer dashboard for SKU management
- Cache reuse (hash(user, garment, pose))
- Multi-user batch rendering optimization
- Model auto-updates via modular config registry

8. Future Enhancements

1. **Full 3D garment digital twin** generation from 2–3 photos.
2. **Virtual fitting avatars** with real-time animation for AR/VR use.
3. **Voice & gesture interface** integration using Neurose VLA stack.
4. **Personalized body-shape estimation** for size recommendation.
5. **Decentralized rendering mesh** (distributed GPU network for scale).

9. Summary

The Neurose Virtual Fitting Room fuses physics-based realism with neural rendering and modular scalability. By integrating Kling AI selectively and maintaining an advanced fallback stack, it achieves premium realism at optimized cost.

This document serves as the **technical and functional reference** for all development, infrastructure setup, and partnership discussions.

10. Model Registry & Pipeline Specs (Authoritative)

This section is the **single source of truth** for all AI models used in VFR, including purpose, inputs/outputs, and where they sit in the pipeline. Teams must keep this table and the configs below in sync with deployments.

10.1 End-to-End Pipeline (Top-Level)

1) **Pre-ingest** → file checks, EXIF strip, PII guard. 2) **User Canonicalization** → SMPL-X fit + neural texture + UV masks. 3) **Garment Assetization** → multi-photo mesh + UV + fabric tag. 4) **Draping** → cloth sim (coarse) + differentiable refinement, collisions. 5) **Photoreal Finisher** → (A) Gaussian-splat render → SDXL/Flux img2img polish; or (B) SDS UV texture bake (per SKU pose family). 6) **Post-processing** → Refiner, upscaler, relight, matting, seam clean. 7) **QA & Retry** → metrics checks; retry once; escalate to Kling if fail. 8) **Cache & Deliver** → CDN/S3, dedupe on (user, garment, pose) hash.

10.2 Model Registry (Table)

Module	Preferred Models	Framework	Inputs	Outputs	Notes
Parsing (person/ clothes/hair/ skin)	SCHP (CIHP/LIP alt)	PyTorch	RGB image	Segmentation masks (person/ garment/skin/ hair)	Use FP16; export to ONNX optional
Pose	YOLOv8-Pose (OpenPose alt)	PyTorch	RGB image	2D keypoints (17/25 set)	Fast and robust; cache per user image
Depth	ZoeDepth (MiDaS alt)	PyTorch	RGB image	Depth map	For ControlNet-Depth & silhouette carving
Normals (optional)	NormalBae / ControlNet-Normal	Diffusers	RGB + depth	Normal map	Boosts seam realism in finisher
Edge/Seams	HED / Lineart	PyTorch	RGB	Edge map	Guides collars/ hem lines
Identity	InstantID **and** IP-Adapter FaceID-PlusV2	Diffusers	Face crop + reference	Face embedding/ adapter features	Use both for stability + fidelity
Garment Style Adapter	IP-Adapter (Image-Plus)	Diffusers	Garment photo	Style/texture features	Drives fabric micro-detail
SMPL-X Fitting	SMPL-X + PIXIE/ SMPLify-X	PyTorch	User image(s), keypoints	Body mesh, pose, UV	Persist per user/ session
Garment Mesh from Photos	Multi-view recon: silhouette carving + ZoeDepth priors	PyTorch	2–3 garment photos	Coarse mesh + UV	No CAD required
UV Inpainting	LaMa (on UV)	PyTorch	Partial UV	Completed UV texture	Fill occlusion gaps
Fabric Classifier	Lightweight CNN (custom)	PyTorch	Garment crop	{denim, knit, satin, leather, printed}	Selects drape + BRDF preset

