PROJECT REPORT

ON

FreeCMS.host

FOR

GHG Managing PVT LTD

BY

Vishwanath Mallappa Hatti

SAVITRIBAI PHULE PUNE UNIVERSITY

MASTERS IN COMPUTER APPLICATIONS

Maharashtra Education Society's
INSTITUTE OF MANGEMENT AND CAREER COURSES
(IMCC), PUNE -411038

MAHARASHTRA EDUCATION SOCIETY'S
(SINCE 1860)

INSTITUTE OF MANAGEMENT & CAREER COURSES (IMCC)
(AUTONOMOUS)

Approved by AICTE and Recognized by Savitribai Phule Pune University, Pune
IMCC Campus, 131, Mayur Colony, Kothrud, Pune 411038, Maharashtra, India | Ph.: 020-2546 6271 / 73 | e-mail: info.imcc@mespune.in | https://imcc.mespune.in

NAAC Accredited with Grade A+

Ref. No. MES IMCC /442/2025-26 **Date:06/05/2026**

CERTIFICATE

This is to certify that the Project Report entitled

"FreeCMS & GHG Manager"

is prepared by

Vishwanath Hatti

M.C.A. Semester IV Course for the Academic Year 2025-26 at M.E. Society's Institute of Management & Career Courses (IMCC), Pune - 411038. M.C.A Course is autonomous and affiliated to Savitribai Phule Pune University.

To the best of our knowledge, this is an original study done by the said student and important sources used by him/her have been duly acknowledged in this report.

The report is submitted in partial fulfillment of M.C.A Course for the Academic Year 2025-26 as per the rules and prescribed guidelines of Savitribai Phule Pune University.

SSParlikar

Dr Shilpa Parlikar
Project Coordinator

MSS

Ms. Manasi Shirurkar
Programme Coordinator

Dr. Ravikant Zirmite
Head, Dept of MCA MES IMCC

Dr. Santosh Deshpande
Director, MES IMCC

Rahul Dorbeswar
External Examiner

GHG MANAGING SOLUTIONS PRIVATE LIMITED

Email ID - admin@ghgmanager.com, Keshav.mitkar@gmail.com
Mobile No. 9890437631, 9373224210
CIN: U01500MH2023PTC412308

INTERNSHIP COMPLETION CERTIFICATE

This is to certify that **Mr. Vishwanath Hatti** has successfully completed his internship as a **Full Stack Developer Intern** at **GHG Managing Solutions Private Limited**.

His internship duration was from **6 February 2026** to **4 April 2026**.

During this period, he worked on real-world web applications, including the development and enhancement of the platform **freecms.host** and the company website **ghgmanager.com**. His responsibilities involved working on both frontend and backend technologies, contributing to feature development, system improvements, and overall application performance.

He demonstrated strong technical skills, problem-solving ability, and professionalism throughout the internship. His dedication and willingness to learn made him a valuable member of our team.

We wish him continued success in all his future endeavors.

Date: 4 April 2026



Authorized Signatory

Nageshwar Mitkar
Director
GHG Managing Solutions Private Limited

Add:- VANSH BLUE BELL SOCIETY, ROW HOUSE NO. D6, VAIBHAV COLONY, VASUNDHARA NAGAR, JALNA. JALNA.(MAHARASHTRA). 431203

Maharashtra Education Society's

Institute of Management and Career Courses (IMCC), Pune

AUTONOMOUS

[CGPA-3.32] in Cyc

CERTIFICATE

This is to certify that Vishwanath Mallappa Hatti has completed the project work entitled "***FreeCMS & GHG Manager***" under my guidance. The report is submitted in partial fulfillment of M.C.A. Course for the Academic Year 2025-2026 as per the rules & prescribed guidelines.

His/her work is found to be satisfactory and complete in all respects.

Date:09/05/2026

Dr. Ashwini Patil

(Internal Project Guide)

Acknowledgements

I would like to express my sincere gratitude to all those who supported and guided me throughout the development of the FreeCMS.host and GHGManager.com project.

I am highly thankful to GHG Managing Pvt Ltd. for providing me with the opportunity to work on this project and for offering valuable practical exposure to modern technologies, industry practices, and structured system development methodologies.

I would also like to express my heartfelt appreciation to my project guide and faculty members for their continuous guidance, valuable suggestions, and encouragement throughout the course of this project. Their support and mentorship played an important role in the successful completion of this work.

Lastly, I sincerely thank everyone who directly or indirectly contributed to the successful completion of this project.

Chapter	Section	Title	Page No
1		Introduction	
	1.1	Company Profile	
	1.2	Problem Statement	
	1.3	Objectives of Proposed System	
	1.4	Scope of Proposed System	
2		Analysis and Design	
	2.1	Entity Relationship Diagram	
	2.2	Business Use Case Diagram	
	2.3	Table Structure /Data Dictionary	
3		Testing	
	3.1	Test Cases with Results	
	3.2	Defect Report / Test Log	
4		Limitations of Proposed System	
5		Proposed Enhancements	
6		References	
7		User Manual	
8		Deployments	

Chapter 1: Introduction

FreeCMS is an AI-powered SaaS Headless Content Management System developed to simplify backend development and dynamic content management. The platform enables users to create structured data models, manage content, and generate APIs automatically without manually building backend infrastructure.

The system combines AI-driven content generation, dynamic schema management, and real-time analytics into a centralized platform. It follows a multi-tenant architecture where each user can manage multiple projects with isolated content structures, API keys, and security configurations.

Modern web application development often requires developers to spend significant time configuring databases, building administrative panels, and developing APIs. FreeCMS addresses these challenges by providing a visual and automated content management environment that reduces backend complexity and accelerates application development.

The platform also integrates an AI Content Filler powered by OpenAI, which generates structured content based on the project description and model schema. This feature helps users quickly populate applications with realistic and context-aware content while reducing manual data entry efforts.

To ensure secure content delivery, the system includes API key management and dynamic CORS domain validation, allowing only authorized applications to access project APIs. Additionally, the platform tracks API activity and usage metrics for monitoring and analytics purposes.

FreeCMS is built using modern technologies such as Next.js, Node.js, PostgreSQL, and MongoDB, providing scalability, performance, and flexibility. The project aims to deliver an intelligent and efficient headless CMS platform that combines modern backend automation with AI-assisted content management.

1.1 Company Profile

GHG Manager is a technology and sustainability-focused company dedicated to helping industries manage greenhouse gas emissions and transition toward cleaner and more sustainable operations. The company specializes in providing data-driven advisory services, carbon management solutions, and renewable energy investment support for organizations seeking environmentally responsible growth. (GHG Manager)

The organization focuses on developing structured and intelligent digital solutions that simplify sustainability management and environmental reporting processes. By combining technology, analytics, and sustainability expertise, GHG Manager enables businesses to monitor emissions,

improve operational efficiency, and align with evolving environmental standards and compliance frameworks.

