



Agent Instructions for GISTer Development

READ THIS FIRST - Mandatory for all AI agents



Quick Start (5-Minute Onboarding)

You Are Here:

- **Project:** GISTer - AI-powered reseller listing app
- **Project Path:** `/home/ubuntu/gist_list`
- **Stack:** Next.js 14, TypeScript, Prisma, PostgreSQL, OpenAI GPT-4
- **Deployed At:** <https://gistlist.abacusai.app>

Before You Start Coding:

1. ☒ Read this file (you're doing it!)
2. ☒ Read **DOCUMENTATION_STRATEGY.md** (understand the system)
3. ☒ Review **docs/CHANGELOG.md** (see recent changes)
4. ☒ Check **docs/FEATURES.md** (know what exists)
5. ☒ Scan **sessions/** folder (see what other agents did)



Pre-Flight Checklist (Every Session)

Before Writing Code:

```
cd /home/ubuntu/gist_list
git status           # Check for uncommitted changes
git pull origin main # Get latest code
```

- ☐ Read latest 2-3 entries in `docs/CHANGELOG.md`
- ☐ Check `sessions/` for recent work by other agents
- ☐ Review `docs/ROADMAP.md` to align with priorities

During Coding:

- ☐ Update living documents (`docs/*.md`) as you make changes
- ☐ Add inline comments for complex business logic
- ☐ Use conventional commits (`feat: , fix: , docs:`)
- ☐ Test your changes incrementally

Before Creating Checkpoint:

- ☐ Update `docs/CHANGELOG.md` (mandatory!)
- ☐ Update `docs/FEATURES.md` (if you added/changed features)
- ☐ Update `docs/API.md` (if you changed endpoints)
- ☐ Update `docs/DATABASE.md` (if you modified schema)

- [] Create session summary in `sessions/YYYY-MM-DD_description.md`
- [] Run tests: `test_nextjs_project`
- [] Create checkpoint: `build_and_save_nextjs_project_checkpoint`
- [] Push to GitHub: `git push origin main`

Project Structure

/gist_list/	
app/	# Next.js App Router
app/	# Routes and pages
api/	# API endpoints
camera/	# Camera capture page
listing/[id]/	# Listing detail page
listings/	# Listings overview
connections/	# Marketplace connections
...	
components/	# Reusable UI components
lib/	# Utilities, DB, types
prisma/	# Database schema
schema.prisma	# Prisma schema (source of truth)
docs/	# Legacy docs (being consolidated)
docs/	# **NEW** Living documents (single source of truth)
CHANGELOG.md	# **UPDATE THIS EVERY SESSION**
FEATURES.md	# Feature inventory
ARCHITECTURE.md	# System design
API.md	# API documentation
DATABASE.md	# Schema documentation
ROADMAP.md	# Future plans
sessions/	# **NEW** Session summaries (append- only)
2025-10-14_ui-ux-fixes.md	
extension/	# Chrome extension (GISTer)
AGENT_INSTRUCTIONS.md	# **THIS FILE**
DOCUMENTATION_STRATEGY.md	# Documentation philosophy

Tech Stack & Architecture

Frontend:

- **Framework:** Next.js 14 (App Router)
- **UI:** Tailwind CSS, Shadcn UI components
- **State:** React hooks, Zustand (minimal global state)
- **Forms:** React Hook Form + Zod validation

Backend:

- **Database:** PostgreSQL (via Prisma ORM)
- **Auth:** NextAuth.js with credential provider
- **Storage:** AWS S3 (via `lib/s3.ts`)
- **AI:** OpenAI GPT-4 Vision (considering migration to Gemini 2.5 Flash)

External Integrations:

- **Marketplaces:** eBay (OAuth), Etsy (OAuth), Reverb (API key)
- **Payments:** Stripe (checkout & subscriptions)
- **Cloud Storage:** AWS S3 for temporary image storage

Key Files to Know:

