# Vercel Deployment Guide - Thought Ramble Parser

## Quick Deployment Overview

This guide will walk you through deploying your Thought Ramble Parser to Vercel in under 10 minutes. The app includes both a React frontend and Python FastAPI backend.

## Prerequisites

✅ **Vercel Account**: Sign up at [vercel.com](https://vercel.com) if you haven’t already  
✅ **GitHub Repository**: Push your code to GitHub (or GitLab/Bitbucket)  
✅ **LLM API Keys**: OpenAI or Anthropic API keys (optional but recommended)

## Step 1: Prepare Your Repository

### Upload to GitHub

1. Create a new repository on GitHub
2. Upload all project files:

* ├── backend/ # Python FastAPI backend  
  ├── thought-ramble-parser/ # React frontend  
  ├── vercel.json # ✅ Deployment configuration  
  ├── README.md  
  └── vercel-deploy.md # This guide

### Verify Files

Ensure these key files are present: - vercel.json (deployment configuration) - backend/app.py (Python backend) - backend/requirements.txt (Python dependencies) - thought-ramble-parser/package.json (Frontend dependencies)

## Step 2: Deploy to Vercel

### Connect Repository

1. **Go to Vercel Dashboard**: https://vercel.com/dashboard
2. **Click “Add New…”** → **“Project”**
3. **Import Git Repository**: Select your GitHub repository
4. **Click “Deploy”** (Vercel will auto-detect the configuration)

### Initial Deployment

* Vercel will automatically detect the vercel.json configuration
* The build process takes 2-3 minutes
* You’ll get a live URL like: https://your-project-name.vercel.app

## Step 3: Configure Environment Variables

### Access Environment Variables

1. **Go to Project Dashboard**: Click on your deployed project
2. **Navigate to Settings**: Click “Settings” tab
3. **Click “Environment Variables”** in the sidebar

### Required Variables

#### For OpenAI Integration (Recommended)

OPENAI\_API\_KEY=sk-your-openai-api-key-here

#### For Anthropic Integration (Alternative)

ANTHROPIC\_API\_KEY=your-anthropic-api-key-here

### Adding Environment Variables in Vercel

1. **Click “Add New”** in Environment Variables section
2. **Enter Variable Details**:
   * **Name**: OPENAI\_API\_KEY
   * **Value**: Your actual API key (starts with sk-)
   * **Environment**: Select “Production”, “Preview”, and “Development”
3. **Click “Save”**
4. **Repeat for other variables** if needed

### Environment Variable Interface

┌─────────────────────────────────────────────┐  
│ Environment Variables │  
├─────────────────────────────────────────────┤  
│ Name: OPENAI\_API\_KEY │  
│ Value: sk-•••••••••••••••••••••••••••••• │  
│ Environments: ☑ Production ☑ Preview ☑ Dev │  
│ [Save] │  
└─────────────────────────────────────────────┘

## Step 4: Redeploy with Environment Variables

After adding environment variables:

1. **Go to “Deployments” tab**
2. **Click “…” menu** on the latest deployment
3. **Select “Redeploy”** to apply environment variables
4. **Wait for deployment** to complete (1-2 minutes)

## Step 5: Test Your Deployment

### Verify Backend API

1. **Visit**: https://your-app.vercel.app/health
2. **Expected Response**:

* {  
   "status": "healthy",  
   "nlp\_ready": true,  
   "parser\_ready": true  
  }

### Test Frontend

1. **Visit**: https://your-app.vercel.app
2. **Click sample buttons**: Small, Medium, Large
3. **Click “Parse Thoughts”** to test the complete pipeline
4. **Verify results** appear in the right panel

## LLM Provider Configuration

### OpenAI Setup (Recommended)

1. **Get API Key**: https://platform.openai.com/api-keys
2. **Copy the key** (starts with sk-)
3. **Add to Vercel**: OPENAI\_API\_KEY=sk-your-key
4. **Models supported**: GPT-3.5-turbo, GPT-4, etc.

### Anthropic Setup (Alternative)

1. **Get API Key**: https://console.anthropic.com/
2. **Copy the key**
3. **Add to Vercel**: ANTHROPIC\_API\_KEY=your-key
4. **Models supported**: Claude family models

### No API Key (Basic Mode)

* The app works without API keys using rule-based parsing
* Limited functionality compared to LLM-enhanced parsing
* Good for testing and basic use cases

## Troubleshooting

### Common Issues

#### ❌ “API Error” or “503 Service Unavailable”

**Solution**: Check environment variables are correctly set

# Verify in Vercel dashboard:  
# Settings → Environment Variables  
# Ensure OPENAI\_API\_KEY is present and correct

#### ❌ Backend not responding

**Solution**: Check deployment logs 1. Go to “Functions” tab in Vercel 2. Click on backend/app.py function 3. Check logs for Python errors

#### ❌ Frontend loads but “Parse Thoughts” fails

**Solution**: API connectivity issue - Verify /health endpoint works - Check browser developer tools for CORS errors - Ensure vercel.json routing is correct

#### ❌ Build failures

**Solution**: Dependencies issue - Check backend/requirements.txt is complete - Verify thought-ramble-parser/package.json is valid - Review build logs in Vercel dashboard

## Advanced Configuration

### Custom Domain

1. **Go to Settings → Domains**
2. **Add your domain**
3. **Update DNS records** as instructed

### Performance Optimization

* Vercel automatically optimizes the React build
* Backend functions scale automatically
* No additional configuration needed

### Monitoring

1. **Analytics**: Built into Vercel dashboard
2. **Function logs**: Available in real-time
3. **Error tracking**: Automatic error reporting

## Project Structure Overview

Thought Ramble Parser/  
├── 🚀 FRONTEND (React + Vite + TypeScript)  
│ ├── thought-ramble-parser/  
│ │ ├── src/  
│ │ │ ├── components/ # UI components  
│ │ │ ├── lib/ # API & utilities  
│ │ │ └── App.tsx # Main app  
│ │ ├── public/data/ # Sample data  
│ │ └── package.json # Dependencies  
│  
├── 🐍 BACKEND (Python + FastAPI)  
│ ├── backend/  
│ │ ├── app.py # Main API  
│ │ ├── sample\_data.py # Test data  
│ │ └── requirements.txt # Dependencies  
│  
└── ⚙️ CONFIGURATION  
 ├── vercel.json # Vercel deployment config  
 ├── docker-compose.yml # Alternative deployment  
 └── README.md # Documentation

## Success Checklist

* ☐ Repository uploaded to GitHub
* ☐ Project imported to Vercel
* ☐ Initial deployment successful
* ☐ Environment variables configured
* ☐ Redeployed with API keys
* ☐ /health endpoint responding
* ☐ Frontend loading correctly
* ☐ Sample data loading
* ☐ Thought parsing working
* ☐ Results displaying with chunks

## Support & Next Steps

### Get Help

* **Vercel Docs**: https://vercel.com/docs
* **GitHub Issues**: Create issues in your repository
* **Discord**: Join Vercel’s community

### Scaling Up

1. **Custom Models**: Integrate additional LLM providers
2. **Database**: Add persistent storage for parsed thoughts
3. **Authentication**: Add user accounts and saved sessions
4. **API Rate Limiting**: Implement usage controls

Congratulations! Your Thought Ramble Parser is now live and ready to analyze stream-of-consciousness text! 🎉