Version 0.2

CODING WITH AI

BUILDING AI AGENTS

BOOK 05 FOR AGES 9+

BY REHAN ALLAHWALA
Technology Educator & Digital Innovator

Preface

Welcome to Book 9—a step-by-step guide to building AI agents using only free tools! This book empowers you to create automated solutions that solve real-world problems while keeping costs to zero. Whether you're a beginner or a coding enthusiast, this book will teach you how to use modern AI technologies, open APIs, and free platforms to develop practical and scalable agents.

By the end of this book, you'll have built several functional agents, mastered free tools, and gained confidence in automating complex tasks.

Let's dive into the world of AI agents without spending a penny!

Introduction to Agents!

What Are AI Agents?

Why: Agents are software programs designed to perform specific tasks automatically using artificial intelligence. They work independently, processing input and providing results without constant human intervention. AI agents simplify complex tasks, enhance productivity, and solve real-world problems efficiently.

What: An AI agent performs one well-defined function or task using AI capabilities. **For example:**

- A Job Analyzer Agent reviews resumes and provides feedback.
- An Email Agent fetches cryptocurrency updates and sends them via email.

AI agents can be considered specialized tools or widgets equipped with intelligence to perform specific functions autonomously.

How: To create an AI agent, you:

1. Identify the task (e.g., analyzing resumes or sending email alerts).
2. Choose the right tools and technologies (free APIs, libraries, or platforms).
3. Combine AI capabilities, such as GPT or OCR, with coding to build the solution.
4. Test and deploy it for users.

Why Use Free Tools?

Why: Free tools make learning accessible to everyone. They reduce the financial barrier, allowing students and developers to focus on learning, experimenting, and building practical solutions.

What: This book focuses on teaching you how to build AI agents using only free tools and platforms.

Examples of free tools include:

- OpenAI Free Tier: For text-based AI capabilities.
- Twilio Free API: For sending SMS and emails.
- Binance API: For real-time cryptocurrency updates.

How:

- Use OpenAI's free GPT API to handle natural language tasks.
- Integrate free APIs like Binance or Twilio for data fetching and communication.
- Host your projects for free using platforms like Render, Gradio, or Streamlit.
- Leverage free libraries in Python for functionalities like web scraping, email handling, and file management.

How This Book Is Organized

Why: To help you build a strong foundation in AI agent development while working on real-world tasks.

What: This book is structured into tasks that gradually increase in complexity. Each task is practical and designed to teach you specific concepts related to AI agents. By the end of the book, you will have built multiple functional agents ready for deployment.

How:

- Daily Task Lists: Each day, you will learn to create a new agent or feature.
- Step-by-Step Instructions: Detailed guidance to help you understand every step.
- Checklists: Ensure you complete all components of a task.
- GitHub and Social Media Sharing: Learn to publish your projects on GitHub and share progress online with hashtags like #rehancodingwithai.

Tools You Will Use in This Book

Tool/Platform	Purpose	Example Use
OpenAI Free API	AI text generation and analysis	Resume analysis, chatbot responses
Twilio Free API	Sending SMS and emails	Sending Bitcoin price updates
Binance API	Real-time cryptocurrency data	Fetching Bitcoin and Ethereum rates
Google Cloud Vision	Image and text analysis (OCR)	Extracting text from CVs
DALL-E Free	Generating images	Creating seasonal greeting cards
Streamlit/Gradio	Hosting user interfaces	Building interactive tools for agents
BeautifulSoup (Python)	Web scraping	Extracting contact details from websites

How to Use These Tools:

1. Read the task instructions carefully.
2. Install the required libraries and set up accounts on free platforms (e.g., Twilio, OpenAI).
3. Follow the provided code snippets to integrate the tools into your project.

TASK:01  SCORE: 10/	Job Analyzer Agent	Day:01

Objective

Create a Job Analyzer Agent that allows users to upload their CVs, analyzes the content, and provides detailed feedback with a quality score. The agent will highlight errors, suggest improvements, and publish the CV on a demo website.

What You'll Build

- A web-based interface for uploading CVs (PDF, Word, or image format).
- A text analyzer for grammar checks, missing details (e.g., photo, skills), and formatting feedback.
- A scoring system to rate the CV out of 100.
- A feedback report generator.
- A platform to host and publish the CV for sharing.

Steps

Setup Environment

- Install VS Code or another code editor.
- Install Python and required libraries:
  - requests for API calls.
  - PyPDF2 for reading PDFs.
  - python-docx for Word documents.

Gather Free APIs

- Free Alternatives to OpenAI API:
  - Use Cohere API (Free Tier) for text analysis. Sign up for Cohere Free Tier
  - For grammar and spelling checks, use LanguageTool API (Free). LanguageTool API.

- **Free Alternatives to Google Cloud Vision API**:
  - Use OCR.Space API (Free) for Optical Character Recognition. OCR.Space.

TASK:01  SCORE: 10/	Job Analyzer Agent	Day:01

Create the Agent

Step 3.1: Build the Upload Interface

- Use Gradio or Streamlit to create a web interface for CV uploads.
- Ensure compatibility with PDF, Word, and image formats.

Step 3.2: Extract Text from CVs

- For PDF and Word files, use Python libraries like PyPDF2 and python-docx.
- For images, use OCR.Space API to extract text for free.

Step 3.3: Analyze the Text

- Pass the extracted text to LanguageTool API for grammar and spelling checks.
- Use Cohere API to identify missing details and provide structure feedback.
- Develop a scoring algorithm that evaluates the CV based on predefined criteria.

Step 3.4: Generate a Feedback Report

- Create a feedback report using Python.
- Include:
  - Spelling/grammar errors.
  - Missing details (e.g., name, photo).
  - Formatting suggestions.

Step 3.5: Publish the CV

- Host the CV and feedback report on a free hosting platform:
  - Replit: For easy deployment.
  - GitHub Pages: For static websites.
- **Test Your Agent**
  - Upload test CVs in different formats to ensure functionality.
  - Verify:
    - Accurate text extraction and analysis.
    - Correct generation of feedback reports.
    - Successful hosting and accessibility of published CVs.

TASK:01  SCORE: 10/	Job Analyzer Agent	Day:01

Checklist

- Functional CV upload interface.
- Text extraction using free OCR and text analysis APIs.
- Feedback report generation with errors, suggestions, and a score.
- CV hosted on a free platform.

Social Media Caption

"Just completed my Job Analyzer Agent! Upload CVs, get instant analysis, and improve them for job opportunities—all for free! 🚀 #rehancodingwithai #codingwithai"

TASK:02  SCORE: 10/	Email Agent	Day:02

Objective

Create an Email Agent that fetches daily updates about Bitcoin and Ethereum rates from cryptocurrency platforms and sends personalized emails to users. This task will utilize free APIs and tools for fetching data and sending emails automatically.

What You'll Build

- A system to fetch real-time cryptocurrency rates (Bitcoin and Ethereum).
- An automated email sender to deliver updates to users daily.
- A web-based interface for users to subscribe and receive email updates.

