Kalamna - API Design

BASE : api/v1/

1- AUTHENTICATION MANAGEMENT :

- POST -> auth/register/ → Register new organization + owner employee account, keep status="pending" until owner verifies email.
Access: Public

-GET → auth/verify/{token}/ → Verify the email to register org and activate account
Access: Public

- POST -> auth/login/ → Authenticate employee and receive access token
Access : Public

- POST -> auth/refresh/ → generate new access token using valid refresh token "custom permission"
Access: Authenticated

- POST -> auth/logout/ → logout user and blacklist refresh token
Access :Authenticated "owner,staff"

- GET -> auth/me/ → Get current authenticated employee details
Access: Authenticated "owner,staff"

2- EMPLOYEES / ADMINS MANAGEMENT :

- POST -> orgs/{org_uuid}/employee/invitations → Create invitation for new employee
Access: Owner Only

- GET -> auth/invitations/verify/{token} → Get invitation details (public - for verification page)
Access: Public

- POST -> auth/invitations/{token}/accept → Accept invitation and create employee account
Access: Public

- GET -> orgs/{org_uuid}/employee → List all employee in this organization
Access: Authenticated "Owner,Staff"

- GET -> orgs/{org_uuid}/employee/{employee_uuid} → Get specific employee details
Access: Authenticated "Owner,Staff"

- PATCH -> orgs/{org_uuid}/employee/{employee_uuid} → Update employee details (role change)
Access: Owner Only

- DELETE -> orgs/{org_uuid}/employee/{employee_uuid} → Remove employee from org
Access: Owner Only

3- PASSWORD MANAGEMENT :

- POST -> auth/password/forgot → Request password reset email
Access: Public

- GET -> auth/password/reset/verify/{token}/ → Verify password reset token validity
Access: Public

- POST -> auth/password/reset → Reset password using token
Access: Public

- POST -> auth/password/change → Change password for authenticated user
Access: Authenticated "Owner,Staff"

4- BUSINESS MANAGEMENT:

- GET -> orgs/{uuid} → get all org details "industry,name,...etc"
Access: Authenticated "owner,staff"

- PUT -> orgs/{uuid} → update org info
Access: Owner Only

- GET -> orgs/{uuid}/config → get all configures how the bot behaves
Access : Authenticated "Owner,Staff"

- PUT -> orgs/{uuid}/config → Update Bot Config to customize it based on org
Access : Owner only

- GET -> orgs/{uuid}/credentials → Show basic info about API key (not the raw key).
Access: Owner Only

- POST -> orgs/{uuid}/credentials → Invalidate old key & generate a new one.
Access: Owner Only

5-KNOWLEDGE-BASE MANAGEMENT:

- POST -> orgs/{org_uuid}/knowledge-bases/ → Create knowledge base entry "text or file"
Access: Authenticated , Owner/Staff.

- GET -> orgs/{org_uuid}/knowledge-bases/ → List all knowledge base entries
Access: Authenticated , Owner/Staff

- GET -> orgs/{org_uuid}/knowledge-bases/{kb_uuid}/ → Get specific knowledge entry
Access: Authenticated , Owner/Staff

- PUT -> orgs/{org_uuid}/knowledge-bases/{kb_uuid}/ → update specific knowledge entry
Access: Authenticated , Owner/Staff

- DELETE -> orgs/{org_uuid}/knowledge-bases/{kb_uuid}/ → Delete specific knowledge entry
Access: Owner Only

6- RAG PIPLINE:

Note : all this endpoints used internal in backend and ai not valid public

- POST -> internal/kb/chunk → run when a KB item is created or updated , split raw text into chunks then save into knowledge_chunks table

- POST -> internal/kb/embed → embedding chunks by generating embedding vectors store in pgvector column

- POST -> internal/kb/rebuild → fully re-chunk + re-embed , used when model changes or KB major update

- POST -> internal/ai/rag/search → Vector Search , Used during chat message processing , return top K similar KB chunks , raw, not merged or valid to be used

-POST -> internal/ai/rag/context → Context Builder (LLM-ready RAG output) ,

---

7- AI PIPLINE:

Note : all this endpoints used internal in backend and ai not valid public

- POST -> internal/ai/generate → AI Response Generation
[take user message , take context , take business config , call the model , return the AI reply]

8-CHAT-WIDGET ENDPOINTS:

Note : all this endpoints uses API Key authentication (X-API-Key header), not JWT

- POST -> chat/sessions/ → Start new chat session
Access: Public (requires API Key)

- POST -> chat/sessions/{session_id}/messages/ → end-users Send message to bot
Access: Public (requires session_token & API Key)

- GET -> chat/sessions/{session_id}/messages/ → Get chat history
Access: Public (requires session_token & API Key)

- POST -> chat/sessions/{session_id}/close/ → End chat session
Access: Public (requires session_token & API Key)

- POST -> chat/sessions/{session_id}/feedback/ → Submit feedback
Access: Public (requires session_token)

---

8. SUPER USER ENDPOINTS:

Note : all endpoints superuser can access but this ones only valid for superuser by access token

- POST -> auth/login/
- POST -> auth/logout/
- POST -> auth/refresh/
- GET -> orgs/
- GET -> orgs/{uuid}/
- POST -> orgs/ >> altrenative option of registeration
- PATCH -> orgs/{uuid}/ >> update on org like status
- GET -> analytics/ >> gets summary of Platform-wide analytics

Notes :

This is the MVP version of the platform's API.

- Additional features-such as per-org analytics, feedback reporting, voice message handling,..etc.

- The employee invitation system can also be expanded later to include invitation listing, status tracking, and the ability to cancel or resend invitations.

- AI Engineer can extend AI endpoints to more detailed ones but at all work is internal so no need to endpoints for it