Module	Preferred Models	Framework	Inputs	Outputs	Notes
Draping (Coarse)	Fast cloth sim (Taichi/ARCSim-lite or NVIDIA Flex alt)	CUDA	Body mesh + garment mesh + pose	Draped mesh	10–30 ms target
Draping (Refine)	Differentiable refinement (projective, contact losses)	PyTorch/ CUDA	Draped mesh + masks	Collision-free, thickness-aware mesh	Corrects sleeve twist/collar lift
Gaussian-Splat Renderer	3D Gaussian Splatting	CUDA	Mesh + textures	Soft render (RGB+A)	Fast lighting/ shadows
Finisher (img2img)	**SDXL** or **Flux-1** + ControlNets	Diffusers	Base render + controls + adapters	Photo-real image	Denoise 0.10–0.24, steps 34–42
ControlNets	Pose, Depth, Seg, Normals, Edge, (Tile optional)	Diffusers	Above maps	Guidance features	Weights: Pose . 55, Depth .65, Seg .75, Normal . 45, Edge .35
Refiner	SDXL-Refiner	Diffusers	Finisher output	Polished image	Denoise 0.10–0.18
Upscaler	Real-ESRGAN x4 / SwinIR	PyTorch	Image	Upscaled image	Downsample to target after
Face Restore (if needed)	CodeFormer	PyTorch	Image	Face-enhanced image	Weight 0.5–0.7
Matting	BGMv2	PyTorch	Image	Alpha matte	Clean edges for composite
Relight	LUT/Light estimation (custom)	PyTorch	Image + bg	Relit image	Match original scene
Seam Cleanup	Poisson/Seamless clone	OpenCV	Image + mask	Artifact-free seams	Hem/collar fix
QA Metrics	CLIPScore, LPIPS, Aesthetic predictor	PyTorch	Image + refs	Scores	Thresholds below
Premium Backend	**Kling AI**	API	User + garment inputs	Ultra-real image	Use when escalated or for hero shots

10.3 Finisher (Img2Img) Baseline Config

```
IMG2IMG_DENOISE=0.20
STEPS=38
CFG=6.2
RES_LONG=1344
CTRL_POSE=0.55
CTRL_DEPTH=0.65
CTRL_NORMAL=0.45
CTRL_SEG=0.75
CTRL_EDGE=0.35
ADAPT_GARMENT=1.00
ADAPT_FACEID=0.85
REFINER_DENOISE=0.14
TILEPASS_DENOISE=0.12
TILEPASS_STEPS=16
```

10.4 Inputs/Outputs (I/O Contracts)

- **User Canonicalization**
- In: RGB image(s) (min 1024 px long side)
- Out: SMPL-X mesh (OBJ/NPZ), UV texture (2048^2 PNG), masks (PNG), keypoints (JSON)
- **Garment Assetization**
- In: 2–3 product photos (≥ 1024 px), fabric tag (string)
- Out: Garment mesh (OBJ), UV (PNG), texture (2048^2 PNG)
- **Draping**
- In: Body mesh + garment mesh + pose
- Out: Draped garment mesh, collision map (NPZ)
- **Finisher**
- In: Soft render (PNG), control maps (PNG), adapters (features)
- Out: Final photo-real PNG/JPEG

10.5 QA Thresholds & Retry

- **Garment fidelity (CLIP/LPIPS):** ≥ 0.28 / ≤ 0.32
- **Aesthetics:** ≥ 0.58
- **Identity (face score):** ≥ 0.75
- **Leak/overpaint checks:** garment vs skin IoU ≥ 0.9
- **Retry policy:** at most 1 retry; adjust seed, +5 steps, +0.05 garment adapter, −0.02 denoise. If still failing → **route to Kling**.

10.6 Resource Profiles (Single 24 GB GPU)

- Controls: 0.3–0.6 s total (parallel)
- VTON expert: 0.9–1.8 s
- Img2img + Refiner: 6–9 s @ 1344 px
- Post: 0.6–1.0 s

- **E2E:** 8–12 s per image (batch 2–4). Cache user/garment assets.

10.7 Failure Modes & Fallbacks

- **Face drift** → raise FaceID weight; reduce denoise; apply CodeFormer.
- **Logo/print smear** → enable Tile Control pass; ensure UV baked texture exists.
- **Hem/collar melt** → increase Edge/Seg weights; Poisson seam fix.
- **Depth/pose mismatch** → recompute controls; prefer YOLOv8-Pose.
- **Lighting mismatch** → relight LUT; re-composite with matting.