GHG Manager follows a solution-oriented approach where data, automation, and systemdriven processes are used to support informed decision-making. Its services are designed to help organizations identify emission patterns, optimize resource usage, and adopt sustainable operational practices through scalable digital platforms.

One of the major objectives of GHG Manager is to make carbon management and sustainability reporting more accessible and manageable for industries and enterprises. The company emphasizes innovation, operational clarity, and practical implementation strategies that assist organizations in reducing environmental impact while maintaining business efficiency.

The company's focus on technology-enabled sustainability solutions reflects its commitment to addressing modern environmental challenges through intelligent system design and digital transformation. By integrating structured workflows, analytics, and advisory capabilities, GHG Manager aims to support organizations in building sustainable and future-ready operational ecosystems.

1.2 Problem Statement

Developing and managing backend systems for modern content-driven applications is often a complex, time-consuming, and resource-intensive task for developers and organizations. Developers frequently spend significant effort writing repetitive backend logic, configuring databases, managing APIs, and building administrative dashboards for every new project. Ensuring scalability, security, and proper API access control further increases the complexity of backend development, especially in multi-project or multi-tenant environments.

Content administrators also face difficulties in maintaining structured content across different models and manually populating large amounts of data during development and testing phases. Traditional CMS platforms often provide limited flexibility, require extensive manual configuration, and lack intelligent automation features that can simplify content generation and management processes. In addition, many existing systems do not provide integrated analytics or centralized monitoring for API usage and performance tracking.

To address these challenges, **FreeCMS** is designed as an AI-powered Headless Content Management System that automates and simplifies the entire content management workflow. The platform provides dynamic schema creation, automatic API generation, secure multi-project management, and an intelligent AI Content Filler that generates context-aware content using artificial intelligence. By reducing backend complexity and minimizing manual content

management efforts, FreeCMS improves development efficiency, scalability, and overall user productivity.

1.3 Objectives of Proposed System

The main objectives of the **FreeCMS** platform are:

- To simplify backend development through automated API generation and dynamic database management.
- To provide an AI-powered Content Filler that generates structured content based on project context and model schema.
- To enable users to create and manage custom content models using a visual Data Model Builder.
- To support secure multi-project management through a scalable multi-tenant architecture.
- To provide secure API access using API key management and dynamic CORS validation.
- To reduce manual content management efforts through intelligent automation and user-friendly administrative tools.
- To provide real-time API usage tracking and analytics for monitoring platform performance.
- To support efficient media handling and optimized asset delivery through integrated cloud-based storage services.
- To deliver a scalable and flexible headless CMS solution that improves development efficiency and content management workflows.

1.4 Scope of the Proposed System

The scope of the **FreeCMS** platform includes the following functionalities:

- **Secure Authentication System**: Implementation of user registration and login functionality using JWT authentication and encrypted password storage.
- **AI-Powered Content Generation**: Integration of OpenAI to generate structured and context-aware content automatically based on project descriptions and model schemas.
- **Visual Data Modeling**: A dynamic schema builder that allows users to create custom content models without manual database configuration.
- **Multi-Tenant Project Management**: Support for managing multiple independent projects with isolated content structures and configurations.
- **Content Management System**: Full CRUD operations for creating, updating, publishing, and managing dynamic content entries.

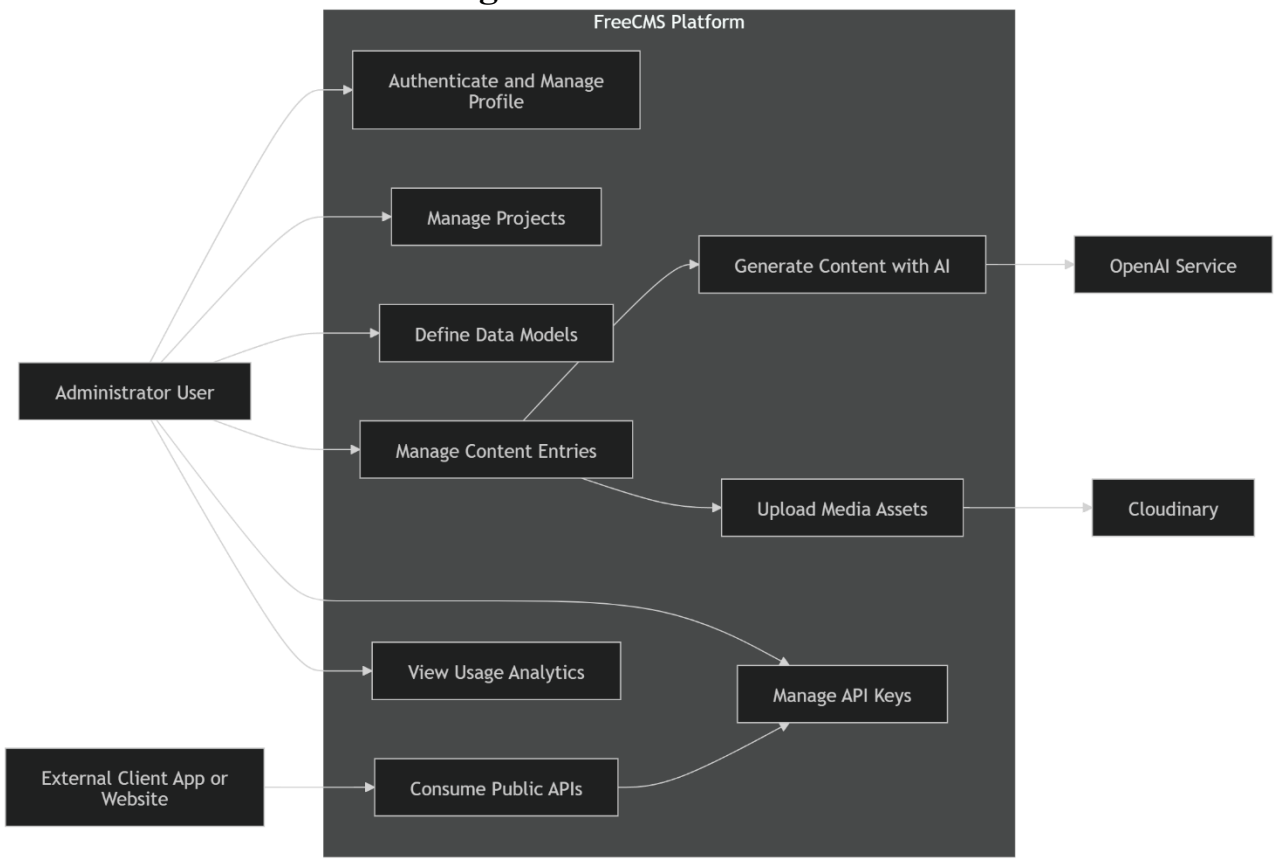
- **Automated API Generation**: Automatic creation of RESTful API endpoints for every defined content model.
- **Media Asset Management**: Integrated media upload and delivery system using cloud-based storage and CDN optimization.
- **API Security & Access Control**: Secure API key management and dynamic CORS validation for authorized client applications.
- **Analytics & Monitoring**: Real-time tracking of API usage, request status, and performance metrics.
- **Responsive Administrative Dashboard**: A modern and responsive web interface accessible across desktop and mobile devices.

