Frontend (React + Vite + TailwindCSS)

🎯 Purpose

Modern, responsive chat interface for your AI model.

📋 Setup Instructions

1. Install Dependencies

```bash
cd frontend
npm install
```

2. Configure Environment Variables

```bash
# Copy example env file
cp .env.example .env

# Edit with your backend URL
nano .env
```

For development:

```env
VITE_API_URL=http://localhost:3000
```

For production:

```env
VITE_API_URL=https://your-backend.railway.app
```

3. Run Development Server

```bash
npm run dev
```

Visit `http://localhost:5173`

4. Build for Production

```bash
npm run build
```

The built files will be in the `dist` folder.

✨ Features

- **Real-time Chat**: Send messages and get instant responses
- **Markdown Support**: AI responses support markdown formatting
- **Conversation History**: Maintains context across messages
- **Settings Panel**: Adjust generation parameters (temperature, max length, top-p)
- **Responsive Design**: Works on desktop, tablet, and mobile
- **Error Handling**: Clear error messages and retry capability
- **Loading States**: Visual feedback during AI generation

🚀 Deployment

Deploy to Vercel (Recommended)

1. Push your code to GitHub
2. Go to vercel.com
3. Click "New Project"
4. Import your GitHub repository
5. Configure:
   - Framework Preset: Vite
   - Root Directory: `frontend`
   - Build Command: `npm run build`
   - Output Directory: `dist`
6. Add Environment Variable:
   - Name: `VITE_API_URL`
   - Value: Your backend URL (e.g., `https://your-backend.railway.app`)
7. Deploy!

Deploy to Netlify

1. Push code to GitHub
2. Go to netlify.com

3. Click "Add new site" → "Import an existing project"
4. Choose your repository
5. Configure:
    - Base directory: `frontend`
    - Build command: `npm run build`
    - Publish directory: `frontend/dist`
6. Add Environment Variable:
    - Key: `VITE_API_URL`
    - Value: Your backend URL
7. Deploy!

Deploy to GitHub Pages

1. Install gh-pages:

```bash
npm install --save-dev gh-pages
```

2. Update `vite.config.js`:

```javascript
export default defineConfig({
  plugins: [react()],
  base: '/your-repo-name/', // Add this
})
```

3. Add to `package.json`:

```json
"scripts": {
  "predeploy": "npm run build",
  "deploy": "gh-pages -d dist"
}
```

4. Deploy:

```bash
npm run deploy
```

🎨 Customization

Change Colors

Edit `src/App.jsx` to change the color scheme:

```javascript
// Change from indigo to your preferred color
className="bg-indigo-600" // Change to bg-blue-600, bg-purple-600, etc.
```

Change Layout

The layout is in `src/App.jsx`. Key sections:

- Header: Lines 110-135
- Settings: Lines 137-175
- Messages: Lines 182-235
- Input: Lines 237-265

Add Features

Add Copy Button

```javascript
import { Copy } from 'lucide-react';

// In message rendering:
<button
  onClick={() => navigator.clipboard.writeText(message.content)}
  className="text-gray-400 hover:text-gray-600"
>
  <Copy className="w-4 h-4" />
</button>
```

Add Dark Mode

```javascript
```

```
const [darkMode, setDarkMode] = useState(false);

// Add toggle button in header
<button onClick={() => setDarkMode(!darkMode)}>
  Toggle Dark Mode
</button>

// Update main div
<div className={darkMode ? 'dark' : ''}>
```

🔧 Troubleshooting

"Cannot connect to server"

- Check if backend is running
- Verify VITE_API_URL is correct
- Check browser console for CORS errors
- Ensure backend FRONTEND_URL includes your domain

Styles not loading

- Run `npm install` to ensure all dependencies are installed
- Check if Tailwind is configured in `tailwind.config.js`
- Restart dev server

Build fails

- Clear node_modules: `rm -rf node_modules && npm install`
- Check for TypeScript errors
- Verify all imports are correct

📱 Mobile Optimization

The app is already mobile-responsive, but you can further optimize:

```css
/* Add to index.css */
@media (max-width: 640px) {
  .message {
    max-width: 90%;
  }
}
```

🔒 Security Notes

- Never commit `.env` file
- Backend URL should use HTTPS in production
- Add Content Security Policy headers
- Validate all user inputs
- Implement rate limiting on frontend too

📊 Analytics (Optional)

Add Google Analytics:

```html
<!-- Add to index.html -->
<script async src="https://www.googletagmanager.com/gtag/js?id=GA_MEASUREMENT_ID"></script>
<script>
  window.dataLayer = window.dataLayer || [];
  function gtag(){dataLayer.push(arguments);}
  gtag('js', new Date());
  gtag('config', 'GA_MEASUREMENT_ID');
</script>
```

🎉 You're Done!

Your frontend is ready! After deployment:

1. Share your URL with users
2. Monitor usage and feedback
3. Iterate and improve