Steps

1. Setup Environment
    - Install VS Code or another preferred code editor.
    - Install Python and required libraries:
        - requests for API calls.
        - smtplib for sending emails.

2,**Gather Free APIs and Tools**

Cryptocurrency Data API:

- Use CoinGecko API (Free) to fetch cryptocurrency rates.
- Sign up for CoinGecko API

3.**Email Sending Tools:**

- Use Google SMTP (Gmail) or any free SMTP service for sending emails.
- If you prefer alternatives, try SendGrid Free Tier (Sign up for SendGrid).

TASK:02  SCORE: 10/	Email Agent	Day:02

Create the Agent

Step 3.1: Fetch Cryptocurrency Data

- Use CoinGecko API to fetch live rates for Bitcoin and Ethereum.
- Extract the required details (e.g., current price, 24-hour change).

Step 3.2: Automate Email Sending

- Use smtplib or SendGrid API to set up an email sender.
- Write a personalized email template that includes:
  - Cryptocurrency name (Bitcoin/Ethereum).
  - Current rate and percentage change.

Step 3.3: Build a Subscription System

- Use Gradio or Streamlit to create a web-based interface:
  - Allow users to subscribe by entering their email address.
  - Store email addresses in a local database or file.

Step 3.4: Schedule Daily Emails

- Use Python's schedule library or a free task scheduler like cron-job.org to automate daily email sending.
- Ensure the script runs daily at a fixed time.

4.Test Your Agent

- Add sample email addresses to the subscription system.
- Fetch cryptocurrency data and ensure accurate integration.
- Verify that emails are sent successfully with the correct data.

TASK:02  SCORE: 10/	Email Agent	Day:02

Checklist

- Real-time cryptocurrency data fetcher (Bitcoin and Ethereum).
- Automated email sender using SMTP or SendGrid API.
- Subscription system to collect user emails.
- Daily email scheduling.

Why CoinGecko API?

CoinGecko API is a reliable and free option for fetching real-time cryptocurrency data.

Can We Use Other APIs?

Yes, alternatives like CoinMarketCap Free Tier or Binance API can also be used for fetching cryptocurrency rates.

Social Media Caption

"Just completed my Email Agent! Fetches daily Bitcoin and Ethereum updates and sends them directly to your inbox—all automated! 🚀 #rehancodingwithai #codingwithai"

TASK:03 SCORE: 10/	International Outreach Agent	Day:03

Objective

Create an International Outreach Agent that automatically finds the contact details of individuals (e.g., senators, CEOs, or other professionals) from a website or directory, generates personalized seasonal greeting messages in multiple languages, and sends them using Twilio or a similar messaging API.

What You'll Build

- A system to scrape or collect contact details (emails or WhatsApp numbers) from any specified source.
- A greeting generator that supports multilingual messages.
- An automated messaging tool to send greetings globally via SMS, WhatsApp, or email.

Steps

Step 1: Setup Environment

1. Install VS Code or another code editor.
2. Install Python and required libraries:
    - requests for API calls.
    - BeautifulSoup for web scraping.
    - twilio for sending SMS/WhatsApp messages.
    - smtplib if using email.

Step 2: Gather Free APIs and Tools

1. Messaging API:
    - Use Twilio Free Tier for sending SMS/WhatsApp messages.
    - Alternatives: Fast2SMS (free for limited messages) or Sendinblue for email.
2. Translation API:
    - Use LibreTranslate API (Free) for multilingual translations.
    - LibreTranslate API

TASK:03  SCORE: 10/	International Outreach Agent	Day:03

Step 3: Create the Agent

Step 3.1: Scrape or Collect Contact Details

- Use BeautifulSoup to scrape public directories, LinkedIn profiles, or other sources for contact details.
- Save the scraped data (e.g., name, email, WhatsApp number) in a CSV or JSON file.

Example Websites:

- Government directories for senators.
- LinkedIn for professionals (export connections manually if needed).
- Public business directories like Yellow Pages.

Step 3.2: Generate Personalized Greetings

1. Write a script to create generic seasonal greetings like:
   - "Wishing you a joyous holiday season!"
2. Use LibreTranslate API to translate the greeting into multiple languages based on the recipient's location.
   - Example:
     - For English: "Happy New Year!"
     - For French: "Bonne année!"
     - For Bengali: "নতুন বছর শুভ হোক!"
3. Combine all messages into a single greeting if necessary.

Step 3.3: Automate Message Sending

1. For SMS/WhatsApp:
   - Use Twilio API to send personalized messages.
   - Each message should include the recipient’s name and the greeting in their preferred language.

TASK:03  SCORE: 10/	International Outreach Agent	Day:03

- Each message should include the recipient's name and the greeting in their preferred language.

2.For Email:

- Use smtplib or Sendinblue to send the greetings via email.
- Format the email with the recipient's name, location, and multilingual greeting.

Step 3.4: Automate Scheduling

- Use the schedule library or a free task scheduler like cron-job.org to automate the script.
- Configure the agent to run during festive seasons or specific intervals.

Step 4: Test Your Agent

1. Scrape or manually input a list of 5–10 sample contacts.
2. Verify:
   - Accurate scraping or collection of contact details.
   - Correct generation of multilingual greetings.
   - Successful sending of messages via SMS/WhatsApp/email.

Checklist

- A functional web scraper or contact input system.
- Greeting generator supporting multilingual messages.
- Automated message sender using Twilio or an email API.
- Scheduled task for running the outreach agent automatically.

Why Twilio API?

Twilio provides robust messaging services for both SMS and WhatsApp, making it ideal for international outreach.

Can We Use Other APIs?

Yes, alternatives like Fast2SMS, Plivo, or Nexmo can also be used. For email, services like SendGrid or Mailgun are excellent choices.

Social Media Caption

"Just completed my International Outreach Agent! Automatically finds contacts, creates multilingual greetings, and sends them globally! 🚀 #rehancodingwithai #codingwithai"

TASK:04  SCORE: 10/	Seasonal Greeting Image Tool Agent	Day:04

Objective

Create a Seasonal Greeting Image Tool Agent that generates personalized greeting images for special occasions (e.g., holidays, New Year, birthdays) using AI. The agent will allow users to create and customize greeting images, share them with friends and family, and post them on social media platforms like Twitter.

What You'll Build

- A tool for generating seasonal greeting images using AI.
- A system to customize messages, fonts, and colors on the images.
- An option to share the created images via WhatsApp, email, and social media.

Steps

Step 1: Setup Environment

1. Install VS Code or any other code editor.
2. Install Python and required libraries:
    - requests for API calls.
    - Pillow for image manipulation.
    - flask or Gradio for creating a user interface.

Step 2: Gather Free APIs and Tools

1. AI Image Generation:
    - Use DALL-E API (Free Tier) to generate creative backgrounds for greeting cards.
        - OpenAI DALL-E Signup
    - Alternative: Use Craiyon (Free DALL-E Alternative) for generating images.
2. Text Customization Tools:
    - Use Pillow (Python Library) for overlaying text on the images.