10.8 Ownership & Responsibilities

- **ML-Perception Team:** parsing, pose, depth, normals, identity modules.
- **Geometry Team:** SMPL-X fitting, garment assetization, draping.
- **GenAI Team:** finisher configs, adapters, refiner, upscaling, post.
- **Backend Team:** job queue, caching, API, CDN, observability.
- **Integrations:** Kling API, retailer SDKs, admin dashboards.
- **QA/Ops:** metric thresholds, retries, dataset logging, LoRA distillation cadence.

10.9 Config Management

- All hyperparams stored in **YAML** (`/configs/pipeline.yaml`) with env overrides.
- Version every model weight/artifact in **Model Registry** (Postgres table + S3 path), with SHA256 and semantic version.
- Canary deploy via feature flags; roll back on trigger metrics.

---

11. Tier-S Gap List → Implementation Plan → Verified Model Links

This section enumerates **what's missing**, **how to implement it (step-by-step)**, and **canonical URLs** for every external model / weight we will use. No stubs. Real VTON.

11.1 Perception & Geometry (must ship first)

A) Human Parsing / Segmentation - **Missing:** Production parser with skin/hair/garment classes, UV mask export. - **Implement:** 1. Add SCHP as default parser (LIP/CIHP heads).
2. Convert masks to UV-aligned masks after SMPL-X fit.
3. Expose `POST /v1/parse` for debugging visualization. - **URLs:**
• SCHP (GitHub, models incl. LIP/CIHP): cite turn0search0 turn0search8 turn0search16

B) Pose (2D keypoints) - **Missing:** Robust, GPU-fast pose detector. - **Implement:** 1. Integrate **Ultralytics YOLOv8-Pose** (torch, `model=yolov8x-pose.pt`).
2. Export 17/25 keypoints to JSON; cache per input. - **URLs:** Ultralytics Pose docs: cite turn0search1 turn0search17

C) Depth & Surface Normals - **Missing:** Monocular depth + normals for ControlNet-Depth/Normal and 2.5D carve. - **Implement:** 1. Depth via **ZoeDepth** (torch.hub), fallback MiDaS.

2. Normals via **NormalBae** preprocessor (controlnet-aux) or ControlNet-Normal. - **URLs:** ZoeDepth repo: cite turn0search2 • MiDaS: cite turn0search3 • NormalBae controlnet-aux: cite turn3search17 • ControlNet-Normal (SD1.5): cite turn3search1 turn3search5

D) Identity Lock (Face) - **Missing:** Dual identity adapters with thresholds + fallback restore. - **Implement:** 1. **InstantID** (feature→Adapter) + **IP-Adapter FaceID-Plus V2**.
2. If identity score < 0.75 → raise FaceID weight, lower denoise. - **URLs:** InstantID: cite turn0search6 turn0search14 • IP-Adapter FaceID threads/binaries: cite turn0search21 turn0search5 turn0search13

E) SMPL-X Fitting - **Missing:** Canonicalized user body mesh with UV texture. - **Implement:** 1. Fit **SMPL-X** with **PIXIE** (preferred) or **SMPLify-X**.
2. Bake UV texture from user image(s); save OBJ/NPZ + 2048^2 UV. - **URLs:** SMPL-X: cite turn1search5 • PIXIE: cite turn1search0 turn1search10 • SMPLify-X: cite turn0search22

F) Garment Assetization (no CAD) - **Missing:** Multi-photo → mesh + UV + fabric tag. - **Implement:** 1. Silhouette carving + **ZoeDepth** priors; HED for seam hints.
2. UV unwrap; LaMa UV inpainting; lightweight fabric classifier. - **URLs:** HED: cite turn0search4 turn0search20 • LaMa: cite turn2search3

G) Differentiable Draping - **Missing:** Coarse cloth sim + collision-aware refinement. - **Implement:** 1. Coarse sim (Taichi/ARCSim-lite or CUDA custom).
2. Refinement losses for contact/collar/penetration + mm thickness. - **URLs:** (internal implementation; no single canonical repo)

11.2 Photoreal Finisher (local, Kling-grade)

A) Renderer Prior - **Missing:** Soft render for lighting/shadows before diffusion. - **Implement:** 3D **Gaussian Splatting** raster to produce RGBA base.
- **URLs:** Official 3DGS: cite turn1search1 turn1search16

