**PlateFul- Setup & Usage Documentation**

**Overview**

This is a Flask-based web application that provides AI-powered recipe recommendations. Users can input ingredients and receive personalized recipe suggestions. The app includes user authentication, subscription management with trial periods, and payment processing through IntaSend.

**Features**

* User registration and authentication
* 14-day free trial for new users
* AI-powered recipe recommendations using OpenRouter API
* Subscription-based access with IntaSend payment integration
* Recipe history and management
* Admin panel for trial management
* MySQL database for data persistence

**Prerequisites**

Before setting up the application, ensure you have:

* Python 3.8 or higher
* MySQL 5.7 or higher
* Git (for version control)
* A web browser for testing

**Installation**

**1. Clone the Repository**

git clone <repository-url>

cd recipe-recommendation-app

**2. Set Up Virtual Environment**

python -m venv venv

source venv/bin/activate # On Windows: venv\Scripts\activate

**3. Install Dependencies**

pip install flask

pip install mysqlclient

pip install python-dotenv

pip install openai

pip install bcrypt

pip install requests

**Database Setup**

**1. Create MySQL Database**

CREATE DATABASE recipe\_app;

USE recipe\_app;

**2. Create Database Tables**

-- Users table

CREATE TABLE users (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(100) NOT NULL,

email VARCHAR(100) UNIQUE NOT NULL,

password VARCHAR(255) NOT NULL,

trial\_end\_date DATETIME,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- Subscription plans table

CREATE TABLE subscription\_plans (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(50) NOT NULL,

price DECIMAL(10,2) NOT NULL,

duration\_days INT NOT NULL,

is\_active BOOLEAN DEFAULT TRUE,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- Subscriptions table

CREATE TABLE subscriptions (

id INT AUTO\_INCREMENT PRIMARY KEY,

user\_id INT NOT NULL,

plan\_id INT NOT NULL,

status ENUM('trial', 'active', 'cancelled', 'expired') DEFAULT 'trial',

start\_date DATETIME NOT NULL,

end\_date DATETIME NOT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (user\_id) REFERENCES users(id),

FOREIGN KEY (plan\_id) REFERENCES subscription\_plans(id)

);

-- Payments table

CREATE TABLE payments (

id INT AUTO\_INCREMENT PRIMARY KEY,

user\_id INT NOT NULL,

subscription\_id INT,

plan\_id INT NOT NULL,

amount DECIMAL(10,2) NOT NULL,

status ENUM('pending', 'completed', 'failed', 'cancelled') DEFAULT 'pending',

payment\_method VARCHAR(50) DEFAULT 'intasend',

transaction\_id VARCHAR(255),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (user\_id) REFERENCES users(id),

FOREIGN KEY (subscription\_id) REFERENCES subscriptions(id),

FOREIGN KEY (plan\_id) REFERENCES subscription\_plans(id)

);

-- Recipes table

CREATE TABLE recipes (

id INT AUTO\_INCREMENT PRIMARY KEY,

recipe\_name VARCHAR(255) NOT NULL,

ingredients TEXT NOT NULL,

instructions TEXT NOT NULL,

user\_id INT NOT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (user\_id) REFERENCES users(id)

);

**3. Insert Sample Subscription Plans**

INSERT INTO subscription\_plans (name, price, duration\_days, is\_active) VALUES

('Monthly', 999.00, 30, TRUE),

('Quarterly', 2499.00, 90, TRUE),

('Yearly', 8999.00, 365, TRUE);

**Environment Configuration**

Create a .env file in the root directory with the following variables:

# Flask Configuration

SECRET\_KEY=your-secret-key-here-make-it-long-and-random

# MySQL Database Configuration

MYSQL\_HOST=localhost

MYSQL\_USER=your-mysql-username

MYSQL\_PASSWORD=your-mysql-password

MYSQL\_DB=recipe\_app

# OpenRouter API Configuration

OPENROUTER\_API\_KEY=your-openrouter-api-key

OPENROUTER\_MODEL=openai/gpt-3.5-turbo

OPENROUTER\_REFERER=http://localhost:5000

# IntaSend Payment Configuration (Test Mode)

INTASEND\_TEST\_MODE=true

INTASEND\_PUBLIC\_KEY\_TEST=your-intasend-test-public-key

INTASEND\_SECRET\_KEY\_TEST=your-intasend-test-secret-key

INTASEND\_WEBHOOK\_SECRET=your-webhook-secret

# IntaSend Payment Configuration (Live Mode)

INTASEND\_PUBLIC\_KEY\_LIVE=your-intasend-live-public-key

INTASEND\_SECRET\_KEY\_LIVE=your-intasend-live-secret-key

**Environment Variables Explained**

**Required Variables**

* **SECRET\_KEY**: Flask session encryption key (generate a random 32+ character string)
* **MYSQL\_**\*: Database connection credentials
* **OPENROUTER\_API\_KEY**: API key from OpenRouter for AI recipe generation

**Payment Integration (IntaSend)**

* **INTASEND\_TEST\_MODE**: Set to true for testing, false for production
* **INTASEND\_PUBLIC\_KEY\_TEST/LIVE**: Your IntaSend public keys
* **INTASEND\_SECRET\_KEY\_TEST/LIVE**: Your IntaSend secret keys
* **INTASEND\_WEBHOOK\_SECRET**: For webhook signature verification

**API Keys Setup**

**1. OpenRouter API Key**

1. Visit [OpenRouter.ai](https://openrouter.ai/)
2. Create an account
3. Navigate to API section
4. Generate an API key
5. Add credits to your account for API usage

