# Chatbot Project Documentation

## 1. Project Overview

This project is a real‑time conversational chatbot built with \*\*Django\*\*, \*\*Django‑Channels\*\*, and \*\*LangChain\*\*.   
It exposes WebSocket endpoints that allow a client (e.g., a React or Flutter front‑end) to communicate with an AI agent powered by OpenAI models.   
The agent fetches live data from your PostgreSQL‑backed student‑information system through custom LangChain tools, enabling responses such as GPA look‑ups, attendance statistics, and aggregate counts (total students, courses, etc.).

## 2. High‑Level Architecture

• \*\*Daphne\*\* serves the ASGI application.   
• \*\*Channels\*\* routes WebSocket traffic to `ChatConsumer`.   
• \*\*LangChain AgentExecutor\*\* orchestrates the conversation, calling six custom tools:   
 `get\_student\_details`, `get\_student\_records`, `count\_total\_records`, `failed\_students`, `topper\_students\_list`, and `get\_student\_session`.   
• \*\*PostgreSQL\*\* stores all student, attendance, grade, course, internship, and performance data.

## 3. Installation & Setup

```bash  
# Clone repository  
git clone <repo-url>  
cd chatbot\_project  
  
# Create & activate virtual env  
python -m venv .venv  
source .venv/bin/activate  
  
# Install dependencies  
pip install -r requirements.txt  
  
# Environment variables  
cp .env.example .env # then edit with your own secrets  
  
# Run migrations & seed demo data  
python manage.py migrate  
python manage.py loaddata demo\_fixture.json  
  
# Launch server  
daphne -b 0.0.0.0 -p 5000 chatbot\_project.asgi:application  
```

## 4. WebSocket Endpoints

| Endpoint | Consumer | Purpose |  
|----------|----------|---------|  
| `/ws/chat/` | `ChatConsumer` | Main conversational interface |  
| (see settings) | | Additional lesson, course, and notification sockets available |

## 5. LangChain Tools

| Tool | Signature | Description |  
|------|-----------|-------------|  
| `get\_student\_details` | `name:str` | Return full `StudentSerializer` payload |  
| `get\_student\_records` | `name:str` | Shortcut to raw `Student` model for internal use |  
| `count\_total\_records` | `items:str` | Returns counts for students, courses, internships… |  
| `failed\_students` | \_no args\_ | List students whose performance status = \*Failed\* |  
| `topper\_students\_list` | \_no args\_ | Top‑10 students ordered by GPA |  
| `get\_student\_session` | `name:str, session:str` | Attendance / Grades / Internships / Performance |

## 6. Recent Fixes & Troubleshooting

### 6.1 Argument Mismatch in `topper\_students\_list`  
\*\*Symptom\*\*: `TypeError: topper\_students\_list() takes 0 positional arguments but 1 was given`   
\*\*Fix\*\*: Register the tool with `func=topper\_students\_list` (not a lambda) so LangChain does not pass the string argument placeholder.  
  
### 6.2 Missing Argument in `get\_student\_session`  
\*\*Symptom\*\*: `TypeError: get\_student\_session() missing 1 required positional argument: 'session'`   
\*\*Fix\*\*: Expose both parameters in the tool definition:   
```python  
student\_session\_tool = Tool(  
 name="get\_student\_session",  
 func=get\_student\_session\_async, # expects (student\_name, session)  
 coroutine=get\_student\_session\_async,  
 description="Fetch session data…"  
)  
```  
  
### 6.3 Common Runtime Checklist  
1. Confirm `OPENAI\_API\_KEY` is exported.   
2. Verify database credentials in `.env`.   
3. Use `python manage.py check` for model/serializer issues.   
4. Run `daphne` with \*\*http2 extra\*\* for production‑grade performance.

## 7. Testing

```bash  
# WebSocket smoke‑test  
websocat ws://127.0.0.1:5000/ws/chat/ -E  
> {"message": "How many students are there in total?"}  
< {"message": "{"total\_students": 123}"}  
```  
Unit tests live under \*chatbot/tests/\* and can be run with: `pytest`.

## 8. Deployment Notes

• \*\*Gunicorn + Uvicorn workers\*\* or \*\*Daphne\*\* behind Nginx terminate TLS.   
• Use `--http2` and `--root\_path` flags when reverse‑proxying.   
• Set `CHANNEL\_LAYERS` to Redis for multi‑instance scaling.

## 9. Credits

Maintained by Turbo DevOps Team • © 2025