B) Diffusion Finisher (Img2Img) - **Missing:** SDXL/Flux + multi-control stack + adapters. - **Implement:** 1. **SDXL 1.0 base + Refiner** with ControlNets (Pose/Depth/Seg/Normal/Edge) and Adapters (Garment/Face).
2. Alternative: **FLUX.1** (dev/schnell) where license fits. - **URLs:** SDXL base + refiner: cite turn1search2 • SDXL usage (Diffusers): cite turn1search7 • FLUX.1 (dev/schnell): cite turn1search3 turn1search8 turn1search13

C) Control Models - **Missing:** Pose/Depth/Seg/Normal/Edge for guidance. - **Implement:** 1. **OpenPose** ControlNet (pose).
2. **Depth** ControlNet.
3. **Segmentation** Control (SD1.5 or SDXL community variants).
4. **Normal** (normalbae).
5. **Edge** (HED/Lineart). - **URLs:** OpenPose ControlNet: cite turn3search0 turn3search11 • Depth ControlNet: cite turn1search4 • Seg (community SDXL options): cite turn3search23 turn3search12 • Normal: cite turn3search1 • ControlNet docs: cite turn1search19

D) Post-processing - **Missing:** Upscale, matting, seam-safe blending, optional face restore. - **Implement:** Real-ESRGAN x4 → downsample; BGMv2 matting & Poisson blend; CodeFormer when identity score drops. - **URLs:** Real-ESRGAN: cite turn2search1 • CodeFormer: cite turn2search2 • BGMv2: cite turn2search4

11.3 Local VTON Experts (garment warping/alignment)

We will support **multiple experts** and auto-route via garment/pose heuristics. - **StableVITON (CVPR'24)** — baseline diffusion VTON.
URL: repo + project page: cite turn2search7 turn2search14 - **GP-VTON (CVPR'23)** — strong parsing + local flow.
URL: cite turn2search5 - **DCI-VTON (MM'23)** — diffusion + appearance flow.
URL: cite turn2search13 - **CatVTON (ICLR'25)** — efficient diffusion VTON, <8 GB VRAM 1024×768.
URL: repo + project: cite turn4search0 turn4search2

> **Note:** License each model; avoid non-commercial checkpoints in paid tiers.

11.4 Kling API Integration (premium backend)

- **Missing:** Production client + SLAs.
- **Implement:**
- `POST /providers/kling/jobs` → returns `external_job_id`.
- Poll + webhook support (`/webhooks/kling`) with HMAC signature.
- Dedupe key: SHA256(user_img, garment_pack, pose, finisher_cfg).
- Error mapping: `EXTERNAL_RATE_LIMIT | BAD_INPUT | RENDER_FAILED` → retry/route.
- **Env:** `KLING_BASE_URL`, `KLING_API_KEY`, `KLING_TIMEOUT=120s`.

11.5 Ops & Observability (must-have for Tier-S)

- **Missing:** Deep metrics, artifact registry endpoints, reproducibility.
- **Implement:**
- Prometheus: `vfr_job_latency_seconds{stage}`, `vfr_job_failures_total{reason}`, GPU VRAM/Util, QC gauges (clip, lpips, aesthetics).
- Artifact Ensure API: `POST /v1/artifacts/ensure` (url, sha256, dest).
- Snapshot job config to `/storage/config_snapshots/<job_id>.yaml`.

11.6 CI/CD & Sandboxes

- **Missing:** E2E demo path with real models.
- **Implement:**
- GitHub Actions: build CPU & GPU images, run smoke E2E on tiny sample.
- `make demo`: downloads weights (via artifacts ensure), runs **StableVITON → SDXL finisher → Real-ESRGAN**, outputs gallery.