**2. IntaSend API Keys**

1. Visit [IntaSend.com](https://intasend.com/)
2. Create an account
3. Go to Developer settings
4. Generate test and live API keys
5. Set up webhook endpoint: https://yourdomain.com/intasend-webhook

**Running the Application**

**1. Start the Application**

python app.py

The application will run on http://localhost:5000

**2. Access the Application**

Open your web browser and navigate to:

* Main app: http://localhost:5000
* Admin panel: http://localhost:5000/admin/trial

**Usage Guide**

**User Registration and Login**

1. **Registration**:
   * Navigate to /register
   * Enter name, email, and password
   * New users get a 14-day free trial automatically
2. **Login**:
   * Navigate to /login
   * Use email and password to log in

**Getting Recipe Recommendations**

1. **Input Ingredients**:
   * On the main page, enter available ingredients
   * Separate multiple ingredients with commas
2. **Generate Recipes**:
   * Click "Get Recommendations"
   * The AI will generate 3 recipe suggestions
   * Recipes are automatically saved to your account
3. **View Recipe History**:
   * Scroll down to see your previously generated recipes
   * Recipes show creation date and time

**Subscription Management**

1. **View Subscription Status**:
   * Main page shows current subscription status
   * Trial users see days remaining
2. **Upgrade Subscription**:
   * Navigate to /subscription
   * Choose from available plans
   * Complete payment through IntaSend
3. **Payment Process**:
   * Click "Subscribe" on chosen plan
   * Redirected to IntaSend payment page
   * Complete payment with M-Pesa or card
   * Automatically redirected back to app

**Admin Functions**

The admin panel (/admin/trial) allows you to:

* View all users and their trial status
* Extend trial periods
* Set specific trial end dates
* End trials immediately
* Bulk extend all active trials

**File Structure**

recipe-app/

├── app.py # Main Flask application

├── .env # Environment variables (not in repo)

├── requirements.txt # Python dependencies

├── templates/

│ ├── index.html # Main page with recipe recommendations

│ ├── login.html # User login page

│ ├── register.html # User registration page

│ ├── subscription.html # Subscription management page

│ └── admin\_trial.html # Admin panel for trial management

├── static/

│ ├── css/

│ │ └── style.css # Main stylesheet (needs to be created)

│ ├── js/

│ │ └── script.js # Frontend JavaScript (needs to be created)

│ └── images/

└── README.md

**Required Static Files**

You'll need to create the following static files:

**1. static/css/style.css - Main Stylesheet**

The HTML templates reference this stylesheet. Create a basic CSS file with styles for:

* Layout and typography
* Form styling
* Button styles (primary-btn, secondary-btn, logout-btn)
* Card layouts for recipes and plans
* Responsive design
* Loading spinner styles
* Flash message styling

**2. static/js/script.js - Frontend JavaScript**

The main page references this script for:

* Recipe recommendation functionality
* Ingredient tag interactions
* Loading states
* AJAX calls to backend APIs

**API Endpoints**

**Authentication**

* GET / - Main page (requires login)
* GET /login - Login page
* POST /login - Process login
* GET /register - Registration page
* POST /register - Process registration
* GET /logout - Logout

**Recipe Management**

* POST /get\_recommendations - Generate recipe recommendations
* GET /get\_user\_recipes - Retrieve user's recipe history

**Subscription Management**

* GET /subscription - Subscription management page
* POST /create\_subscription - Create payment with IntaSend
* GET /payment\_callback - Handle payment return
* POST /verify\_payment - Manual payment verification
* POST /intasend-webhook - Handle IntaSend webhooks

**Troubleshooting**

**Common Issues**

1. **Database Connection Failed**
   * Check MySQL service is running
   * Verify credentials in .env file
   * Ensure database and tables exist
2. **OpenRouter API Errors**
   * Verify API key is correct
   * Check account has sufficient credits
   * Ensure model name is correct
3. **Payment Issues**
   * Verify IntaSend credentials
   * Check webhook URL is accessible
   * Review IntaSend dashboard for transaction status
4. **Trial/Subscription Issues**
   * Use admin panel to check user status
   * Verify trial end dates in database
   * Check subscription table for active records

**Debug Mode**

Enable debug logging by setting:

app.run(debug=True)

**Logs and Monitoring**

Key areas to monitor:

* Payment webhook calls
* OpenRouter API responses
* Database connection status
* User subscription states

**Security Considerations**

1. **Environment Variables**: Never commit .env file to version control
2. **Database Security**: Use strong passwords and limit database access
3. **API Keys**: Rotate keys regularly and monitor usage
4. **HTTPS**: Use HTTPS in production for secure payment processing
5. **Input Validation**: The app includes basic input sanitization

**Deployment**

For production deployment:

1. **Database**: Use a managed MySQL service
2. **Environment**: Set INTASEND\_TEST\_MODE=false
3. **Domain**: Configure proper domain for IntaSend callbacks
4. **SSL**: Enable HTTPS for payment security
5. **Monitoring**: Set up logging and error tracking

**Support and Maintenance**

**Regular Maintenance Tasks**

* Monitor API usage and costs
* Review payment transactions
* Update subscription plans as needed
* Clean old recipe data if needed
* Monitor trial conversion rates

**Updates and Modifications**

* The app is designed to be easily extensible
* Add new recipe categories or cuisines
* Implement additional payment methods
* Add email notifications for subscriptions
* Implement recipe sharing features

**License and Usage**

This application is provided as-is for educational and commercial use. Ensure you comply with:

* OpenRouter API terms of service
* IntaSend payment processing agreements
* Local data protection regulations
* MySQL licensing requirements