1. **Sharing Options:**
    - Use Twilio API for sending images via WhatsApp.
    - Use Twitter API for posting on social media.

TASK:04  SCORE: 10/	Seasonal Greeting Image Tool Agent	Day:04

Step 3: Create the Agent

Step 3.1: Generate Background Image

- Use DALL-E API or Craiyon to create a visually appealing background image for greetings.
- Example Prompt: "Create a festive holiday greeting background with snowflakes and a glowing Christmas tree."

Step 3.2: Add Custom Text to the Image

- Use Pillow to overlay personalized messages on the image.
- Allow customization of:
  - Message text (e.g., "Happy New Year 2024!")
  - Font size and style.
  - Text color and position.

Step 3.3: Build the Interface

- Use Gradio or Flask to create a simple web interface where users can:
  - Select or generate a background image.
  - Enter their personalized message.
  - Preview the final image.

Step 3.4: Enable Sharing Options

1. Send via WhatsApp:
   - Use Twilio API to send the generated image to multiple recipients.
2. Post on Social Media:
   - Use Twitter API to allow users to post the image directly to their Twitter account.
3. Email Sharing:
   - Use Python’s smtplib to attach the image and send it via email.

Step 4: Test Your Agent

1. Test image generation with different prompts and backgrounds.
2. Customize text and verify overlays on generated images.
3. Share test images via WhatsApp, email, and Twitter to ensure successful delivery and posting.

TASK:04  SCORE: 10/	Seasonal Greeting Image Tool Agent	Day:04

Checklist

- AI-powered image generation for greeting backgrounds.
- Text overlay tool with customizable options.
- Sharing functionality via WhatsApp, email, and social media.
- User-friendly web interface for creating and sharing images.

Why DALL-E API?

DALL-E offers high-quality and creative AI-generated images for free (with limits).

Can We Use Other Free Tools?

Yes, Craiyon or other open-source image generators can be used as alternatives for DALL-E.

Social Media Caption

"Just completed my Seasonal Greeting Image Tool Agent! Generate, customize, and share festive greetings with AI—quick and easy! 🚀 #rehancodingwithai #codingwithai"

TASK:05 SCORE: 10/	Resume Builder Agent	Day:05

Objective

Create a Resume Builder Agent that generates professional and customized resumes based on user inputs, job descriptions, or LinkedIn profiles. The agent will allow users to select templates, add details, and download resumes in multiple formats such as PDF or Word.

What You'll Build

- A web-based interface for users to input details or upload job descriptions.
- A system to generate resumes dynamically using AI.
- Support for multiple resume templates with download options.

Steps

Step 1: Setup Environment

1. Install VS Code or any other code editor.
2. Install Python and required libraries:
    - jinja2 for creating HTML templates.
    - pdfkit or WeasyPrint for converting HTML to PDF.
    - flask or Gradio for building a user interface.

Step 2: Gather Free APIs and Tools

1. AI for Content Generation:
    - Use Cohere API (Free Tier) for generating professional summary and skills.
        - Cohere API Signup
    - Alternatively, use Hugging Face Transformers (Open Source) for text generation.
2. **Resume Templates:**
    - Use free resume template repositories like Overleaf or Canva.
    - Create HTML-based templates for dynamic data insertion.

TASK:05 SCORE: 10/	Resume Builder Agent	Day:05

Step 3: Create the Agent

Step 3.1: Build the Input Form

- Use Gradio or Flask to create a form where users can:
  - Enter their personal details (name, contact, education, experience).
  - Upload job descriptions or LinkedIn profile data.
  - Select a preferred resume template.

Step 3.2: Generate Resume Content

- Use AI (e.g., Cohere API) to:
  - Write a professional summary based on user inputs.
  - Suggest relevant skills and achievements for the selected job role.
- Organize data into predefined sections (e.g., Education, Work Experience, Skills).

Step 3.3: Format Resume Using Templates

- Create HTML-based resume templates with placeholders for dynamic content.
- Use jinja2 to insert user-provided data into the selected template.

Step 3.4: Convert to Downloadable Format

- Use pdfkit or WeasyPrint to convert the formatted HTML resume into a PDF.
- Provide download options for PDF and Word formats.

Step 3.5: Allow Customization

- Enable users to preview the resume and make edits directly in the interface.

Step 4: Test Your Agent

1. Enter sample data into the form and generate resumes in different templates.
2. Verify:
   - AI-generated summaries and skills match the job description.
   - Formatting is consistent across all templates.
   - Resume downloads are error-free in PDF and Word formats.

TASK:05  SCORE: 10/	Resume Builder Agent	Day:05

Checklist

- User-friendly form for inputting details or uploading job descriptions.
- AI-generated content for professional summaries and skills.
- Multiple resume templates for users to choose from.
- Downloadable resumes in PDF and Word formats.

Why Cohere API?

Cohere’s free tier allows efficient and professional text generation, ideal for creating summaries and skills.

Can We Use Other Free Tools?

Yes, Hugging Face Transformers or OpenAI Playground (Free Tier) can also be used for text generation.

Social Media Caption

"Just completed my Resume Builder Agent! Build professional resumes dynamically with AI and download them in minutes. 🚀 #rehancodingwithai #codingwithai"

TASK:06  SCORE: 10/	Interview Coach Agent	Day:06

Objective

Create an Interview Coach Agent that simulates job interviews by asking questions, analyzing user responses, and providing feedback to help improve interview skills. The agent will also offer tips for better communication and suggest improvements in answers.

What You'll Build

- A conversational interface for real-time interview simulations.
- An AI-driven system to analyze user responses and provide feedback.
- A summary report highlighting strengths, weaknesses, and tips for improvement.

Steps

Step 1: Setup Environment

1. Install VS Code or any other code editor.
2. Install Python and required libraries:
    - flask or Gradio for building a user interface.
    - speech_recognition for voice input (optional).
    - pytesseract if you need OCR for text extraction.

Step 2: Gather Free APIs and Tools

1. AI for Question-Answer Analysis:
    - Use Hugging Face Transformers (Free) for natural language understanding and response analysis.
        - Hugging Face Transformers
    - Alternatively, use Cohere API (Free Tier) for response evaluation.

2. **Voice Input (Optional):**
    - Use the Google Speech-to-Text API for converting spoken answers to text.

TASK:06  SCORE: 10/	Interview Coach Agent	Day:06

Step 3: Create the Agent

Step 3.1: Build the Interview Interface

- Use Gradio or Flask to create an interface where users can:
  - Select a job category or role (e.g., Marketing, Engineering, HR).
  - Start a mock interview with predefined or AI-generated questions.
  - Provide answers via text or voice input.

Step 3.2: Generate and Ask Questions

- Preload a set of interview questions for different roles or use AI to generate them dynamically.
- Display one question at a time and collect user responses.

Step 3.3: Analyze User Responses

- Use AI (e.g., Hugging Face Transformers) to evaluate the response based on:
  - Relevance to the question.
  - Grammar and language quality.
  - Confidence and clarity.
- Assign a score (e.g., 1–5) for each response and provide detailed feedback.