11.7 Exact Weight IDs / Downloads (initial set)

- **SDXL Base/Refiner:** `stabilityai/stable-diffusion-xl-base-1.0`, `stabilityai/stable-diffusion-xl-refiner-1.0`. cite turn1search2

- **FLUX.1 (optional):** `black-forest-labs/FLUX.1-dev`, `FLUX.1-schnell` (license gates). cite turn1search3 turn1search8
- **ControlNets:** `lllyasviel/sd-controlnet-openpose`, `lllyasviel/sd-controlnet-depth`, `control_v11p_sd15_normalbae`, SDXL union model `xinsir/controlnet-union-sdxl-1.0` (community). cite turn3search0 turn1search4 turn3search1 turn3search18
- **Annotators (pip):** `controlnet-aux` (HED, normalbae, etc.). cite turn3search17
- **ZoeDepth:** `isl-org/ZoeDepth` (torch.hub names: ZoeD_N/K/NK). cite turn0search2
- **MiDaS (fallback):** `isl-org/MiDaS`. cite turn0search3
- **HED edges:** original / PyTorch reimpls. cite turn0search4 turn0search20
- **InstantID:** `instantX-research/InstantID` + project page. cite turn0search6 turn0search14
- **IP-Adapter FaceID:** references & model notes. cite turn0search21 turn0search5
- **SMPL-X:** model & docs; **PIXIE** for fitting. cite turn1search5 turn1search0
- **Gaussian Splatting:** official repo + project page. cite turn1search1 turn1search16
- **StableVITON / GP-VTON / DCI-VTON:** core VTON experts. cite turn2search7 turn2search5 turn2search13
- **LaMa / Real-ESRGAN / CodeFormer / BGMv2:** post stack. cite turn2search3 turn2search1 turn2search2 turn2search4

11.8 Acceptance Criteria (Tier-S)

- **Quality:** CLIP $\geq$ 0.28, LPIPS $\leq$ 0.32, Identity $\geq$ 0.75, Aesthetics $\geq$ 0.58; zero hem/collar melt in 20-image stress set.
- **Latency:** $\leq$ 12 s @ 1344px on 24 GB; batch 2–4.
- **Determinism:** Seed = hash(user, garment, pose); config snapshot saved; artifact SHA256 pinned.
- **Escalation:** Two failed retries → Kling job created automatically.

11.9 Work Breakdown (2-week sprint)

1. Perception pack (parse/pose/depth/normals) + tests.
2. SMPL-X + PIXIE canonicalization & UV bake.
3. Garment assetization (carve + HED + LaMa UV).
4. Draping (coarse + refine).
5. Finisher: SDXL + Control stack + adapters; post stack.
6. Expert router (StableVITON/GP-VTON/DCI-VTON/CatVTON).
7. Kling provider client + webhooks.
8. Metrics/Grafana + Artifact Ensure + `make demo`.

   Keep this section synchronized with `/configs/pipeline.yaml` and the **Model Registry** table above. All engineers must pin exact versions and record SHA256 in the artifacts table.

---

12. Tier-Ω (Exceed-Kling) Stack – Best-in-Class Blueprint

Goal: Surpass Kling-level still-image try-on realism and consistency via geometry-first + diffusion-last pipeline, mixture-of-experts VTON, and 2.5D rendering with minimal denoise finishing. All components are production-grade (no stubs).

12.1 Modules & Chosen Models (authoritative)

Stage	Best-in-class pick	Purpose	Notes
Person Parsing	**SCHP** (CIHP/LIP heads)	Person/garment/skin/hair masks	Export UV-aligned masks post SMPL-X fit
Pose	**YOLOv8-Pose**	Fast & robust 2D keypoints	Cache per input; supports 17/25 KP sets
Depth	**ZoeDepth**	High-quality monocular depth	Torch-hub ready; tiling for hi-res
Normals	**controlnet-aux normalbae**	Surface normal map	Feeds Finisher control
Edges/Seams	**HED/Lineart**	Collar/hem/print edges	Guides seam fidelity
Identity (face)	**InstantID + IP-Adapter FaceID-PlusV2**	Identity lock	Use both; thresholds enforced
Body Model	**SMPL-X** (fit via **PIXIE**)	Canonicalized body mesh + UV texture	Cache per user/session
Garment Recon	**Silhouette-carve + ZoeDepth priors + HED seams**	2–3 photo → coarse mesh + UV + fabric tag	No CAD needed
UV Inpaint	**LaMa (UV space)**	Fill occlusions in garment UV	Works on unwrapped texture
Fabric Classifier	**Light CNN**	{denim, knit, satin, leather, printed}	Select drape & BRDF presets
Cloth Sim (coarse)	**Taichi/ARCSim-lite**	Fast drape to pose	10–30 ms target
Cloth Refine (diff)	**Custom differentiable refinement**	Contact, collar lift, penetration fix	Adds mm thickness
Renderer Prior	**3D Gaussian Splatting**	Soft lighting/shadows RGBA	Fast 2.5D render prior
VTON Experts (MoE)	**CatVTON / StableVITON / GP-VTON / DCI-VTON**	Garment alignment & appearance prior	Heuristic router by garment type/pose
Finisher (img2img)	**SDXL 1.0 or FLUX-1** + ControlNets + IP-Adapters	Photoreal bake at low denoise	Pose/Depth/Seg/Normal/Edge controls
Refiner	**SDXL-Refiner**	Final micro-detail polish	Small denoise
Upscale	**Real-ESRGAN x4**	Sharpen then downsample	Tile mode if needed
Face Restore	**CodeFormer** (only if needed)	Subtle face fix	Weight 0.5–0.7

