UY-JOY

Real Estate Visualization Platform

Technical Documentation & Development Report

Version 1.0 | February 19, 2026

Table of Contents

1. Executive Summary

Uy-Joy is a modern real estate visualization platform designed for property developers in Uzbekistan. The platform enables interactive floor plan exploration, apartment browsing, and lead generation through an intuitive user interface.

Key Achievements:

- Full-stack Next.js 14 application with TypeScript
- Multi-language support (Uzbek, English, Russian) with Uzbek as default
- Interactive floor plan editor with drag-and-drop unit placement
- Real-time apartment filtering and search
- Integrated contact system (Telegram, Phone, Web forms)
- Image optimization pipeline for fast loading
- Responsive design for all devices

2. Technology Stack

Category	Technology	Version
Frontend	Next.js	14.2.35
Frontend	React	18.x
Frontend	TypeScript	5.x
Frontend	Tailwind CSS	3.4.x
Backend	Next.js API Routes	14.x
Database	PostgreSQL (Neon)	Serverless
ORM	Prisma	5.22.x
i18n	next-intl	3.x
Image Storage	Cloudinary	2.9.x
Image Processing	Sharp	0.34.x
Authentication	NextAuth.js	4.x
Hosting	Vercel	Serverless

3. System Architecture

Application Structure:

The application follows a modern monolithic architecture using Next.js App Router:

```
src/
■■■ app/ # Next.js App Router pages
■ ■■■ api/ # REST API endpoints
■ ■■■ kvartiralar/ # Apartments listing page
■ ■■■ kvartiralarni-korish/ # Explore page
■ ■■■ portal/ # Admin management portal
■ ■■■ projects/ # Project detail pages
■■■ components/ # Reusable React components
■■■ lib/ # Utilities and configurations
■■■ messages/ # i18n translation files
```

Data Flow:

- User requests page → Next.js Server Component fetches data
- Prisma ORM queries PostgreSQL database
- Data returned and rendered with React Server Components
- Client components handle interactivity (filters, modals, forms)
- Form submissions → API routes → Database updates

4. Features Implemented

4.1 Public Features

- Homepage with animated statistics and featured apartments
- Apartment listing page with advanced filters (rooms, area, price, status)
- Interactive floor plan exploration (Building → Floor → Unit selection)
- Unit detail modal with sketch image, specifications, and pricing
- Contact form with lead capture (name + phone)
- Floating contact sidebar (Telegram message, Phone call)
- Multi-language support with automatic browser detection
- Responsive design for mobile, tablet, and desktop

4.2 Admin Features

- Secure admin portal with authentication
- Project management (create, edit, delete)
- Building management with floor configurations
- Interactive floor plan editor with unit placement
- Unit management (rooms, area, price, status, sketch upload)
- Reservation tracking (customer name, phone, notes)
- Image gallery management per building
- User management for admin accounts

5. Database Schema

The database uses PostgreSQL with Prisma ORM. Key models include:

Model	Key Fields	Relationships
Project	id, name, description, address	Has many Buildings
Building	id, name, floors, projectId	Belongs to Project, Has many Floors
Floor	id, number, planImage, buildingId	Belongs to Building, Has many Units
Unit	id, unitNumber, rooms, area, status, price	Belongs to Floor
Lead	id, name, phone, unitId, createdAt	Optional Unit reference
User	id, email, password, role	Authentication

Unit Status Values:

Status	Color	Description
available	Green (#4CAF50)	Ready for sale
reserved	Amber (#F9A825)	Customer interested, pending payment
sold	Red (#E53935)	Transaction completed

6. API Endpoints

Endpoint	Method	Description
/api/projects	GET, POST	List/Create projects
/api/projects/[id]	GET, PUT, DELETE	Project CRUD
/api/buildings	GET, POST	List/Create buildings
/api/buildings/[id]	GET, PUT, DELETE	Building CRUD
/api/floors	GET, POST	List/Create floors
/api/floors/[id]	GET, PUT, DELETE	Floor CRUD
/api/units	GET, POST	List/Create units
/api/units/[id]	GET, PUT, DELETE	Unit CRUD
/api/leads	POST	Submit contact form
/api/upload	POST	Image upload with optimization
/api/auth/[...nextauth]	GET, POST	Authentication

7. Design System

7.1 Color Palette

Name	Hex Code	Usage
Navy 900 (Primary)	#1E2A38	Headers, buttons, text
Gold 400 (Accent)	#C9A86A	CTAs, highlights, phone button
Background	#F7F8FA	Page backgrounds, cards
Available	#4CAF50	Available unit status
Reserved	#F9A825	Reserved unit status
Sold	#E53935	Sold unit status

7.2 Typography

Element	Font	Weight
Headings	Poppins	600-700 (Semibold/Bold)
Body Text	Inter	400-500 (Regular/Medium)

7.3 Components

- Border Radius: 10px (rounded-btn class)
- Shadow: 0 4px 6px rgba(0,0,0,0.1) (shadow-card)
- Buttons: Navy background, white text, gold for phone CTA
- Forms: Compact, minimal fields (name + phone only)
- Cards: White background, subtle shadow, hover lift effect

8. Security & Performance

8.1 Security Measures

- Protected admin routes with NextAuth.js authentication
- Hidden admin URL path (/portal/management-x7k9)
- CSRF protection via Next.js built-in mechanisms
- Input validation on all API endpoints
- Environment variables for sensitive configuration

8.2 Performance Optimizations

- Image optimization with Sharp (resize, compress on upload)
- Next.js Image component for automatic optimization
- Server Components for reduced JavaScript bundle
- Database query optimization with Prisma includes
- Static generation where possible

Image Handling (Cloudinary):

Feature	Description
Auto Format	Converts to WebP/AVIF based on browser
Auto Quality	Optimizes compression automatically
CDN Delivery	Global edge network for fast loading
Transformations	Resize, crop on-the-fly via URL params

9. Deployment Guide

9.1 Cloud Services Used

- Vercel - Serverless hosting platform (automatic deployments from GitHub)
- Neon - Serverless PostgreSQL database with connection pooling
- Cloudinary - Image storage and CDN with automatic optimization
- GitHub - Source code repository (wxusan/uy-joy)

9.2 Environment Variables

Variable	Description
DATABASE_URL	Neon PostgreSQL connection string with pgbouncer
NEXTAUTH_SECRET	Random secret for session encryption
NEXTAUTH_URL	Production URL (https://uy-joy-qmw3.vercel.app)
CLOUDINARY_CLOUD_NAME	Cloudinary cloud name
CLOUDINARY_API_KEY	Cloudinary API key
CLOUDINARY_API_SECRET	Cloudinary API secret

9.3 Deployment Process

1. Push code to GitHub repository
2. Vercel automatically detects changes and starts build
3. Build runs: prisma generate && next build
4. Vercel deploys to production URL
5. Database already configured in Neon (serverless, always on)
6. Images stored in Cloudinary CDN (global delivery)

10. Future Roadmap

10.1 Planned Features

- AI Chatbot integration for customer support
- Telegram Bot for notifications and inquiries
- Comparison tool (compare multiple apartments)
- Favorites/Wishlist functionality
- PDF export for apartment details
- Virtual tour / 3D visualization links
- Analytics dashboard for admins
- Bulk operations for unit management

10.2 Technical Improvements

- Implement Redis caching for frequently accessed data
- Add comprehensive test suite (Jest, Playwright)
- Set up CI/CD pipeline with GitHub Actions
- Implement rate limiting on API endpoints
- Add error tracking with Sentry

— End of Technical Report —