Step 3.4: Generate Feedback and Tips

- After the interview, compile the feedback into a report:
  - Highlight strong answers with explanations.
  - Suggest areas of improvement (e.g., "Provide more specific examples in your answers").
  - Offer general tips for better communication and body language.

Step 4: Test Your Agent

1. Test the mock interview with sample questions and responses.
2. Verify:
   - Accurate analysis of user responses.
   - Feedback is relevant and actionable.
   - Voice input (if enabled) works seamlessly.

TASK:06  SCORE: 10/	Interview Coach Agent	Day:06

Checklist

- A functional interface for conducting mock interviews.
- AI-generated questions tailored to job roles.
- Response analysis and scoring system.
- Comprehensive feedback report with tips for improvement.

Why Hugging Face Transformers?

Hugging Face provides powerful and free models for natural language processing, making it ideal for analyzing interview responses.

Can We Use Other Free Tools?

Yes, Cohere API or even OpenAI Playground (Free Tier) can be used for analyzing responses.

Social Media Caption

"Just completed my Interview Coach Agent! Simulates job interviews, analyzes your answers, and provides feedback to help you ace your next interview. 🚀 #rehancodingwithai #codingwithai"

TASK:07  SCORE: 10/	Job Matchmaker Agent	Day:07

Objective

Create a Job Matchmaker Agent that scans job boards, matches users with roles based on their skills, location, and interests, and sends them personalized job recommendations. This agent will automate the job search process, making it faster and more efficient.

What You'll Build

- A system to scrape or access job listings from online job boards.
- A matching engine that compares user profiles with job requirements.
- A notification system to deliver job recommendations via email or WhatsApp.

Steps

Step 1: Setup Environment

1. Install VS Code or another code editor.
2. Install Python and required libraries:
    - requests for API calls.
    - BeautifulSoup for web scraping.
    - smtplib for email notifications.

Step 2: Gather Free APIs and Tools

1. Job Board APIs (Free):
    - Use Indeed API or LinkedIn Jobs API (limited free access).
    - Alternatively, scrape job boards like Glassdoor or Monster if APIs are unavailable.
2. Matching Algorithm:
    - Use Python libraries like fuzzywuzzy or difflib for skill and requirement matching.
3. Notification System:
    - Use SendGrid API or smtplib for sending job alerts via email.
    - Use Twilio API for sending WhatsApp or SMS notifications.

TASK:07  SCORE: 10/	Job Matchmaker Agent	Day:07

Step 3: Create the Agent

Step 3.1: Build the User Profile Input System

- Create a form using Gradio or Flask to collect user details:
  - Name, email, location, and preferred job roles.
  - Skills and experience.
  - Preferred job boards (e.g., Indeed, LinkedIn).

Step 3.2: Fetch Job Listings

- Use job board APIs or web scraping to collect job postings based on user preferences.
- Extract relevant information from each listing:
  - Job title, company name, location, and skills required.

Step 3.3: Match Jobs with User Profiles

- Compare user skills and preferences with job descriptions using a matching algorithm.
- Rank job listings based on relevance and save the top matches.

Step 3.4: Notify Users

- Generate a personalized job recommendation email or WhatsApp message for the user.
- Include:
  - Job title, company, and a brief description.
  - A direct link to apply for the job.
- Send the notifications via SendGrid or Twilio API.

Step 3.5: Automate the Process

- Schedule the agent to run daily or weekly using the schedule library or cron-job.org.

Step 4: Test Your Agent

1. Input sample user profiles with different skill sets and preferences.
2. Verify:
   - Job listings are fetched accurately.
   - The matching algorithm ranks jobs correctly.
   - Notifications are delivered successfully.

TASK:07  SCORE: 10/	Job Matchmaker Agent	Day:07

Checklist

- A functional user profile input system.
- Job listing fetcher using APIs or web scraping.
- Matching algorithm for job-user compatibility.
- Notification system for delivering personalized job alerts.

Why Use Indeed or LinkedIn APIs?

These platforms provide reliable and up-to-date job listings, making them ideal for fetching relevant opportunities.

Can We Use Other Free Tools?

Yes, you can scrape publicly available job boards like Glassdoor, Rozee.pk, or Monster.

Social Media Caption

"Just completed my Job Matchmaker Agent! Matches skills with jobs and sends personalized recommendations—making job hunting easy! 🚀 #rehancodingwithai #codingwithai"

TASK:08  SCORE: 10/	Stock Tracker Agent	Day:08

Objective

Create a Stock Tracker Agent that monitors selected stocks and cryptocurrencies, tracks their price changes in real-time, and sends alerts based on user-defined criteria. The agent will provide users with instant updates to make timely investment decisions.

What You'll Build

- A system to fetch real-time stock and cryptocurrency prices.
- A notification system to alert users of price changes.
- A user-friendly interface to select stocks, set criteria, and receive alerts.

Steps

Step 1: Setup Environment

1. Install VS Code or any other code editor.
2. Install Python and required libraries:
    - requests for API calls.
    - smtplib or Twilio API for notifications.

Step 2: Gather Free APIs and Tools

1. **Stock and Cryptocurrency APIs (Free):**
    - Use Yahoo Finance API (via yfinance library) for stock prices.
    - Use CoinGecko API for cryptocurrency prices.

2. **Notification System:**
    - Use Twilio API or smtplib for sending alerts via SMS or email.
    - Alternatively, use Telegram Bot API for free notifications.

TASK:08  SCORE: 10/	Stock Tracker Agent	Day:08

Step 3: Create the Agent

Step 3.1: Build the Input Form

- Create a form using Gradio or Flask for users to:
  - Select stocks or cryptocurrencies to track.
  - Set price thresholds for alerts (e.g., "Notify me if Bitcoin drops below $30,000").

Step 3.2: Fetch Real-Time Prices

- Use yfinance library to fetch stock prices from Yahoo Finance.
- Use CoinGecko API to fetch cryptocurrency prices.
- Example Code Snippet for Crypto Prices:
- python
- Copy code
- import requests

```
def get_crypto_price(crypto_id):
    url = f"https://api.coingecko.com/api/v3/simple/price?ids={crypto_id}&vs_currencies=usd"
    response = requests.get(url)
    return response.json()
```

Step 3.3: Compare Prices with User Criteria

- Continuously monitor prices and compare them with user-defined thresholds.
- If the price matches the criteria, trigger an alert.

Step 3.4: Send Alerts

- Use Twilio API to send SMS alerts.
- Use smtplib to send email alerts if Twilio is unavailable.
- Example Alert Message:
- "Alert! Bitcoin price just dropped below $30,000. Current price: $29,800."

TASK:08  SCORE: 10/	Stock Tracker Agent	Day:08

Step 3.5: Automate the Monitoring

- Use Python's schedule library to run the script at regular intervals (e.g., every minute).

