# **MAIN APPLICATION FLOW**

BEGIN African Critical Minerals Platform

INITIALIZE Flask web application

SET secret key for security

// Load mock database

minerals\_data = LOAD minerals information

countries\_data = LOAD African countries data

mining\_sites\_data = LOAD mining locations

users\_data = LOAD user accounts with roles

// Define application routes

CREATE route '/' for home page

CREATE route '/login' for user authentication

CREATE route '/dashboard' for main application

CREATE route '/admin' for administration

CREATE API routes for data access

START web server on port 8080

DISPLAY startup information

END PROGRAM

1. **USER AUTHENTICATION SYSTEM**

FUNCTION handle\_login(username, password):

IF username EXISTS IN users\_database THEN

IF password MATCHES stored\_password THEN

CREATE user session

SET session.username = username

SET session.role = user\_role

RECORD login time

REDIRECT to dashboard

ELSE

SHOW "Invalid password" error

END IF

ELSE

SHOW "User not found" error

END IF

END FUNCTION

FUNCTION check\_login\_required():

IF user NOT logged in THEN

REDIRECT to login page

ELSE

ALLOW access to page

END IF

END FUNCTION

FUNCTION handle\_logout():

CLEAR user session

REDIRECT to home page

END FUNCTION

1. **ROLE-BASED ACCESS CONTROL**

FUNCTION check\_role\_permission(required\_role):

current\_user\_role = GET user role from session

IF current\_user\_role == "admin" OR current\_user\_role == required\_role THEN

RETURN True // Allow access

ELSE

RETURN False // Deny access

END IF

END FUNCTION

DASHBOARD SYSTEM

text

FUNCTION show\_dashboard():

VERIFY user is logged in

GET user information from session

DISPLAY:

- Welcome message with user name and role

- Statistics cards (minerals count, countries count, sites count)

- Market intelligence section with price trends

- Interactive map of Africa with mineral sites

- Mineral information cards

- Country profile cards

- Mining site details

- API access information

IF user is investor OR admin THEN

SHOW investment opportunities section

END IF

IF user is admin THEN

SHOW admin panel link

END IF

END FUNCTION

1. **DATA MANAGEMENT**

FUNCTION get\_all\_minerals():

RETURN list of all minerals with:

- Name, description, price

- Production data, reserves

- Major producers, applications

- Risk factors, demand growth

END FUNCTION

FUNCTION get\_all\_countries():

RETURN list of African countries with:

- Name, capital, GDP, population

- Mining contribution to GDP

- Key minerals, investment rating

- Political stability, infrastructure

END FUNCTION

FUNCTION get\_mining\_sites():

RETURN list of mining sites with:

- Name, country, mineral type

- Coordinates (latitude, longitude)

- Production capacity, operator

- Employment, environmental rating

END FUNCTION

1. **INTERACTIVE MAP SYSTEM**

FUNCTION initialize\_map():

CREATE map centered on Africa

ADD base map layer from OpenStreetMap

FOR EACH country IN countries\_data:

CREATE marker at country coordinates

SET popup with country information

ADD marker to map

FOR EACH mining\_site IN mining\_sites\_data:

CREATE marker at site coordinates

SET popup with site details

ADD marker to map

ADD map legend and controls

RETURN interactive map

END FUNCTION

FUNCTION show\_location\_on\_map(location\_type, id):

IF location\_type == "country" THEN

FIND country by ID

CENTER map on country coordinates

OPEN country information popup

ELSE IF location\_type == "mining\_site" THEN

FIND mining site by ID

CENTER map on site coordinates

OPEN site information popup

END IF

END FUNCTION

1. **API SYSTEM**

FUNCTION api\_get\_minerals():

VERIFY user is logged in

RETURN minerals data in JSON format:

status: "success"

data: minerals\_list

count: number\_of\_minerals

timestamp: current\_time

END FUNCTION

FUNCTION api\_get\_countries():

VERIFY user is logged in

RETURN countries data in JSON format

END FUNCTION

FUNCTION api\_get\_mining\_sites():

VERIFY user is logged in

RETURN mining sites data in JSON format

END FUNCTION

FUNCTION api\_get\_market\_intelligence():

VERIFY user is logged in

RETURN market data with:

price\_trends for each mineral

market\_news with impact ratings

END FUNCTION

1. **ADMINISTRATION PANEL**

FUNCTION show\_admin\_panel():

VERIFY user is logged in

VERIFY user has admin role

DISPLAY:

- User management table

- System statistics

- Activity logs

- Database operations:

\* Import data

\* Export data

\* Update prices

\* Clean database

END DISPLAY

END FUNCTION

1. **ACTIVITY TRACKING**

FUNCTION track\_user\_activity(description):

IF user is logged in THEN

CREATE activity record with:

timestamp: current\_time

activity: description

user: username

ADD activity to user's activity log

KEEP only last 10 activities

END IF

END FUNCTION

MARKET INTELLIGENCE

text

FUNCTION get\_market\_data():

RETURN structured data with:

PRICE\_TRENDS for each mineral:

current\_price

change\_percentage

trend\_direction (up/down/stable)

MARKET\_NEWS:

title, summary, date

impact\_level (high/medium/low)

END FUNCTION

1. **INVESTMENT OPPORTUNITIES**

FUNCTION get\_investment\_opportunities():

VERIFY user is investor OR admin

RETURN list of projects with:

title, country, mineral

investment\_required

estimated\_return

risk\_level

timeline, status

END FUNCTION

ERROR HANDLING

text

FUNCTION handle\_errors():

IF page\_not\_found THEN

RETURN custom 404 error page

ELSE IF access\_denied THEN

RETURN "Insufficient permissions" message

ELSE IF server\_error THEN

RETURN custom 500 error page

END IF

END FUNCTION

1. **SESSION MANAGEMENT**

FUNCTION manage\_user\_session():

WHEN user logs in:

CREATE new session

STORE user details

SET login timestamp

INITIALIZE activity log

WHEN user accesses pages:

UPDATE last activity time

RECORD user actions

WHEN user logs out:

CLEAR session data

STOP tracking

END FUNCTION