DENTRA Testing Guide

🧪 Quick Local Testing

1. Health Check

```
curl http://localhost:3000/health
```

Expected Response:


```
{
  "status": "ok",
  "timestamp": "2026-01-10T21:25:00.000Z",
  "service": "DENTRA Backend",
  "version": "1.0.0"
}
```


2. View All Clinics

```
curl http://localhost:3000/clinics | jq
```

What to check:
- 5 clinics returned
- Each clinic has 5 services
- Appointment and call counts are included

3. View All Patients

```
curl http://localhost:3000/patients | jq
```

What to check:
- 20 patients returned
- Each patient has name, phone, email, DOB, insurance info
- Appointment counts are included

4. View All Calls

```
curl http://localhost:3000/calls | jq
```

Initial state: Should return empty array `[]` (no calls yet)

5. Test Twilio Webhook (Simulated Call)

Start a new call:

```
curl -X POST http://localhost:3000/webhook/voice \
  -H "Content-Type: application/x-www-form-urlencoded" \
  -d "To=%2B15551234567&From=%2B15559999999&CallSid=TEST_001"
```

Expected Response: TwiML XML with greeting message

Simulate user speech:

```
curl -X POST "http://localhost:3000/webhook/gather?callSid=TEST_001" \
  -H "Content-Type: application/x-www-form-urlencoded" \
  -d "SpeechResult=I%20need%20to%20schedule%20a%20dental%20cleaning&CallSid=TEST_001"
```

Expected Response: TwiML XML with AI response asking for patient details

Verify call was recorded:

```
curl http://localhost:3000/calls | jq '.[0]'
```

What to check:
- Call record exists with `call_sid: "TEST_001"`
- `status: "in_progress"`
- `clinic` object is populated
- `transcript` contains the conversation

6. Filter Calls by Clinic

```
curl "http://localhost:3000/calls?clinicId=<CLINIC_ID>" | jq
```

7. Get Specific Call Details

```
curl http://localhost:3000/calls/<CALL_ID> | jq
```

📝 API Documentation

Open in browser: http://localhost:3000/api-docs

What to test:
- All 9 endpoints are visible
- Can expand each endpoint to see parameters
- Can try out endpoints directly from Swagger UI
- Responses show example data

📡 Testing with Real Twilio

Prerequisites

- Service must be deployed to a public URL
- Twilio phone number configured

Steps:

1. **Deploy the service** (after testing locally)

2. **Configure Twilio webhooks:**
   - Go to Twilio Console (https://console.twilio.com/)
   - Navigate to your phone number
   - Set “A CALL COMES IN” to: `https://your-url/webhook/voice` (POST)
   - Set “STATUS CALLBACK URL” to: `https://your-url/webhook/status` (POST)

3. **Test the call flow:**
   - Call your Twilio phone number
   - Listen for: “Thank you for calling [Clinic Name]. This is Dentra, your AI assistant...”
   - Speak naturally: “I need to schedule an appointment”
   - AI should respond and ask for details
   - Continue the conversation

4. **Verify in database:**
   `bash`
   `curl https://your-url/calls | jq`

🔍 Testing Conversation Flow

Test Scenario 1: New Appointment

User says: “I want to book a dental cleaning”

Expected AI response:
- Asks for name, phone, date of birth
- Asks for preferred appointment date
- Confirms the information

Verify:

```
curl http://localhost:3000/calls | jq '.[0].intent'
# Should show: "new_appointment"
```

Test Scenario 2: Emergency

User says: “I have severe tooth pain and bleeding”

Expected AI response:
- Recognizes urgency
- Asks for immediate contact information
- Offers to prioritize the appointment

Verify:

```
curl http://localhost:3000/calls | jq '.[0].intent'
# Should show: "emergency"
```

Test Scenario 3: Reschedule

User says: “I need to change my appointment from tomorrow”

Expected AI response:
- Asks for patient identification (name/phone)
- Asks for new preferred date
- Confirms the change

Verify:

```
curl http://localhost:3000/calls | jq '.[0].intent'
# Should show: "reschedule"
```

Test Scenario 4: Inquiry

User says: “What are your office hours?”

Expected AI response:
- Provides clinic hours information
- Asks if they need anything else

Verify:

```
curl http://localhost:3000/calls | jq '.[0].intent'
# Should show: "inquiry"
```

⚠️ Troubleshooting

Issue: Service won’t start

Check:

```
tail -f /tmp/dentra-dev.log
```

Look for errors in database connection or missing environment variables

Issue: Webhook returns 500 error

Check:
1. Database is accessible
2. API keys are valid
3. Clinic phone number exists in database

Debug:

```
curl -v http://localhost:3000/webhook/voice \
  -H "Content-Type: application/x-www-form-urlencoded" \
  -d "To=%2B15551234567&From=%2B15559999999&CallSid=DEBUG"
```

Issue: AI not responding

Check:
1. OpenAI API key is valid
2. Check logs for OpenAI errors
3. Test OpenAI directly:

`bash`

```
  curl https://api.openai.com/v1/models \
    -H "Authorization: Bearer $OPENAI_API_KEY"
```

Issue: No call records in database

Check:

```
curl http://localhost:3000/calls
```

If empty, the webhook might not be receiving requests or there's a database issue.

📊 Performance Testing

Test Response Times

```
time curl http://localhost:3000/health
time curl http://localhost:3000/clinics
time curl http://localhost:3000/patients
```

Expected:
- Health: < 50ms
- Clinics: < 200ms
- Patients: < 200ms
- Webhook: < 500ms (includes AI processing)

Test Concurrent Requests

```
for i in {1..10}; do
  curl -X POST http://localhost:3000/webhook/voice \
    -H "Content-Type: application/x-www-form-urlencoded" \
    -d "To=%2B15551234567&From=%2B15559999999&CallSid=LOAD_TEST_$i" &
done
wait
```

Check that all 10 calls were created:

```
curl http://localhost:3000/calls | jq 'length'
# Should show: 10 (or more if previous tests)
```

✅ Test Checklist

- [ ] Health endpoint returns 200 OK
- [ ] All 5 clinics are returned with services
- [ ] All 20 patients are returned
- [ ] 50 appointments exist in database
- [ ] Webhook creates call record
- [ ] AI responds to user input
- [ ] Conversation transcript is saved
- [ ] Intent is identified correctly
- [ ] Call can be queried by ID

- [ ] Calls can be filtered by clinic
- [ ] Swagger documentation is accessible
- [ ] All endpoints return proper JSON
- [ ] CORS is working
- [ ] Error handling works (try invalid IDs)

🚀 Ready for Production?

Before deploying:

1. ✅ All tests pass
2. ✅ API documentation is complete
3. ✅ Database is seeded
4. ✅ Environment variables are set
5. ✅ Twilio webhooks are ready to configure
6. ✅ Logs are clean (no errors)
7. ✅ Response times are acceptable

Next Step: Deploy to production and configure Twilio phone number!