Step 4: Test Your Agent

1. Add sample stocks and cryptocurrencies to the tracker.
2. Set criteria and verify alerts are sent when prices hit thresholds.
3. Test both SMS and email notifications to ensure reliability.

Checklist

- A form for users to select stocks/cryptocurrencies and set alert criteria.
- Real-time price fetcher using Yahoo Finance and CoinGecko APIs.
- Alert system for sending notifications via SMS or email.
- Automated monitoring and alert generation.

Why Use Yahoo Finance and CoinGecko APIs?

These APIs are free, reliable, and provide real-time data for both stocks and cryptocurrencies.

Can We Use Other Free Tools?

Yes, alternatives like Alpha Vantage or Binance API can also be used for similar functionality.

Social Media Caption

"Just completed my Stock Tracker Agent! Monitors stocks and crypto in real-time and sends instant alerts—stay ahead of the market! 🚀 #rehancodingwithai #codingwithai"

TASK:09 SCORE: 10/	Language Translator Agent	Day:09

Objective

Create a Language Translator Agent that translates documents, messages, or websites into multiple languages in real-time. The agent will allow users to input text or upload files and receive translations in their desired language.

What You'll Build

- A web interface to input text or upload documents.
- A translation engine supporting multiple languages.
- Downloadable translated files for documents.
- Real-time translation for instant results.

Steps

Step 1: Setup Environment

1. Install VS Code or another code editor.
2. Install Python and required libraries:
    - requests for API calls.
    - Flask or Gradio for building a user interface.

Step 2: Gather Free APIs and Tools

1. Free Translation API:
    - Use LibreTranslate API (completely free).
        - LibreTranslate API Signup
    - Alternative: Use Google Translate API (with free trial).
2. File Handling:
    - Use libraries like PyPDF2 for PDFs and python-docx for Word documents.

TASK:09  SCORE: 10/	Language Translator Agent	Day:09

Step 3: Create the Agent

Step 3.1: Build the Input Interface

- Use Gradio or Flask to create a user-friendly interface:
  - Allow users to:
    - Input text for instant translation.
    - Upload documents (PDFs or Word files) for translation.
  - Select source and target languages from a dropdown menu.

Step 3.2: Fetch Translations

- For text translation:
  - Send the input text to LibreTranslate API and retrieve the translated text.
- For document translation:
  - Extract text from uploaded files using PyPDF2 or python-docx.
  - Translate the extracted text using the translation API.

Step 3.3: Generate Translated Output

- For text:
  - Display the translated text in the interface.
- For documents:
  - Replace the original text in the document with the translated text.
  - Allow users to download the translated document in the original format.

Step 3.4: Real-Time Translation

- Enable real-time translation for short messages or conversations.
- Example Use Case: Translate live chat messages between two users.

Step 4: Test Your Agent

1. Test text translation with various languages and phrases.
2. Upload sample documents and verify:
   - Accurate text extraction.
   - Correct and meaningful translations.
   - Proper formatting in the output file.
3. Test the speed and accuracy of real-time translations.

TASK:09  SCORE: 10/	Language Translator Agent	Day:09

Checklist

- Input interface for text and file uploads.
- Translation engine powered by LibreTranslate API.
- Real-time translation for instant results.
- Downloadable translated documents with preserved formatting.

Why Use LibreTranslate API?

LibreTranslate is free, open-source, and supports multiple languages, making it an ideal choice for this task.

Can We Use Other Free Tools?

Yes, Google Translate API (Free Trial) or open-source libraries like Fairseq can also be used.

Social Media Caption

"Just completed my Language Translator Agent! Translate documents and messages into multiple languages in real-time—all for free! 🚀 #rehancodingwithai #codingwithai"

TASK:10  SCORE: 10/	Content Writer Agent	Day:10

Objective

Create a Content Writer Agent that generates blog posts, captions, or articles tailored to specific topics or audiences. The agent will allow users to input a topic or keywords and produce well-structured and engaging content that can be used for blogs, social media, or marketing purposes.

What You'll Build

- A web interface where users can input topics or keywords.
- An AI-driven content generator that creates detailed content based on user input.
- Options for users to customize the tone, style, or length of the content.

Steps

Step 1: Setup Environment

1. Install VS Code or any other code editor.
2. Install Python and required libraries:
    - requests for API calls.
    - Flask or Gradio for building a user interface.

Step 2: Gather Free APIs and Tools

1. Content Generation API:
    - Use Cohere API (Free Tier) for generating creative content.
        - Cohere API Signup
    - Alternatives: Use Hugging Face Transformers for open-source text generation.
2. Customization and Formatting Tools:
    - Use Python's built-in libraries for text formatting or simple Markdown libraries for clean output.

TASK:10  SCORE: 10/	Content Writer Agent	Day:10

Step 3: Create the Agent

Step 3.1: Build the Input Interface

- Use Gradio or Flask to create a user-friendly interface:
  - Allow users to input:
    - Topic or keywords (e.g., "Benefits of AI in Education").
    - Content preferences:
      - Tone: Professional, Casual, or Creative.
      - Length: Short (100–200 words), Medium (500–800 words), or Long (1000+ words).

Step 3.2: Generate Content

- Pass the user's input to the Cohere API or Hugging Face Transformers:
  - Generate content based on the given topic or keywords.
  - Adjust the output's tone and length based on user preferences.

- **Example Code Snippet for Content Generation:**
- python
- Copy code

```
import cohere

def generate_content(topic, length, tone):
    client = cohere.Client('YOUR_API_KEY')
    response = client.generate(
        model='xlarge',
        prompt=f"Write a {length} article in a {tone} tone about {topic}",
        max_tokens=500
    )
    return response.generations[0].text
```

TASK:10  SCORE: 10/	Content Writer Agent	Day:10

Step 3.3: Preview and Edit Content

- Display the generated content in the interface.
- Allow users to make edits directly before downloading or copying.

Step 3.4: Enable Download and Sharing Options

- Provide options for:
  - Downloading the content as a text or Markdown file.
  - Copying the content directly for use in blogs or social media posts.

Step 4: Test Your Agent

1. Test with various topics and preferences:
   - Short captions for social media.
   - Detailed blog posts for professional audiences.
2. Verify that:
   - The content aligns with the user's input and preferences.
   - Output is well-structured, grammatically correct, and engaging.

Checklist

- User-friendly interface for inputting topics and preferences.
- AI-driven content generation tailored to user needs.
- Options to preview, edit, and download content.
- Clean and structured output for blogs, captions, or articles.

Why Use Cohere API?

Cohere provides an easy-to-use free tier for generating high-quality text, making it ideal for content writing.

Can We Use Other Free Tools?

Yes, Hugging Face Transformers or OpenAI Playground (Free Tier) can also be used for generating content.

Social Media Caption

"Just completed my Content Writer Agent! Generates high-quality blog posts, captions, and articles instantly—tailored to your needs. 🚀 #rehancodingwithai #codingwithai"

TEN DAYS REPORT SHEET!

