REPLIT AI — ONE■SHOT MULTI■TENANT PLATFORM CONVERSION PROMPT
Paste the following into Replit AI inside your FORKED platform project.

START PROMPT

You are Replit AI working on a forked project that will become a multi-tenant AI chatbot platform + demo hub for Treasure Coast AI.

This forked project is SAFE to modify. Perform full refactoring as long as the Faith House bot still works as the first real tenant.

You MUST follow the specification document named:
treasure_coast_ai_multi_tenant_full_spec.pdf
Implement everything described in that document.

----
1) INITIAL ANALYSIS (DO THIS FIRST)
----

1. Scan all files and identify:
- Framework (Node/Express OR Python/Flask/FastAPI)
- Main server entry file
- Current OpenAI integration
- Current routing
- Where prompts/UI are defined
2. Output a short architecture summary BEFORE making any changes.

----
2) PROJECT RESTRUCTURE → MULTI-TENANT READY

Restructure into:
/bots/
faith_house.json
restaurant_demo.json
barber_demo.json
homeservice_demo.json
autoservice_demo.json
gym_demo.json
/clients/
/logs/{clientId}/
/public/
/views/ or /templates/
server.js OR app.py
README.md

All bot logic moves into JSON files under /bots/.
Create helper:
getBotConfig(clientId, botId)
All chat routes MUST load configs through this helper.

----
3) DATA MODEL (JSON-BASED NOW, DB LATER)

Implement logical model:
Client:
id, name, type, bots[]
Bot:
clientId, botId, name, description,
systemPrompt, rules, businessProfile,
faqs[], emergencyGuidance

Design all code so that JSON can later be replaced by MySQL/Postgres.

----

4) ROUTING REQUIREMENTS

Create:

POST /api/chat/:clientId/:botId

- Validate IDs
- Load JSON config
- Build OpenAI request
- Log conversation
- Return reply + metadata

GET /demos

- Show all demo bots + Faith House

GET /demo/:botId

- Load demo UI
- Use clientId="demo" unless Faith House

GET /admin

- List clients + bots
- Buttons: Open Demo, View Config

----

5) LOGGING

Log each message to:

/logs/{clientId}/{botId}-YYYYMMDD.log

Each line JSON:

```
{
"timestamp": "...",
"clientId": "...",
"botId": "...",
"userMessage": "...",
"botReply": "..."
}
```

----

6) GLOBAL SAFETY RULES

All bots must:

- Never give medical or legal advice
- Never give crisis counseling
- If self-harm, suicide, overdose → return crisis-safe template:

"I can’t help in emergencies. Call 911 or local emergency number immediately."

- No dangerous instructions
- No sensitive data storage
- Follow OpenAI safety standards

Apply via systemPrompt injection + simple keyword checks.

----

7) FAITH HOUSE CONFIG (FIRST REAL TENANT)

Create /bots/faith_house.json using content from the PDF:

- clientId="faith_house"
- botId="faith_house_main"
- Supportive sober-living assistant
- Full businessProfile, rules, crisisHandling, faqs
- Strong “no medical/legal advice” boundaries

Route Faith House demo properly.

----

8) CREATE ALL 5 DEMO BOTS

Create:
restaurant_demo.json — "Coastal Breeze Grill"
barber_demo.json — "Fade Factory Barbershop"
homeservice_demo.json — "Treasure Coast Home Services"
autoservice_demo.json — "Coastline Auto Care"
gym_demo.json — "Treasure Coast Performance Gym"
Each JSON includes:

- businessProfile (name, phone, email, hours, services)
- systemPrompt (from PDF)
- rules (from PDF)
- faqs (from PDF)
- clientId="demo"
- botId="{bot name}"

All must appear in /demos and load via /demo/:botId.

----

9) README.md REQUIREMENTS

README must include:

- Project overview
- How to run
- All routes
- How to add new bot/client
- JSON schema
- Logging location
- Future DB migration notes

----

10) FINAL VERIFICATION (MANDATORY)

Before finishing:

1. Run project
2. Verify /demos works
3. Verify /demo/{botId} works for ALL bots
4. Verify /admin lists clients + bots
5. Verify JSON configs load dynamically
6. Modify a JSON value → confirm updates in responses
7. Verify logging works
8. Test crisis message → safe response required

Once all pass, the platform is complete.

END PROMPT