Chapter 2: Analysis and Design

3.1 Entity Relationship Diagram (ERD)



3.2 Business Use Case Diagram



3.3 Table Structure/Data Dictionary

1. PostgreSQL Tables (Core CMS)

Table: User

Column	Type	Constraints	Description
id	UUID	PK, Default: uuid()	Unique identifier for the user.
name	String	Required	Full name of the user.
email	String	Required, Unique	Email address used for login.
password	String	Required	Hashed password (bcrypt).
preferences	JSON	Default: {}	User-specific UI settings (e.g., dark mode).
createdAt	DateTime	Default: now()	Timestamp of account creation.

Table: Project

The main tenant container for models and content.

Column	Type	Constraints	Description
id	UUID	PK, Default: uuid()	Unique identifier for the project.
name	String	Required	Display name of the project.
slug	String	Required, Unique	URL-friendly name used in API endpoints.
Column	Type	Constraints	Description
description	String	Optional (Default: "")	Used for AI context to guide content generation.
userId	UUID	FK (User.id)	The owner of the project.
allowedDomains	String[]	Default: []	CORS whitelist for public API access.
createdAt	DateTime	Default: now()	Timestamp of project creation.

Table: Model

Defines the structure of content types (e.g., "Blog Post", "Product").

Column	Type	Constraints	Description
id	UUID	PK, Default: uuid()	Unique identifier for the model.
name	String	Required	Display name of the content type.
slug	String	Required	Used in API routes (e.g., /api/v1/blog-post).
schema	JSON	Required	The Zod-based field definitions and types.
isSingle	Boolean	Default: false	True if the model represents a single page (singleton).
projectId	UUID	FK (Project.id)	The parent project.

Table: Content

Stores the actual data entries for models.

Column	Type	Constraints	Description
id	UUID	PK, Default: uuid()	Unique identifier for the entry.
slug	String	Optional, Unique	The URL path for this specific entry.
data	JSON	Required	The actual content fields and values.
status	String	Default: "draft"	Workflow state (draft/published).
projectId	UUID	FK (Project.id)	Tenant identifier.
modelId	UUID	FK (Model.id)	The schema this entry follows.
updatedAt	DateTime	Auto-update	Timestamp of last modification.

Table: ApiKey

Access tokens for external applications.

Column	Type	Constraints	Description
id	UUID	PK	Unique identifier.
key	String	Required, Unique	The secret key passed in headers.
name	String	Required	Label for the key (e.g., "Production Web").
projectId	UUID	FK (Project.id)	Project this key has access to.

---

2. MongoDB Collections (AI & Logs)

Collection: PromptLog

Logs every interaction with the AI Content Filler.

Field	Type	Description
_id	ObjectId	MongoDB unique identifier.
projectId	String	Logical reference to the PostgreSQL Project.
modelId	String	Logical reference to the PostgreSQL Model.
userPrompt	String	The text provided by the user (e.g., "Write a recipe").
systemInstruction	String	The personality/rules sent to OpenAI.
generatedContent	Object (Mixed)	The raw JSON object returned by the AI. tokensUsed
	Number	Total tokens (prompt + response) consumed. modelName
String	The OpenAI model used (e.g., gpt-4o-mini).	
createdAt	Date	Timestamp of generation.

---

3. Supporting Tables

Table: AnalyticsLog

High-volume log for API usage tracking.

Column	Type	Description
id	UUID	Unique log ID.
projectId	UUID	Project that received the request.
method	String	HTTP Method (GET, POST, etc.).

route **String** **The exact endpoint hit.**

status **Integer** **HTTP Status code returned (200, 404, etc.).**

duration **Integer** **Response time in milliseconds.**
timestamp DateTime When the request occurred.

Student Name	Roll No.	Date
Vishwanath Mallappa Hatti	2401066	09/05/2026

---

3. Testing & Quality Assurance

3.1 Test Case / Test Script

Test ID	Module	Test Case	Steps	Expected Result	Actual Result	Status
TC-01	IAM	Valid Admin Login	1. Enter email/pass 2. Submit	Secure JWT session established	Redirected to dashboard	PASS
TC-02	IAM	Duplicate Registration	1. Use existing email 2. Submit	Error: "Email already in use"	Validation triggered	PASS
TC-03	IAM	Password Validation	1. Use < 6 chars 2. Submit	Front-end validation error	UI blocked submission	PASS
TC-04	Core	Create Project	1. Name: "Test Project" 2. Save	Isolated workspace created	Metadata persisted	PASS
TC-05	Core	Slug Uniqueness	1. Create duplicate slug 2. Save	Unique constraint error	Backend 400 response	PASS

TC-06	Build	Define Text Field	1. Add "Title" field 2. Commit	Model schema updated (JSON)	Schema valid	PASS
TC-07	Build	Singleton Constraint	1. Set model to Singleton 2. Add 2nd entry	Entry button disabled	UI enforced singleton	PASS
TC-08	Ops	Content Creation	1. Fill fields 2. Save & Publish	Entry visible in Live grid	Status: PUBLISHED	PASS
TC-09	Ops	Draft Persistence	1. Save entry without publish	Status remains DRAFT	Entry saved locally	PASS
TC-10	Assets	Cloudinary Upload	1. Upload 5MB image 2. Wait for callback	Secure URL returned from CDN	Image rendered in grid	PASS
TC-11	Secure	API Key Gen	1. Click 'Generate' 2. Verify key	32-char hashed key visible	Key generated	PASS
TC-12	Secure	CORS Domain Check	1. Request from rogue domain 2. Check headers	403 Forbidden / CORS Block	Response blocked	PASS
TC-13	Deliver	Public API Hit	1. GET request with xapi-key	Valid JSON payload returned	HTTP 200 OK	PASS
TC-14	Deliver	Invalid Model Slug	1. Request non-existent slug	404 Model Not Found	Correct error code	PASS

TC-15	Insight	Analytics Logging	1. Hit API 5 times 2. View Charts	Volume counter increments by 5	Graph updated	PASS
TC-16	Build	Rich Text Validation	1. Save HTML in RichText field	Content sanitized and saved	Valid HTML stored	PASS