Stage	Best-in-class pick	Purpose	Notes
Matting/ Relight	**BGMv2 + LUT**	Edge-clean + light match	Poisson seam blend

12.2 Execution Order (deterministic)

1) Pre-ingest → EXIF strip, PII guard.
2) Perception pack → parsing, pose, depth, normals, edges.
3) SMPL-X + PIXIE fit → bake user UV texture & UV-align masks.
4) Garment assetization → mesh+UV+texture+fabric tag; UV inpaint.
5) Cloth drape (coarse) → differentiable refine → collision/thickness OK.
6) 2.5D prior → 3D Gaussian Splat render (RGBA).
7) VTON MoE (if required by garment/pose) → aligned base.
8) Finisher (SDXL/FLUX) with controls + adapters, **denoise 0.16–0.24**.
9) Refiner → Upscale → Relight & Matting → Seam blend → Output.
10) QA & Retry → cache; escalate to Kling only on double-fail.

12.3 Baseline Finisher Hyperparams (Tier-Ω)

```
DENOISE=0.20
STEPS=40
CFG=6.2
RES_LONG=1408
CTRL_POSE=0.55
CTRL_DEPTH=0.65
CTRL_NORMAL=0.45
CTRL_SEG=0.78
CTRL_EDGE=0.35
ADAPT_GARMENT=1.00
ADAPT_FACEID=0.85
REFINER_DENOISE=0.14
TILEPASS_DENOISE=0.12
TILEPASS_STEPS=16
```

12.4 VTON Expert Router (rules)

- **Dress/coat/skirt:** GP-VTON → StableVITON fallback.
- **T-shirt/hoodie/knit:** CatVTON/DCI-VTON → StableVITON fallback.
- If sleeve/collar complexity high → enable **Edge/Seg** higher weights.
- If prints/logos critical → add **Tile Control** pass.

12.5 QA Thresholds (strict)

- **Garment fidelity:** CLIP ≥ 0.30, LPIPS ≤ 0.30.
- **Identity:** Face score ≥ 0.80.

- **Aesthetics:** ≥ 0.60.
- **Leak checks:** garment vs skin IoU ≥ 0.92.
- **Retry once** with: +5 steps, +0.05 garment adapter, −0.02 denoise. Else → Kling.

12.6 Performance Targets (24–48 GB GPU)

- Controls: 0.3–0.6 s (parallel)
- Drape+refine: 0.1–0.3 s
- VTON expert: 0.9–1.8 s
- Finisher+Refiner: 6–9 s @ 1408 px
- Post: 0.6–1.0 s
  E2E: 8–12 s/image (batch 2–4). Cache user & garment assets.

12.7 Acceptance (Exceed-Kling)

- Pass 95% of stress set with strict QA thresholds and no hem/collar melt; superior texture compliance on prints/embroidery; deterministic seeds with full artifact/version pinning.

12.8 Engineering Notes

- Store **artifact SHA256**, **config snapshot**, and **seed** per job.
- License audit gates for community models; block NC weights in paid tiers.
- Periodic **self-distillation** from in-house photoshoots (not third-party outputs) to keep improving garment micro-detail adapters.