- `app/prisma/schema.prisma` - Database schema (35+ models)
- `app/lib/db.ts` - Prisma client singleton
- `app/lib/s3.ts` - S3 file operations
- `app/api/listings/[id]/analyze/route.ts` - AI analysis endpoint
- `app/components/ui/` - Shadcn UI components



Common Tasks

Adding a New Feature:

1. **Design:** Check `docs/ARCHITECTURE.md` for system design patterns
2. **Database:** Update `schema.prisma` if needed → run `cd app && yarn prisma generate`
3. **API:** Create endpoint in `app/api/` → document in `docs/API.md`
4. **UI:** Create components in `app/components/` → use Shadcn UI
5. **Test:** Run `test_nextjs_project` to verify
6. **Document:** Update `docs/FEATURES.md` and `docs/CHANGELOG.md`
7. **Checkpoint:** Create session summary and checkpoint

Modifying the Database Schema:

```
cd /home/ubuntu/gist_list/app
# Edit prisma/schema.prisma
yarn prisma generate           # Regenerate Prisma client
# No migration needed - production DB, be careful!
```

⚠ CRITICAL: GISTer uses a **production database**. Schema changes must be **backward compatible**. Never drop columns or tables without user consent.

Adding an API Endpoint:

1. Create `app/api/your-endpoint/route.ts`
2. Use `NextRequest` and `NextResponse`
3. Add authentication if needed (check `app/lib/auth.ts`)
4. Document in `docs/API.md`
5. Update `docs/CHANGELOG.md`

Working with AI (OpenAI):

- **API Key:** Stored in `.env` as `OPENAI_API_KEY`
- **Main Endpoint:** `app/api/listings/[id]/analyze/route.ts`
- **Prompt Engineering:** Follow existing patterns in analyze route
- **Cost Tracking:** Update `Listing.tokensUsed` and `User.totalApiCost`

Working with Marketplaces:

- **eBay:** OAuth 2.0 - see `app/api/marketplace/ebay/`
- **Etsy:** OAuth 2.0 - see `app/api/marketplace/etsy/`
- **Reverb:** API key - see `app/api/marketplace/reverb/`
- **Credentials:** Stored in `EbayCredential` , `EtsyCredential` , `ReverbCredential` models



Critical Rules (DON'T BREAK THESE)

✗ NEVER:

- ✗ Drop database tables or columns without explicit user permission
- ✗ Skip updating `docs/CHANGELOG.md` before checkpoint
- ✗ Push code without running tests
- ✗ Modify `.env` file completely (append only)
- ✗ Hard-code API keys or secrets
- ✗ Deploy without creating a checkpoint
- ✗ Overwrite another agent's session summary



✓ ALWAYS:

- ✓ Use conventional commits (`feat:` , `fix:` , `docs:` , `refactor:`)
- ✓ Update living documents in the **same commit** as code changes
- ✓ Test before checkpointing: `test_nextjs_project`
- ✓ Create session summary before ending work
- ✓ Read recent session summaries before starting
- ✓ Preserve user data at all costs (production DB)



Debugging & Testing

Running Tests:

```
// Use the test_nextjs_project tool
test_nextjs_project("/home/ubuntu/gist_list")
```

Common Issues:

1. **Hydration Errors:** Ensure server/client HTML matches (no `Math.random()` , `new Date()` in render)
2. **Shadcn Select Crashes:** Never use empty string `""` as `value` prop
3. **Image Loading:** Use Next.js `Image` with `fill` prop inside fixed aspect ratio container
4. **Prisma Client Not Found:** Run `cd app && yarn prisma generate`

Debugging in Browser:

- Dev server runs on `http://localhost:3000` (after `yarn dev`)
- Use browser dev tools, console, network tab
- Check `docs/ARCHITECTURE.md` for system flow

Data Models (Key Entities)

Core Models:

- **User:** Users, subscription tiers, preferences, costs
- **Listing:** Items to sell (35+ fields)
- **Photo:** Images with metadata (no raw data stored long-term)
- **AI Notification:** Smart chips/alerts for users
- **Market Research:** Cached market data from APIs
- **Search Index:** Buyer search with facet-based quality grading