3.2 Defect Report / Test Log

Defect ID	Module	Description	Severity	Status	Resolution
D-01	IAM	Login spinner missing on slow connections	Low	Closed	Added Lucide-react Loader
D-02	Core	Slug collision during highconcurrency	Medium	Closed	Added unique DB constraint
D-03	Assets	Image upload timeout for 10MB+ files	High	Closed	Increased Multer buffer limits
D-04	Ops	Grid pagination logic skipping items	High	Closed	Refactored SQL offset logic
D-05	Secure	API Key visible in client console	Critical	Closed	Masked keys in nonsecret views
D-06	Insight	Latency chart shows 0ms for local hits	Low	Closed	Improved micro-timing accuracy
D-07	Build	Schema update wipes existing content	Critical	Closed	Added schema-delta migration
D-08	Delivery	CORS header inconsistent on 404s	Medium	Closed	Unified error middleware CORS
D-09	UI	Mobile layout break on analytics charts	Medium	Closed	Added responsive CSS breakpoints

D-10	SDK	Initial fetch fails on weak network	Low	Closed	Added retry-logic with backoff

---

4. Limitations of the Proposed System

While **FreeCMS** provides an efficient and automated content management solution, the system has certain limitations in its current implementation:

- **Internet Dependency**: The platform requires a stable internet connection for content management, AI generation, and media uploads.
- **AI Content Accuracy**: AI-generated content may occasionally require manual verification to ensure factual correctness and contextual accuracy.
- **OpenAI Service Dependency**: AI-related features depend on external OpenAI APIs and may be affected by service availability, rate limits, or pricing changes.
- **Large Media File Handling**: The system primarily focuses on image optimization and may have limited support for processing very large media files or advanced video transcoding.
- **No Offline Support**: The administrative dashboard currently does not support offline editing or synchronization.
- **Serverless Cold Start Delays**: Certain serverless hosting environments may introduce slight latency during the first request after inactivity.
- **Complex Relational Query Support**: Deep relational filtering and highly complex database queries may be limited compared to traditional relational database systems.
- **Limited Native Notification Services**: The platform does not currently include built-in email or workflow notification services and may require third-party integrations.

---

5. Proposed Enhancements

To further improve and expand the capabilities of **FreeCMS**, the following enhancements are proposed for future development:

- **Integrated Site Builder**: Develop a visual drag-and-drop page builder for creating complete websites directly within the platform.
- **Webhook Integration**: Implement webhook support to notify external applications whenever content is created, updated, or published.
- **GraphQL API Support**: Add GraphQL functionality alongside REST APIs to support flexible and optimized data querying.
- **Role-Based Access Control (RBAC)**: Introduce advanced permission management with customizable user roles and access levels.
- **Multi-Language Support (i18n)**: Enable localization features for managing content in multiple languages and regional formats.
- **Content Versioning & History**: Implement a version control system to track changes and restore previous content states.
- **Scheduled Publishing**: Allow users to schedule automatic publishing of content at specific dates and times.
- **Real-Time Collaborative Editing**: Support simultaneous content editing by multiple users using real-time synchronization technologies.
- **Custom Field Plugins**: Provide an extensible plugin system for creating and integrating custom field types and components.

---

6. References

The development of **FreeCMS** was guided by the following technical documentation, industry standards, and architectural references.

6.1 Core Technologies & Frameworks

- Next.js Documentation

  Used for App Router architecture, Server Components, and frontend optimization.
- Prisma ORM Documentation

  Used for database modeling, migrations, and type-safe queries.
- OpenAI API Documentation

  Used for GPT-4o-mini integration and AI-powered content generation.
- Express.js Documentation

  Used for RESTful API routing and middleware management.
- Tailwind CSS Documentation

  Used for responsive UI styling and utility-first design.
- Mongoose Documentation

  Used for MongoDB schema management and AI prompt logging.

6.2 Infrastructure & Cloud Services

- Neon Serverless Postgres Documentation

  Used for scalable serverless PostgreSQL infrastructure.
- Cloudinary Documentation

  Used for media uploads, optimization, and CDN delivery.
- MongoDB Atlas Documentation

  Used for managed cloud document database services.

6.3 CMS Standards & Architectural References

- Strapi Documentation

  Reference for headless CMS architecture and dynamic content modeling.
- Contentful Developer Documentation

  Reference for SaaS-based content delivery systems.

- Sanity Documentation

  Reference for structured content management and real-time editing.
- Directus Documentation

  Reference for database-driven API generation.
- Jamstack Architecture

  Reference for decoupled frontend-backend architecture principles.
- HeadlessCMS.org

  Used for feature comparisons and headless CMS industry standards.

6.4 Security & Validation Standards

- JWT Introduction

  Reference for stateless authentication mechanisms.
- MDN CORS Documentation

  Used for implementing secure cross-origin resource sharing.
- Zod Documentation

  Used for runtime schema validation and type safety.

---

7. Comprehensive User Manual: Workflow Guide

Screen 1: Public Landing & Portfolio Overview

Role in Workflow

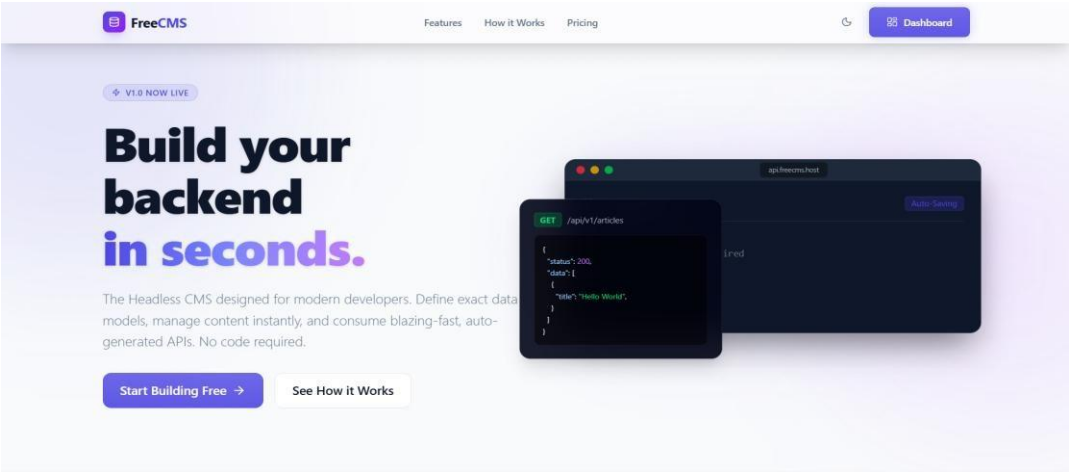
The initial discovery point for users where the platform's core features and value proposition are introduced.

Description

A modern landing page designed with interactive animations, premium gradients, and glassmorphic UI elements to showcase the capabilities of FreeCMS.

Workflow

1. Open your browser and navigate to the official platform domain.
2. Explore the platform overview and feature highlights.
3. Click the "Sign Up" or "Get Started" button to begin onboarding.



Screen 2: Secure Identity Gateway (Authentication)

Role in Workflow

Acts as the secure access point for administrators to authenticate and access their project environments.

