

Kalamna - API Design

BASE : api/v1/

1- AUTHENTICATION MANAGEMENT :

- POST -> auth/register/
Access: Public
→ Register new organization + owner employee account, keep status="pending" until owner verifies email.
- GET -> auth/verify/{token}/
Access: Public
→ Verify the email to register org and activate account
- POST -> auth/login/
Access : Public
→ Authenticate employee and receive access token
- POST -> auth/refresh/
Access: Authenticated
→ generate new access token using valid refresh token "custom permission"
- POST -> auth/logout/
Access :Authenticated "owner,staff"
→ logout user and blacklist refresh token
- GET -> auth/me/
Access: Authenticated "owner,staff"
→ Get current authenticated employee details

2- EMPLOYEES / ADMINS MANAGEMENT :

- POST -> orgs/{org_uuid}/employee/invitations
Access: Owner Only
→ Create invitation for new employee
- GET -> auth/invitations/verify/{token}
Access: Public
→ Get invitation details (public - for verification page)
- POST -> auth/invitations/{token}/accept
Access: Public
→ Accept invitation and create employee account
- GET -> orgs/{org_uuid}/employee
Access: Authenticated "Owner,Staff"
→ List all employee in this organization
- GET -> orgs/{org_uuid}/employee/{employee_uuid}
Access: Authenticated "Owner,Staff"
→ Get specific employee details
- PATCH -> orgs/{org_uuid}/employee/{employee_uuid}
Access: Owner Only
→ Update employee details (role change)
- DELETE -> orgs/{org_uuid}/employee/{employee_uuid}
Access: Owner Only
→ Remove employee from org

3- PASSWORD MANAGEMENT :

- POST -> auth/password/forgot
Access: Public
→ Request password reset email
- GET -> auth/password/reset/verify/{token}/
Access: Public
→ Verify password reset token validity
- POST -> auth/password/reset
Access: Public
→ Reset password using token
- POST -> auth/password/change
Access: Authenticated "Owner,Staff"
→ Change password for authenticated user

4- BUSINESS MANAGEMENT:

- GET -> orgs/{uuid}
Access: Authenticated "owner,staff"
→ get all org details "industry,name,...etc"
- PUT -> orgs/{uuid}
Access: Owner Only
→ update org info
- GET -> orgs/{uuid}/config
Access : Authenticated "Owner,Staff"
→ get all configures how the bot behaves
- PUT -> orgs/{uuid}/config
Access : Owner only
→ Update Bot Config to customize it based on org
- GET -> orgs/{uuid}/credentials
Access: Owner Only
→ Show basic info about API key (not the raw key).
- POST -> orgs/{uuid}/credentials
Access: Owner Only
→ Invalidate old key & generate a new one.

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 registration
- 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