Task No.	Task Name	Score (Out of 10)	Remarks
Task 1	Job Analyzer Agent		
Task 2	Email Agent		
Task 3	International Outreach Agent		
Task 4	Seasonal Greeting Image Tool Agent		
Task 5	Resume Builder Agent		
Task 6	Interview Coach Agent		
Task 7	Job Matchmaker Agent		
Task 8	Stock Tracker Agent		
Task 9	Language Translator Agent		
Task 10	Content Writer Agent		

TASK:11  SCORE: 10/	Social Media Scheduler Agent	Day:11

Objective

Create a Social Media Scheduler Agent that helps users plan, schedule, and post content across multiple platforms like Twitter, Instagram, and Facebook to optimize audience engagement.

What You'll Build

- A web interface to schedule posts with text, images, and hashtags.
- A system to post scheduled content on social media platforms automatically.

Steps

1. Setup Environment
    - Install Python and required libraries:
        - Flask or Gradio for interface.
        - requests for API calls.
2. Free APIs and Tools
    - Use Meta Graph API for Facebook and Instagram.
    - Use Twitter API for posting on Twitter.
3. Build the Agent
    - Create a form to input post content, image, hashtags, and schedule time.
    - Save scheduled posts in a database or JSON file.
    - Use APIs to post content automatically at the scheduled time.
4. Test Your Agent
    - Test by scheduling posts with images and hashtags for each platform.
    - Verify that posts are published on time.

Checklist

- Functional scheduling interface.
- Social media integration via APIs.
- Automated posting system.

Social Media Caption

"Just completed my Social Media Scheduler Agent! Plan and post content seamlessly across platforms. 🚀 #rehancodingwithai #codingwithai"

TASK:12  SCORE: 10/	Fitness Tracker Agent	Day:12

Objective

Create a Fitness Tracker Agent that helps users monitor their daily activities, including steps, calories burned, and workout routines, and provides personalized fitness recommendations.

What You'll Build

- A system to track daily activity data.
- A recommendation engine for personalized fitness goals.
- A web interface to input or view fitness metrics.

Steps

1. Setup Environment
    - Install Python and required libraries:
        - Flask or Gradio for the interface.
        - matplotlib for activity visualization.
2. Free APIs and Tools
    - Use Google Fit API or OpenHealth API for tracking fitness data.
3. Build the Agent
    - Create a form to input steps, calories, and activity time manually or via API.
    - Display progress using charts and graphs.
    - Generate recommendations based on user fitness goals (e.g., "Increase steps by 10% tomorrow").
4. Test Your Agent
    - Input test data manually and via APIs.
    - Verify accurate tracking and actionable recommendations.

Checklist

- Functional input system for activity data.
- Integration with fitness APIs.
- Graphs and recommendations based on user goals.

Social Media Caption

"Just completed my Fitness Tracker Agent! Monitor daily activity and get personalized recommendations to stay healthy. 🚀 #rehancodingwithai #codingwithai"

TASK:13  SCORE: 10/	Learning Buddy Agent	Day:13

Objective

Create a Learning Buddy Agent that provides real-time tutoring on various subjects, quizzes users, and tracks learning progress. This agent acts as a virtual assistant for students to enhance their learning experience.

What You'll Build

- A system for real-time tutoring and answering user queries.
- A quiz generator based on the selected subject.
- A progress tracker to monitor performance.

Steps

1. Setup Environment
    - Install Python and libraries:
        - Flask or Gradio for the interface.
        - pandas for progress tracking.
2. Free APIs and Tools
    - Use Hugging Face Transformers for real-time tutoring.
    - Use Open Trivia Database API for quiz questions.
3. Build the Agent
    - Create a form for selecting a subject and entering questions.
    - Generate quiz questions using the API and evaluate user responses.
    - Store scores in a local database to track progress.
4. Test Your Agent
    - Test tutoring with sample questions.
    - Verify quizzes and scoring system.
    - Ensure accurate progress tracking.

Checklist

- Functional tutoring interface.
- Quiz generation and evaluation system.
- Progress tracker with user history.

Social Media Caption

"Just completed my Learning Buddy Agent! Real-time tutoring, quizzes, and progress tracking in one tool. 🚀 #rehancodingwithai #codingwithai"

TASK:14  SCORE: 10/	Mental Health Companion Agent	Day:14

Objective

Create a Mental Health Companion Agent that provides mindfulness exercises, daily affirmations, and journaling prompts to help users improve their mental well-being.

What You'll Build

- A system to suggest mindfulness exercises (e.g., meditation, breathing techniques).
- A daily affirmation generator.
- A journaling tool to log user thoughts and track mood.

Steps

1. Setup Environment
    - Install Python and libraries:
        - Flask or Gradio for the interface.
        - matplotlib for mood tracking graphs.
2. Free APIs and Tools
    - Use ZenQuotes API for daily affirmations.
    - Use Python's random module to suggest mindfulness exercises.
3. Build the Agent
    - Create an interface to display daily affirmations and mindfulness exercises.
    - Add a journaling section for users to log their thoughts.
    - Track user mood and visualize progress with graphs.
4. Test Your Agent
    - Test the affirmations and mindfulness suggestions for variety.
    - Verify journaling and mood tracking functionality.

Checklist

- Functional interface with affirmations and exercises.
- Journaling tool for user thoughts.
- Mood tracking and visualization.

Social Media Caption

"Just completed my Mental Health Companion Agent! Provides daily affirmations, mindfulness tips, and mood tracking for a healthier you. 🚀 #rehancodingwithai #codingwithai"

TASK:15  SCORE: 10/	Personal Finance Manager Agent	Day:15

Objective

Create a Personal Finance Manager Agent that tracks expenses, provides budgeting tips, and identifies saving opportunities to help users manage their finances effectively.

What You'll Build

- A system to log daily expenses and categorize them.
- A budget planner that calculates savings and spending limits.
- Suggestions for reducing expenses and increasing savings.

Steps

1. Setup Environment
   - Install Python and libraries:
     - Flask or Gradio for the interface.
     - pandas for data organization and analysis.
2. Free APIs and Tools
   - Use Python's matplotlib or plotly for data visualization.
3. Build the Agent
   - Create an interface for users to input expenses with categories (e.g., food, transport).
   - Display total expenses, categorized spending, and remaining budget.
   - Generate tips based on spending patterns (e.g., "Reduce dining expenses by 10% to save more").
4. Test Your Agent
   - Log sample expenses and verify accurate calculations.
   - Test the system's ability to generate actionable tips and graphs.

Checklist

- Functional expense input system.
- Budget tracker with visualizations.
- Expense analysis and saving tips.

Social Media Caption

"Just completed my Personal Finance Manager Agent! Tracks expenses, budgets smartly, and provides saving tips—your financial buddy! 🚀 #rehancodingwithai #codingwithai"

TASK:16  SCORE: 10/	Event Planner Agent	Day:16

Objective

Create an Event Planner Agent that helps users organize events, find venues, manage RSVPs, and create to-do lists for seamless event management.

What You'll Build

- A system to input event details and requirements.
- A venue finder and RSVP manager.
- A to-do list generator for event planning.