Description

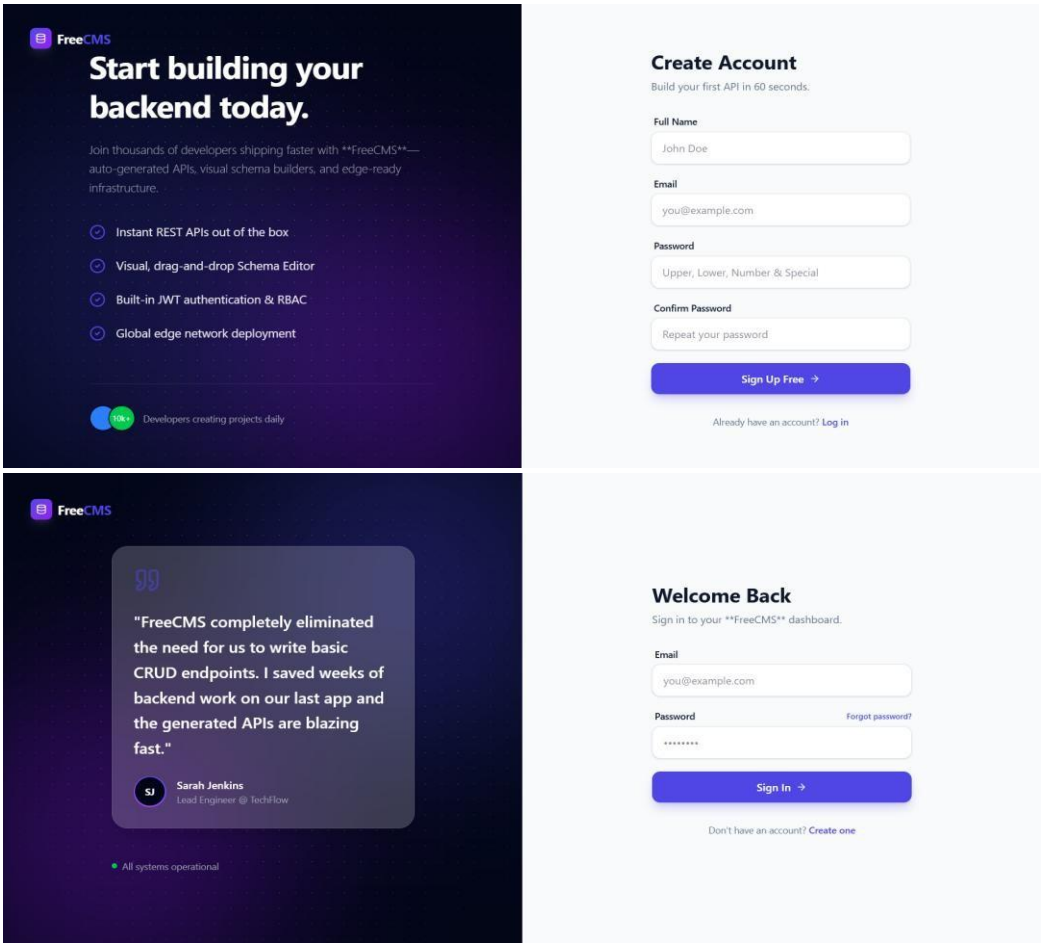
A minimalist authentication interface with secure JWT-based session handling and real-time form validation.

UI Components

- Login/Register form card
- Password visibility toggle
- Animated validation messages
- Primary authentication buttons

Workflow

1. Click “Sign Up” if you are a new user.
2. Enter your name, email, and password.
3. Submit the registration form.
4. Login using your credentials.
5. Upon successful authentication, the system redirects you to the Dashboard.



Screen 3: Administrative Multi-Tenant Dashboard

Role in Workflow

Serves as the command center where users manage multiple independent projects.

Description

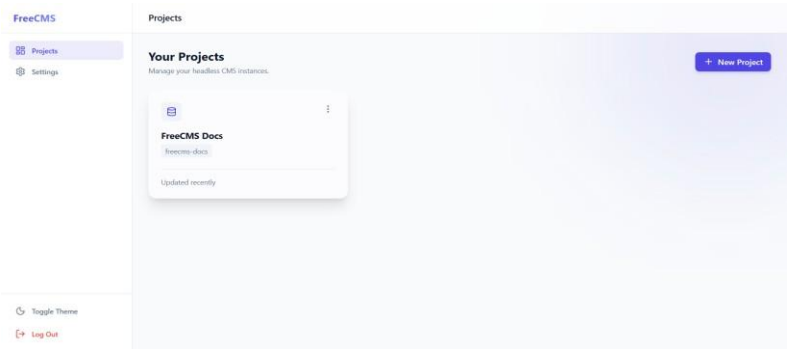
A project overview dashboard displaying active workspaces with quick management actions and metadata.

UI Components

- “Create New Project” button
- Searchable project grid
- Project cards with slugs and metadata

Workflow

1. Access the dashboard after login.
2. Click “New Project”.
3. Enter a project name and slug.
4. Create the project workspace.



5.

Screen 4: Project Workspace Hub (Overview)

Role in Workflow

Provides a centralized workspace for managing a specific project.

Description

Displays project-level summaries including models, content statistics, and API health indicators.

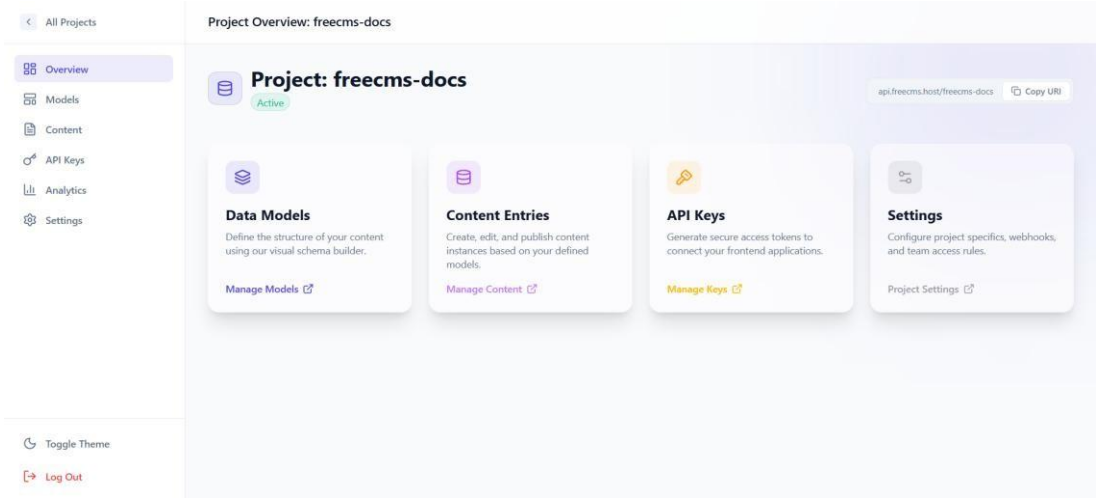
UI Components

- Sidebar navigation
- Model summary counters

- API status indicators
- Project overview widgets

Workflow

1. Select a project from the dashboard.
2. View project summaries and statistics.
3. Navigate to Models, Content, Analytics, or API sections.



