Crypto Checkout Widget

A responsive crypto payment checkout widget built with Next.js and TypeScript, designed to be embedded on any website for seamless crypto-to-fiat conversions.

🚀 Demo

Live Demo

📋 Features

- **Multi-tab Interface**: Crypto to cash, Cash to crypto, and Crypto to fiat loan options
- **Real-time Conversion**: Live exchange rate calculations with debounced input
- **Responsive Design**: Optimized for desktop, tablet, and mobile devices
- **Form Validation**: Comprehensive validation with user-friendly error messages
- **Loading States**: Visual feedback during conversion processes
- **Dropdown Animations**: Smooth open/close animations for all dropdowns
- **Toast Notifications**: Success and error notifications using react-toastify
- **Accessibility**: Proper ARIA labels and keyboard navigation support

🛠️ Setup Instructions

Prerequisites

- Node.js 18+
- npm or yarn

Installation

1. **Clone the repository**

```
git clone https://github.com/your-username/novacrust-widget.git
cd crypto-checkout-widget
```

2. **Install dependencies**

```
npm install
# or
yarn install
```

3. **Add font files** (if using local Clash Display font)

```
public/fonts/ClashDisplay-Variable.woff2
```

4. **Run development server**

```
npm run dev
# or
yarn dev
```

5. **Open in browser**

```
http://localhost:3000
```

Build for Production

```
npm run build
npm start
```

🏗 Project Structure

```
app/
├── components/
│   ├── AmountInput.tsx          # Currency input with integrated dropdown
│   ├── CryptoConverterWidget.tsx # Main widget component
│   ├── CryptoDropdown.tsx       # Crypto selection dropdown
│   ├── CurrencySelector.tsx     # Currency display button
│   ├── Dropdown.tsx             # Generic dropdown component
│   ├── WalletDropdown.tsx       # Wallet selection dropdown
│   ├── tab.tsx                  # Tab selector component
│   └── interfaces.ts            # TypeScript interfaces
├── fonts.ts                     # Font configurations
├── globals.css                  # Global styles
├── layout.tsx                   # Root layout with toast container
└── page.tsx                     # Main page component
```

🎨 Design Implementation

This project implements 2 key screens from the Figma design:

1. **Main Conversion Interface**: Multi-step form with currency selection, amount input, and wallet selection
2. **Interactive States**: Loading, error, and success states with proper user feedback

🔧 Technical Decisions & Trade-offs

Assumptions Made

- **Mock Exchange Rates**: Used hardcoded conversion rates (ETH-NGN: 6.5M, USDT-NGN: 1650) instead of real API integration
- **Simulated API Calls**: 2-second delay with 80% success rate for demonstration purposes
- **Font Fallback**: Used Space Grotesk as fallback for Clash Display font
- **Icon Assets**: Assumed icon files exist in `/icons/` directory

Trade-offs

1. **Component Integration vs Separation**
    - **Chosen**: Integrated dropdown logic into AmountInput component
    - **Trade-off**: Less reusable but better positioning control and simpler state management

2. **Animation Performance vs Complexity**
    - **Chosen**: CSS transitions over complex animation libraries
    - **Trade-off**: Simpler implementation but fewer animation options
3. **Form Validation Strategy**
    - **Chosen**: Client-side validation with toast notifications
    - **Trade-off**: Immediate feedback but no server-side validation
4. **State Management**
    - **Chosen**: Local component state with useEffect for conversions
    - **Trade-off**: Simpler setup but less scalable for complex applications

Technical Highlights

- **Responsive Design**: Mobile-first approach with Tailwind CSS breakpoints
- **TypeScript**: Full type safety with custom interfaces
- **Performance**: Debounced conversion calculations (300ms delay)
- **Accessibility**: Proper focus management and keyboard navigation
- **Error Handling**: Comprehensive error states with auto-dismiss timers

🚀 Deployment

Vercel (Recommended)

1. Push code to GitHub
2. Connect repository to Vercel
3. Deploy automatically

Manual Deployment

```
npm run build
npm run export  # if using static export
```

🔮 Future Enhancements

- Real exchange rate API integration
- Backend payment processing
- Multi-language support
- Advanced form validation
- Transaction history
- Webhook integration

📝 License

MIT License - feel free to use this project as a reference or starting point.