Steps

1. Setup Environment
    - Install Python and libraries:
        - Flask or Gradio for the interface.
        - pandas for data management.
2. Free APIs and Tools
    - Use Eventbrite API for venue suggestions.
    - Use Google Sheets API for RSVP tracking.
3. Build the Agent
    - Create an interface to input event details (e.g., name, date, budget).
    - Suggest venues based on location and budget.
    - Manage RSVP lists with options to update attendee status.
    - Generate a dynamic to-do list based on the event type (e.g., birthday, conference).
4. Test Your Agent
    - Test by planning different types of events.
    - Verify venue suggestions, RSVP management, and to-do list accuracy.

Checklist

- Event details input system.
- Venue finder and RSVP manager.
- Dynamic to-do list generator.

Social Media Caption

"Just completed my Event Planner Agent! Organize events, manage RSVPs, and create to-do lists effortlessly. 🚀 #rehancodingwithai #codingwithai"

TASK:17  SCORE: 10/	AI Chef Agent	Day:17

Objective

Create an AI Chef Agent that suggests recipes based on available ingredients, dietary preferences, and cuisine choices. The agent will help users create delicious meals effortlessly.

What You'll Build

- A system where users can input available ingredients.
- A recipe suggestion engine tailored to dietary needs and preferences.
- An option to view step-by-step cooking instructions.

Steps

1. Setup Environment
   - Install Python and libraries:
     - Flask or Gradio for the interface.
     - requests for API calls.
2. Free APIs and Tools
   - Use Spoonacular API (Free Tier) for recipe suggestions.
     - Spoonacular API Signup
3. Build the Agent
   - Create an interface to input ingredients and select dietary preferences (e.g., vegan, gluten-free).
   - Fetch recipe suggestions from the Spoonacular API based on user input.
   - Display a list of recipes with step-by-step instructions and nutritional information.
4. Test Your Agent
   - Input different combinations of ingredients and dietary preferences.
   - Verify that recipe suggestions match user requirements.

Checklist

- Ingredient input and dietary preference selection.
- Recipe suggestion engine integrated with Spoonacular API.
- Step-by-step instructions and nutritional information display.

Social Media Caption

"Just completed my AI Chef Agent! Suggests recipes based on your ingredients and preferences—turn leftovers into culinary masterpieces! 🚀 #rehancodingwithai #codingwithai"

TASK:18  SCORE: 10/	Virtual Travel Agent	Day:18

Objective

Create a Virtual Travel Agent that plans trips, books tickets, and suggests itineraries based on user preferences such as destination, budget, and travel dates.

What You'll Build

- A system to collect travel preferences like destination, dates, and budget.
- An itinerary generator with suggested activities and attractions.
- A ticket and accommodation finder integrated with booking platforms.

Steps

1. Setup Environment
   - Install Python and libraries:
     - Flask or Gradio for the interface.
     - requests for API calls.
2. Free APIs and Tools
   - Use Skyscanner API (Free Tier) for flight and accommodation search.
   - Use OpenTripMap API for activity and attraction suggestions.
3. Build the Agent
   - Create an interface to input travel preferences: destination, budget, and travel dates.
   - Fetch available flights, hotels, and attractions using Skyscanner and OpenTripMap APIs.
   - Generate a detailed itinerary with activities and estimated costs.
4. Test Your Agent
   - Test with different destinations and budgets.
   - Verify that the itinerary is practical and matches user preferences.

Checklist

- Travel preference input system.
- Flight, accommodation, and attraction search integration.
- Dynamic itinerary generator with estimated costs.

Social Media Caption

"Just completed my Virtual Travel Agent! Plans your trips, finds flights, and creates itineraries—your travel buddy! 🚀 #rehancodingwithai #codingwithai"

TASK:19  SCORE: 10/	Idea Generator Agent	Day:19

Objective

Create an Idea Generator Agent that provides creative ideas for businesses, content, or projects based on user input such as keywords, industries, or niches.

What You'll Build

- A system where users can input keywords or select categories.
- An AI-powered idea generator that suggests unique and actionable ideas.
- A functionality to refine ideas further or generate variations.

Steps

1. Setup Environment
   - Install Python and libraries:
     - Flask or Gradio for the interface.
     - requests for API calls.
2. Free APIs and Tools
   - Use Cohere API (Free Tier) for text generation.
     - Cohere API Signup
3. Build the Agent
   - Create an interface to input keywords or select a category (e.g., "Business," "Content," "Project").
   - Generate ideas using the Cohere API based on the provided input.
   - Display a list of 3–5 unique ideas with an option to refine or generate more.
4. Test Your Agent
   - Input different keywords and categories to test variety and relevance of ideas.
   - Verify the refinement feature generates logical variations.

Checklist

- Functional input system for keywords or categories.
- AI-powered idea generator using Cohere API.
- Refinement option for further idea exploration.

Social Media Caption

"Just completed my Idea Generator Agent! Get creative ideas for businesses, content, or projects with a single click. 🚀 #rehancodingwithai #codingwithai"

TASK:20  SCORE: 10/	E-Library Organizer Agent	Day:20

Objective

Create an E-Library Organizer Agent that helps users manage their digital library by categorizing books, PDFs, and documents and suggesting reading materials based on user preferences.

What You'll Build

- A system to upload and organize digital files.
- A categorization engine that sorts files by genre, topic, or type.
- A recommendation engine for suggesting reading materials.

Steps

1. Setup Environment
    - Install Python and libraries:
        - Flask or Gradio for the interface.
        - PyPDF2 for processing PDF metadata.
2. Free APIs and Tools
    - Use Google Books API to fetch book details and genres.
    - Use Python's os module for file management.
3. Build the Agent
    - Create an interface to upload books or documents (PDFs, eBooks).
    - Extract metadata (e.g., title, author) using PyPDF2 or file properties.
    - Fetch additional information (e.g., genre) from Google Books API.
    - Organize files into categories and display recommendations based on past uploads.
4. Test Your Agent
    - Upload a mix of documents and books.
    - Verify accurate categorization and relevant recommendations.

Checklist

- Functional upload system for digital files.
- Categorization engine for organizing files by genre or topic.
- Recommendation engine integrated with Google Books API.

Social Media Caption

"Just completed my E-Library Organizer Agent! Organize books, categorize PDFs, and get personalized reading recommendations. 🚀 #rehancodingwithai #codingwithai"

TEN DAYS REPORT SHEET!

Task No.	Task Name	Score (Out of 10)	Remarks
Task 11	Social Media Scheduler Agent		
Task 12	Fitness Tracker Agent		
Task 13	Learning Buddy Agent		
Task 14	Mental Health Companion Agent		
Task 15	Personal Finance Manager Agent		
Task 16	Event Planner Agent		
Task 17	AI Chef Agent		
Task 18	Virtual Travel Agent		
Task 19	Idea Generator Agent		
Task 20	E-Library Organizer Agent		

TASK:21  SCORE: 10/	Time-Travel Data Simulator Agent	Day:21

Objective

Simulate historical reconstructions or future predictions based on user-input timeframes, locations, or topics using AI-driven data analysis.