Screen 5: Architectural Schema Designer (Model Builder)

Role in Workflow

Allows administrators to architect custom content structures visually.

Description

A dynamic schema builder used to define models such as Blogs, Products, or Pages.

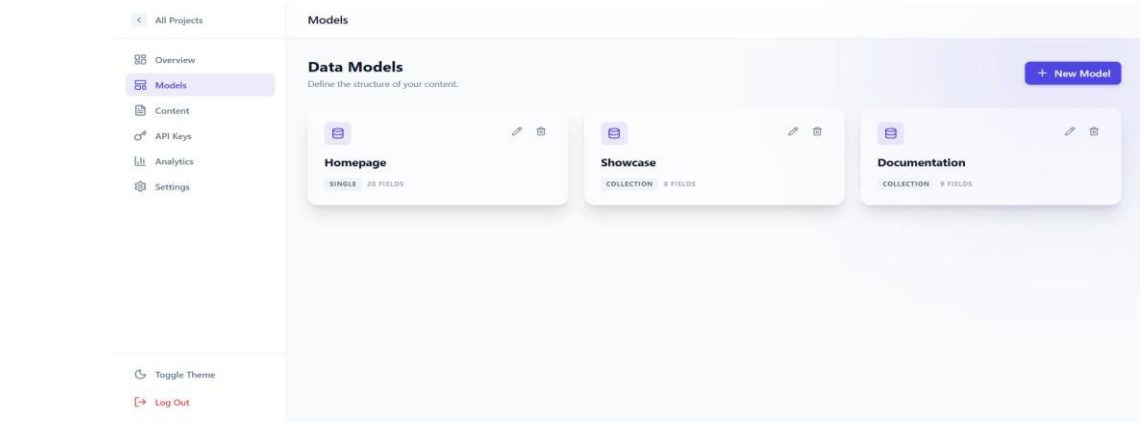
UI Components

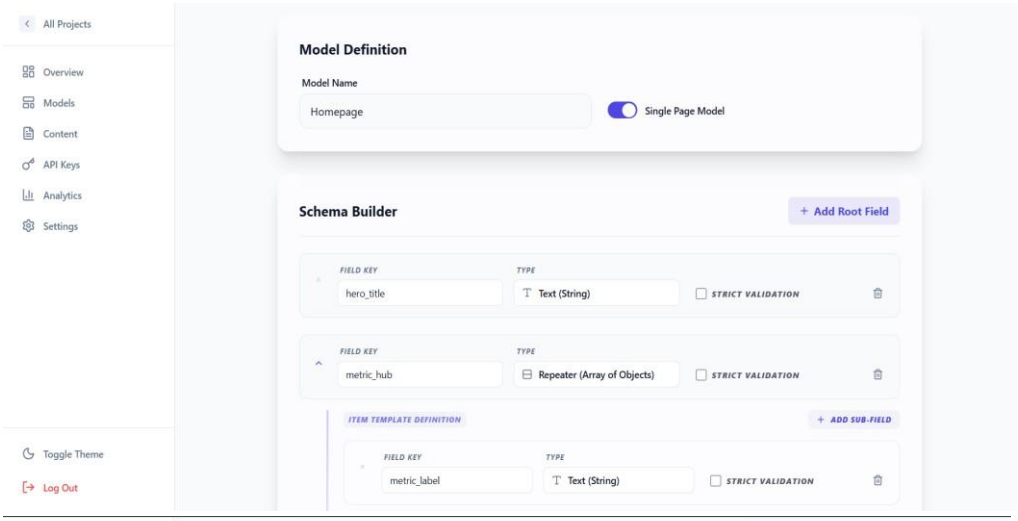
- Field-type sidebar
- Dynamic field configuration dialogs
- Singleton/Collection toggle
- Save Model button

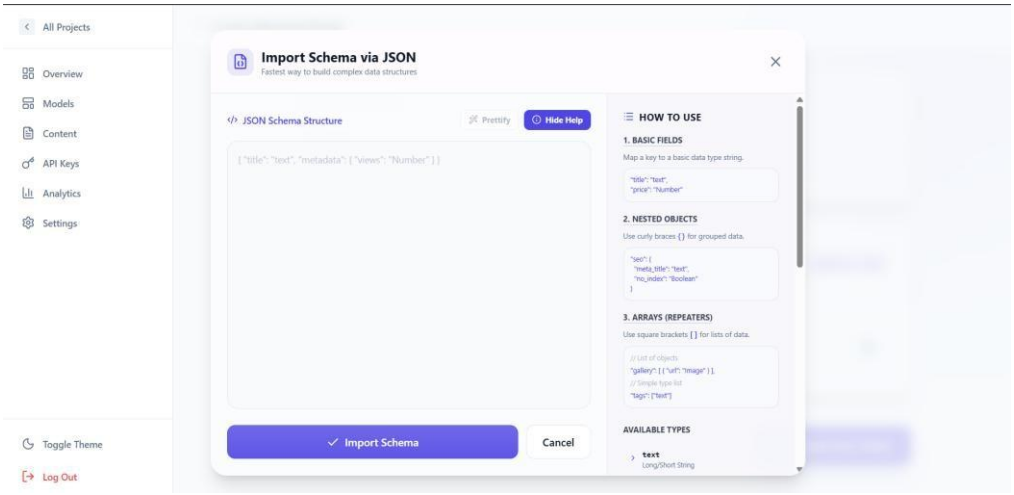
Workflow

1. Navigate to the “Models” section.
2. Click “Add Model”.
3. Select Collection or Singleton.
4. Add fields such as Text, Number, Image, or Rich Text.

   Save the model schema.







Screen 6: Content Operational Hub (Management Grid)

Role in Workflow

Handles content creation, editing, publishing, and operational workflows.

Description

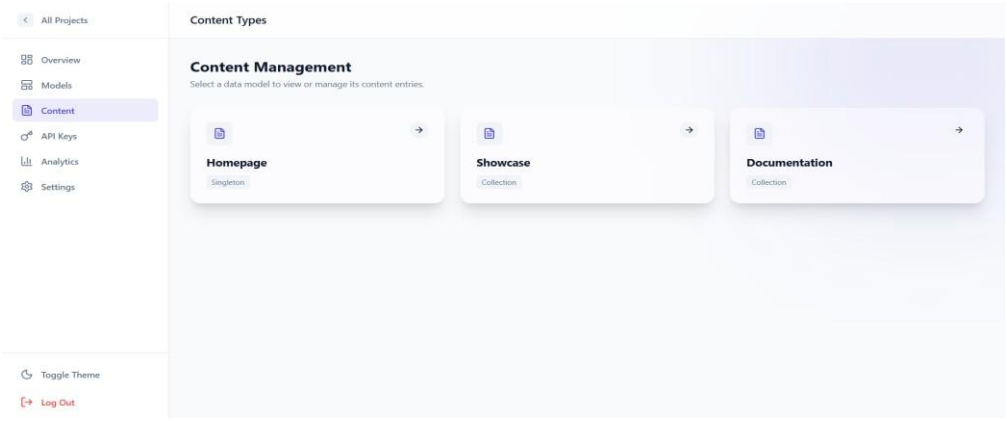
A content management interface displaying all entries associated with a selected model.

