OTP Generator Documentation

Overview

The OTP Generator is a web application that allows users to generate and send One-Time Passwords (OTPs) via SMS. The application is built using Next.js and integrates with Twilio for SMS functionality.

Core Features

- Generate OTPs with customizable length and expiration time
- Send OTPs via SMS to contacts
- View message history and delivery status

Technology Stack

Frontend

- **Next.js 14**: React framework with App Router
- **TypeScript**: For type safety and better developer experience
- **Tailwind CSS**: For styling and responsive design
- **Framer Motion**: For smooth animations and transitions
- **React Hook Form**: For form handling and validation
- **Shadcn/ui**: For pre-built, accessible UI components

Backend

- **Next.js API Routes**: For server-side functionality
- **Twilio**: For SMS sending capabilities

Architecture

Key Components

1. **OTP Generation**

```
// Core OTP generation logic
const generateOTP = (length: number = 6): string => {
  const digits = '0123456789';
  let otp = '';
  for (let i = 0; i < length; i++) {
    otp += digits[Math.floor(Math.random() * 10)];
  }
  return otp;
};
```

1. **SMS Service**

```
// SMS service interface
interface SMSService {
  sendMessage(phoneNumber: string, message: string): Promise<MessageRe
cord>;
  getMessageHistory(phoneNumber?: string): Promise<MessageRecord[]>;
}
```

1. **Message History**

```
// Message record structure
interface MessageRecord {
  id: string;
  phoneNumber: string;
  message: string;
  status: 'sent' | 'delivered' | 'failed';
  timestamp: Date;
}
```

Design Decisions

1. **State Management**

- Used React's built-in state management (useState, useEffect)
- Kept state local to components where possible
- Used context for global state when needed

2. **Error Handling**
   - Implemented comprehensive error handling for SMS sending
   - Added user-friendly error messages
   - Included retry mechanisms for failed messages
3. **UI/UX Considerations**
   - Responsive design for all screen sizes
   - Loading states for async operations
   - Clear feedback for user actions
   - Accessible components using shadcn/ui
4. **Security**
   - OTPs are generated server-side
   - Phone number validation
   - Rate limiting for OTP generation
   - Message expiration handling

Best Practices Implemented

1. **Code Organization**
   - Feature-based folder structure
   - Separation of concerns
   - Reusable components
   - Type safety with TypeScript
2. **Performance**
   - Server-side rendering where appropriate

    - Optimized image loading
    - Efficient state updates
    - Proper cleanup in useEffect
3. **Testing**
    - Unit tests for OTP generation
    - Integration tests for SMS sending
    - Component testing with React Testing Library

Live Demo

The application is available at https://otp-gen.cyth.me

Getting Started

To run the application locally:

1. Clone the repository
2. Install dependencies: `npm install`
3. Set up environment variables (Twilio credentials)
4. Run development server: `npm run dev`

Future Improvements

1. Add more OTP generation algorithms
2. Implement message templates
3. Add analytics dashboard
4. Support for multiple languages
5. Enhanced security features by integrating authentication using NextAuth
6. Database integration with PostgreSQL and Redis caching
7. Contact Management feature (add, edit, delete)
8. Import and Export of contacts in CSV format with validation