# Deployment Guide: Optimized Edge Function

## 🚀 Performance Improvements Overview

The optimized edge function addresses the 9.48s latency bottleneck with these key improvements:

### Response Time Breakdown (Current vs Optimized):

* **Current**: 9.48s total (OpenAI: 6.64s | Glyph: 1.42s | DB: 0.95s | Network: 0.47s)
* **Target**: <2s total with aggressive caching and parallel processing

### Key Optimizations:

1. **Response Caching**: 1-minute TTL Map cache for repeated queries
2. **Quick Responses**: Instant replies for common emotions (grief, joy, anxiety, etc.)
3. **Parallel Processing**: Tag lookup and AI calls run simultaneously
4. **Faster Model**: gpt-4o-mini with max\_tokens limit
5. **Selective Glyph Processing**: Only for complex messages

## 📋 Deployment Steps

### Step 1: Access Supabase Dashboard

1. Go to [Supabase Dashboard](https://supabase.com/dashboard)
2. Select your Emotional OS project
3. Navigate to **Edge Functions** in the left sidebar

### Step 2: Locate Current Function

* Find the existing saori-fixed edge function
* Click on it to open the editor

### Step 3: Replace Function Code

1. **BACKUP CURRENT CODE**: Copy the existing code to a safe place
2. **Replace with optimized version**: Copy the entire content of optimized\_edge\_function.ts
3. **Verify Environment Variables**: Ensure these are still set:
   * SUPABASE\_URL
   * SUPABASE\_ANON\_KEY or PROJECT\_ANON\_KEY
   * SUPABASE\_SERVICE\_ROLE\_KEY or PROJECT\_SERVICE\_ROLE\_KEY
   * OPENAI\_API\_KEY

### Step 4: Deploy and Test

1. Click **Deploy** in the Supabase dashboard
2. Wait for deployment to complete (usually 30-60 seconds)
3. Test with a simple message to verify functionality

### Step 5: Performance Validation

Use the test script below to measure improvements:

import time

import requests

import json

from config import SUPABASE\_URL, SUPABASE\_ANON\_KEY

def test\_optimized\_performance():

url = f"{SUPABASE\_URL}/functions/v1/saori-fixed"

headers = {

"Authorization": f"Bearer {SUPABASE\_ANON\_KEY}",

"Content-Type": "application/json"

}

test\_messages = [

"I'm feeling overwhelmed by everything", # Should hit quick response

"Joy is bubbling up inside me today", # Should hit quick response

"I'm struggling with grief from my divorce", # Should hit quick response

"Complex philosophical thoughts about existence and meaning" # Should use full processing

]

results = []

for message in test\_messages:

start\_time = time.time()

response = requests.post(url, headers=headers, json={

"message": message,

"mode": "hybrid"

})

response\_time = time.time() - start\_time

results.append({

"message": message[:50] + "...",

"response\_time": f"{response\_time:.2f}s",

"success": response.status\_code == 200

})

print(f"✓ {message[:50]}... | {response\_time:.2f}s")

time.sleep(1) # Avoid rate limits

return results

# Run the test

print("Testing optimized edge function performance...")

test\_results = test\_optimized\_performance()

## 🎯 Expected Performance Gains

### Response Time Targets:

* **Common emotions** (grief, joy, anxiety): **<0.5s** (cached/quick responses)
* **Pattern matches**: **<1.0s** (local pattern + light AI)
* **Complex processing**: **<2.0s** (full AI + optimized glyph)
* **Cached responses**: **<0.2s** (direct cache hits)

### Cache Effectiveness:

* **1-minute TTL**: Handles immediate follow-ups and corrections
* **Emotion-based grouping**: Similar emotional content shares cache entries
* **Progressive enhancement**: Instant acknowledgment → cached/pattern → full AI

## 🔧 Monitoring and Debugging

### Check Performance in Supabase:

1. Go to **Edge Functions** → **saori-fixed** → **Logs**
2. Look for performance metrics in the console output
3. Monitor error rates and response times

### Debug Mode in Streamlit:

Add this to your Streamlit app for performance visibility:

if st.sidebar.checkbox("Debug Performance"):

st.sidebar.json({

"response\_source": result.get("source", "unknown"),

"processing\_time": result.get("processing\_time", "unknown"),

"cache\_stats": st.session\_state.get("performance\_stats", {})

})

## 🚨 Rollback Plan

If issues occur, quickly revert:

1. Go back to Supabase Edge Functions dashboard
2. Replace optimized code with backed-up original code
3. Deploy the original version
4. Investigate issues in development environment

## 📊 Success Metrics

Monitor these improvements:

* **Average Response Time**: Target <2s (down from 9.48s)
* **Cache Hit Rate**: Target >30% for repeat emotional patterns
* **User Experience**: Seamless conversational flow without long pauses
* **Error Rate**: Maintain <1% error rate with fallbacks

## Next Steps After Deployment

1. **Integrate Client Optimizations**: Update Streamlit UI to use instant acknowledgments
2. **Monitor Real Usage**: Track performance with actual user interactions
3. **Fine-tune Cache TTL**: Adjust based on usage patterns
4. **Add Streaming**: Implement progressive response loading for complex queries

**Ready to deploy?** The optimized edge function is backward-compatible with your existing system while providing dramatic performance improvements. The worst case is we revert to the original version if any issues arise.