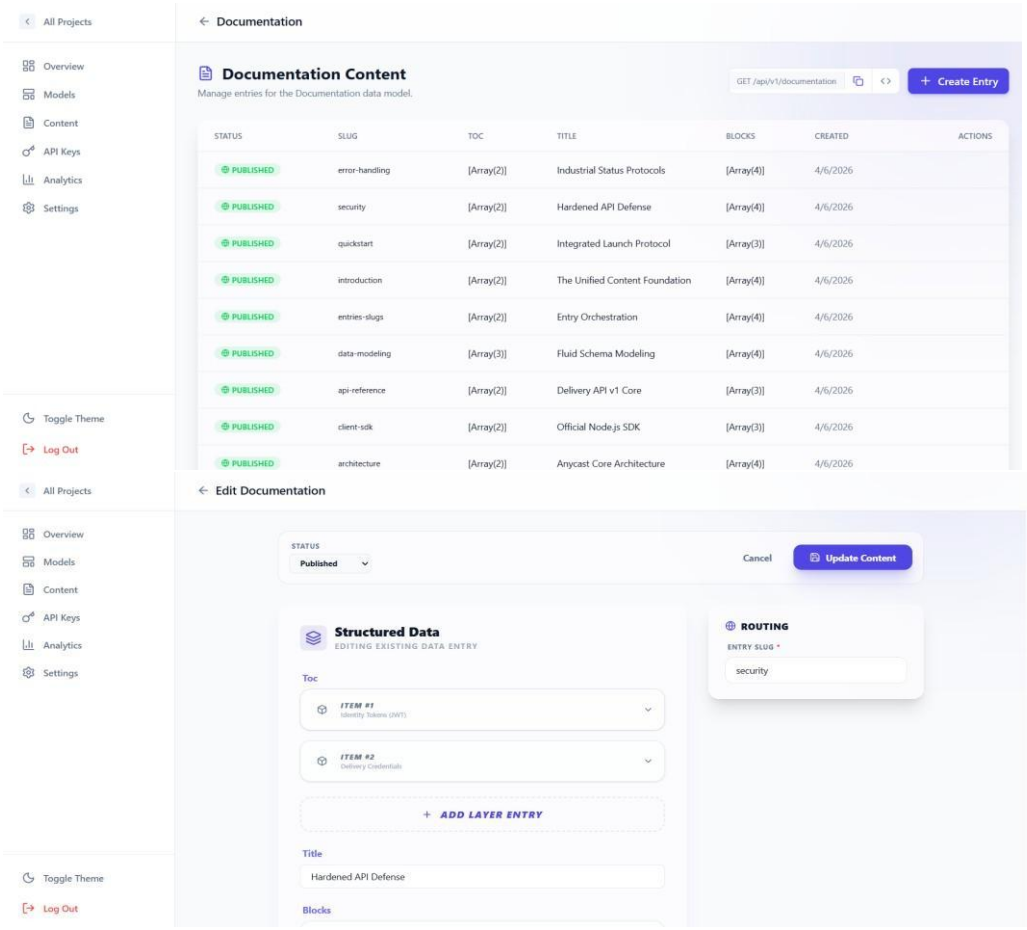
UI Components

- Sortable data table
- Draft/Published status badges
- Edit/Delete action buttons
- Dynamic content forms

Workflow

1. Open the “Content” section.
2. Select a model.
3. Click “Create Entry”.
4. Fill the generated form fields.
5. Save as Draft or Publish the content entry.





Screen 7: Security Credentialing Center (API Keys)

Role in Workflow

Manages secure API credentials for external application access.

Description

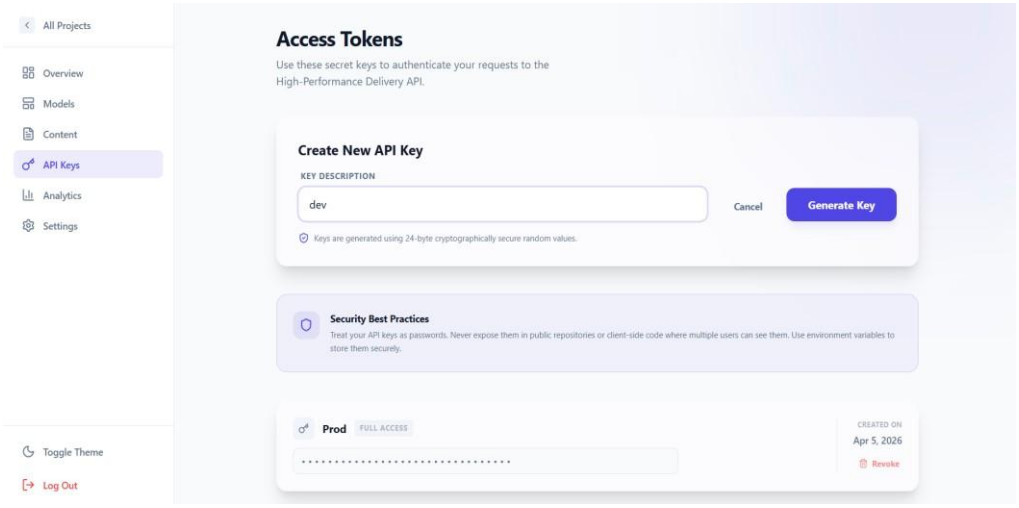
A protected utility for generating and managing API keys.