Marketplace Models:

- **EbayCredential, EtsyCredential, ReverbCredential:** OAuth tokens
- **PlatformData:** Platform-specific listing data (JSON)
- **ScheduledPost:** Timed posting to marketplaces

Feature Models:

- **UserChip:** User's saved quick-add chips
- **PostingQueue:** Semi-automated posting queue
- **PushSubscription:** PWA push notifications

Full schema: See `app/prisma/schema.prisma` or `docs/DATABASE.md`

Current Priorities (October 2025)

In Progress:

1. **Telemetry:** Being handled by GPT (no conflicts with other work)
2. **UI/UX Refinements:** Dropdown sensitivity, button placement (just completed)
3. **Special Items Detection:** Premium gating for special/vintage items (just completed)

Next Up:

1. **Unified Inventory System** ("Shelf"):
 - Batch image upload
 - CSV/Excel import
 - Text writeup parsing
 - Manual entry
 - Gallery-like inventory page
2. **Extension Integration:**
 - Rebranding to GISTer
 - API endpoint integration
 - Scheduled posting UI
3. **AI Provider Migration:**
 - Consider Gemini 2.5 Flash for cost savings

See `docs/ROADMAP.md` for full roadmap.

Cost Considerations

GISTer operates on a **tight budget** with a **freemium model**:

- **Free Tier**: 4 premium AI analyses per user
- **Premium Tier**: Unlimited analyses + advanced features

Cost-Sensitive Areas:

- **OpenAI API**: Largest cost driver (considering Gemini migration)
- **S3 Storage**: Images deleted after AI analysis or item sold
- **Marketplace APIs**: eBay/Etsy rate limits

Track costs: Update `User.totalApiCost` and `Listing.apiCost`

Multi-Agent Coordination

Working with Other Agents:

1. **Check session summaries**: See what other agents did recently
2. **Pull before push**: Always `git pull` before starting work
3. **Communicate via docs**: Update CHANGELOG.md to signal changes
4. **Avoid conflicts**: If unsure, check recent session summaries

Handling Conflicts:

- **Docs win over code**: If docs say X but code does Y, fix the mismatch
 - **Newer session wins**: If conflict, check which agent worked most recently
 - **Ask for clarification**: If truly unclear, create a session summary explaining the issue
-

Getting Help

Resources:

1. **DOCUMENTATION_STRATEGY.md**: How docs work
2. **docs/ARCHITECTURE.md**: System design patterns
3. **docs/FEATURES.md**: What exists and where
4. **docs/API.md**: Endpoint reference
5. **sessions/**: See how other agents solved problems

Stuck?

1. Check session summaries in `/sessions/` for similar issues
 2. Review `docs/ARCHITECTURE.md` for design patterns
 3. Read code comments in similar features
 4. Create a session summary documenting the blocker
-

✓ Session End Checklist

Before finishing your work:

- [] All tests passing (`test_nextjs_project`)
 - [] `docs/CHANGELOG.md` updated
 - [] `docs/FEATURES.md` updated (if applicable)
 - [] `docs/API.md` updated (if applicable)
 - [] `docs/DATABASE.md` updated (if applicable)
 - [] Session summary created in `sessions/YYYY-MM-DD_description.md`
 - [] Checkpoint created with descriptive message
 - [] Code pushed to GitHub (`git push origin main`)
 - [] No uncommitted changes (`git status`)
-

🎓 Philosophy

Good documentation is code. Treat docs with the same rigor as code:

- Update docs **in the same commit** as code changes
- Write docs **for your future self** and other agents
- Document **why**, not just **what**
- Keep docs **accurate**, not just **complete**

You are not alone. Multiple agents and humans work on this project. Your documentation is your primary communication channel with them.

Welcome to GISTer development! 🚀

Last Updated: 2025-10-14

Version: 1.0

Questions: Create a session summary in `/sessions/` and other agents will respond