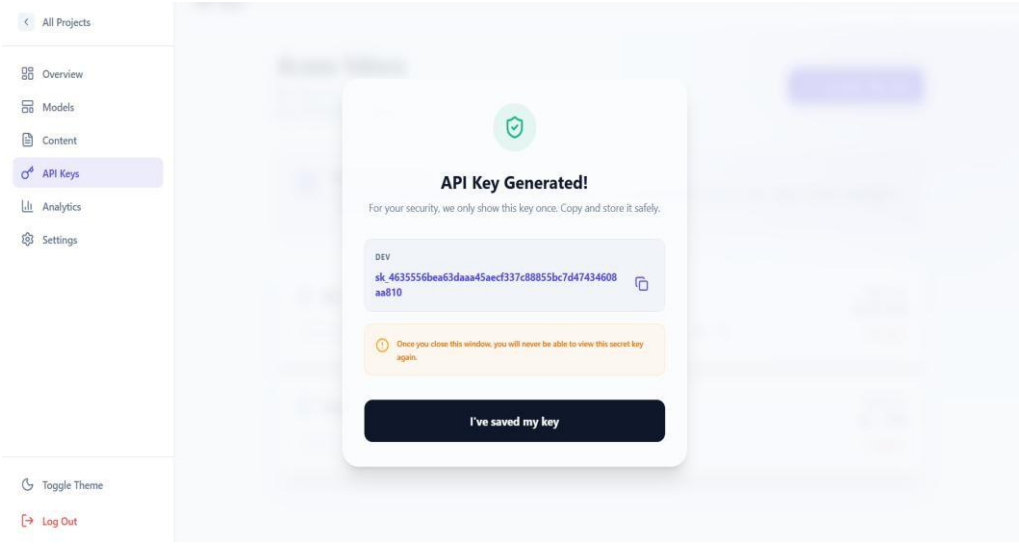
UI Components

- Generate Key button
- Copy-to-clipboard functionality
- Revoke key actions

Workflow

1. Navigate to the “API Keys” section.
2. Click “Generate New Key”.
3. Copy the generated secret key immediately.
4. Use the key in external applications or API clients.


All Projects
Overview
Models
Content
API Keys
Analytics
Settings
Toggle Theme
Log Out
Access Tokens
Use these secret keys to authenticate your requests to the High-Performance Delivery API.
Create New API Key
KEY DESCRIPTION
dev
Cancel
Generate Key
Keys are generated using 24-byte cryptographically secure random values.
Security Best Practices
Treat your API keys as passwords. Never expose them in public repositories or client-side code where multiple users can see them. Use environment variables to store them securely.
Prod
FULL ACCESS
CREATED ON
Apr 5, 2026
Revoke


All Projects
Overview
Models
Content
API Keys
Analytics
Settings
Toggle Theme
Log Out
API Key Generated!
For your security, we only show this key once. Copy and store it safely.
DEV
sk_4635556bea63daaa45aecf337c88855bc7d47434608aa810
Once you close this window, you will never be able to view this secret key again.
I've saved my key

Screen 8: Operational Intelligence Dashboard (Analytics)

Role in Workflow

Provides real-time monitoring and API usage analytics.

Description

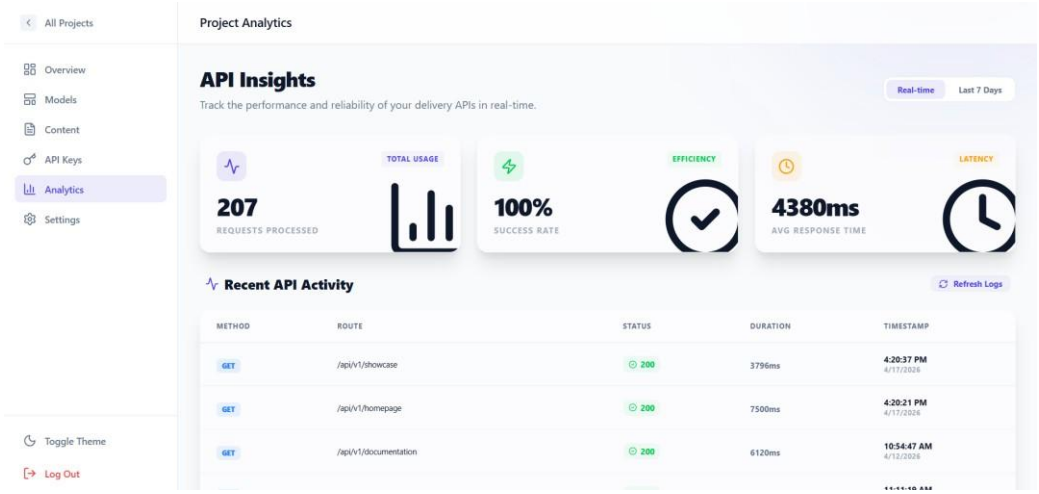
An analytics dashboard displaying request activity, response status, and API performance metrics.

UI Components

- Usage line charts
- API activity logs
- Performance summary cards

Workflow

1. Open the “Analytics” tab.
2. Monitor API traffic and latency.
3. Analyze response statistics and recent request activity.



Additional Operational Workflow

API Validation via Postman

Workflow

1. Open Postman or another API testing tool.
2. Set the request method to GET.

3. Enter the generated API endpoint URL.
4. Add the x-api-key header.
5. Send the request to retrieve published content.

Quick-Copy API Workflow

Workflow

1. Open a model’s content management page.
2. Click the “Copy API URL” button.
3. Paste the copied endpoint into your application or API client.

Frontend Integration (SDK Usage)

Installation *npm install freecms-*

client **Example Usage**

i*mport { CMSClient } from 'freecms-client'; const cms = new*

CMSClient('YOUR_API_KEY'); const data = await

cms.getEntries('your-model-slug');

Operational Monitoring & Settings

Workflow

1. Visit the Analytics section periodically.
2. Monitor API usage and performance.
3. Update CORS domains from the Settings section for secure frontend access.

8. Technical Architecture & Infrastructure

8.1 Technology Stack Frontend

Technology Purpose

Next.js 16	Frontend framework using App Router and React 19 features
TypeScript	Static type checking
Zustand	Global state management
Tailwind CSS 4	UI styling
Framer Motion	Animations and transitions
Lucide React	Icon library

Backend

Technology	Purpose
Node.js	Server runtime
Express.js	REST API framework
OpenAI SDK	AI content generation using GPT-4o-mini
JWT	Authentication
Bcrypt.js	Password hashing
express-rate-limit	API rate limiting
Dynamic CORS	Domain-based API access control

Database & Persistence

Technology	Purpose
PostgreSQL (Neon)	Primary relational database
Prisma ORM	Database ORM and migrations
MongoDB Atlas	AI prompt and response logging

Technology	Purpose
PgBouncer	Connection pooling
ClickHouse (Optional)	Analytics and request tracking

Media Management

Technology Purpose

Cloudinary	Media uploads, optimization, and CDN delivery

8.2 Data Flow Architecture

Management Flow

1. Admin updates Model Schema.
2. Backend stores schema in PostgreSQL.
3. Zustand updates frontend state.

AI Content Flow

1. User submits prompt.
2. Backend fetches Project Description and Model Schema.
3. Context is sent to OpenAI.
4. AI response is logged in MongoDB.
5. Generated JSON is returned to frontend.

Delivery Flow

1. Client requests content.
2. Middleware validates API Key and CORS Origin.
3. Content is fetched from PostgreSQL.
4. JSON response is returned.

8.3 Hosting & Deployment Strategy

The platform is deployed across a distributed infrastructure to ensure high availability:

- **Administrative Interface**: `https://freecms.host` (Vite-optimized deployment).
- **Core API Gateway**: `https://api.freecms.host` (Load-balanced Node.js environment).
- **Public Documentation**: `https://docs.freecms.host` (Static documentation deployment).