What You'll Build

- A system to simulate past or future scenarios.
- Interactive visualizations of predictions or reconstructions.

Steps

1. Setup Environment: Use Python, matplotlib, and Flask/Gradio.
2. APIs & Tools: Google Earth Engine API, Open Historical Map.
3. Build Agent: Input timeframes/locations, fetch historical or predictive data, and visualize.
4. Test: Run scenarios for various locations and timelines.

Checklist

- Functional simulation and visualization system.
- Accurate data reconstruction/prediction.

Social Media Caption

"Travel through time with AI! My Time-Travel Data Simulator Agent predicts the future and reconstructs the past. 🚀 #rehancodingwithai #codingwithai"

TASK:22  SCORE: 10/	Emotion Clone Agent	Day:22

Objective

Develop an AI agent that clones and emulates emotions based on user interactions, such as text, voice, or facial expressions, to provide empathetic responses and personalized support.

What You'll Build

- An emotion detection system to analyze user input.
- An emotion-emulation engine for personalized interactions.

Steps

1. Setup Environment: Use Python, Flask, and emotion analysis tools like DeepFace or Hugging Face.
2. APIs & Tools: Use Affectiva API or Microsoft Emotion Recognition API.
3. Build Agent: Analyze input (text, voice, or face) to detect emotions and generate appropriate responses.
4. Test: Test with various emotional inputs to ensure accurate detection and responses.

Checklist

- Accurate emotion detection.
- Empathetic and personalized responses.

Social Media Caption

"My Emotion Clone Agent mirrors your emotions for personalized and empathetic support. AI that truly feels! 🚀 #rehancodingwithai #codingwithai"

TASK:22  SCORE: 10/	Emotion Clone Agent	Day:22

Objective

Develop an AI agent that clones and emulates emotions based on user interactions, such as text, voice, or facial expressions, to provide empathetic responses and personalized support.

What You'll Build

- An emotion detection system to analyze user input.
- An emotion-emulation engine for personalized interactions.

Steps

1. Setup Environment: Use Python, Flask, and emotion analysis tools like DeepFace or Hugging Face.
2. APIs & Tools: Use Affectiva API or Microsoft Emotion Recognition API.
3. Build Agent: Analyze input (text, voice, or face) to detect emotions and generate appropriate responses.
4. Test: Test with various emotional inputs to ensure accurate detection and responses.

Checklist

- Accurate emotion detection.
- Empathetic and personalized responses.

Social Media Caption

"My Emotion Clone Agent mirrors your emotions for personalized and empathetic support. AI that truly feels! 🚀 #rehancodingwithai #codingwithai"

TASK:23  SCORE: 10/	Dream Visualizer Agent	Day:23

Objective

Create an AI agent that translates user-provided dream descriptions into stunning visual art, offering a unique representation of their subconscious thoughts.

What You'll Build

- A system to interpret textual dream descriptions.
- A visual generator that creates AI-driven art based on the descriptions.

Steps

1. Setup Environment: Use Python, Flask, and DALL-E or Stable Diffusion API for image generation.
2. APIs & Tools: Use OpenAI API for interpreting descriptions and generating prompts.
3. Build Agent: Input dream descriptions, convert them into art prompts, and generate visuals.
4. Test: Test with diverse descriptions to ensure accuracy and creativity.

Checklist

- Dream description input system.
- Stunning and accurate visual generation.

Social Media Caption

"Turn dreams into reality! My Dream Visualizer Agent creates stunning art from your subconscious. 🚀 #rehancodingwithai #codingwithai"

TASK:24  SCORE: 10/	AI Weather Manipulation Simulator Agent	Day:24

Objective

Create an AI agent that simulates weather manipulation scenarios, predicting the potential impact of artificially altering weather patterns for specific locations and events.

What You'll Build

- A system to simulate weather modification effects.
- Visualization of changes based on user input.

Steps

1. Setup Environment: Use Python, Flask, and weather prediction tools like OpenWeatherMap API.
2. APIs & Tools: Leverage OpenWeatherMap API and climate simulation datasets.
3. Build Agent: Input desired weather changes (e.g., increase rainfall), simulate effects, and visualize results.
4. Test: Simulate various weather-altering scenarios and verify accuracy.

Checklist

- Functional simulation engine for weather changes.
- Accurate visualization of predicted impacts.

Social Media Caption

"Explore the possibilities of weather manipulation! My AI Weather Simulator Agent visualizes climate-altering scenarios. 🚀 #rehancodingwithai #codingwithai"

TASK:25  SCORE: 10/	Task Automation Assistant Agent	Day:25

Objective

Create an AI agent that automates 50% of repetitive tasks by analyzing workflows, identifying automatable actions, and executing them seamlessly for increased productivity.

What You'll Build

- A system to analyze user workflows and identify repetitive tasks.
- Automation for tasks like email sorting, report generation, or file organization.

Steps

1. Setup Environment: Use Python, Flask, and pandas for workflow analysis.
2. APIs & Tools: Use Zapier API for integrations and task automation.
3. Build Agent: Input user workflows, identify repetitive tasks, and automate actions like moving files or sending reminders.
4. Test: Automate various repetitive tasks and measure time saved.

Checklist

- Workflow analysis and task identification system.
- Automated execution of repetitive tasks.

Social Media Caption

"Work smarter, not harder! My Task Automation Assistant Agent automates 50% of repetitive tasks to boost productivity. 🚀 #rehancodingwithai #codingwithai"

FIVE DAYS REPORT SHEET!

Task No.	Task Name	Score (Out of 10)	Remarks
Task 21	Time-Travel Data Simulator Agent		
Task 22	Emotion Clone Agent		
Task 23	Dream Visualizer Agent		
Task 24	AI Weather Manipulation Simulator Agent		
Task 25	Task Automation Assistant Agent		

FINAL REPORT SHEET!

Task No.	Task Name	Score (Out of 10)	Remarks
Task 1	Job Analyzer Agent		
Task 2	Email Agent		
Task 3	International Outreach Agent		
Task 4	Seasonal Greeting Image Tool Agent		
Task 5	Resume Builder Agent		
Task 6	Interview Coach Agent		
Task 7	Job Matchmaker Agent		
Task 8	Stock Tracker Agent		
Task 9	Language Translator Agent		
Task 10	Content Writer Agent		
Task 11	Social Media Scheduler Agent		
Task 12	Fitness Tracker Agent		
Task 13	Learning Buddy Agent		
Task 14	Mental Health Companion Agent		
Task 15	Personal Finance Manager Agent		
Task 16	Event Planner Agent		
Task 17	AI Chef Agent		
Task 18	Virtual Travel Agent		
Task 19	Idea Generator Agent		
Task 20	E-Library Organizer Agent		
Task 21	Time-Travel Data Simulator Agent		
Task 22	Emotion Clone Agent		
Task 23	Dream Visualizer Agent		
Task 24	AI Weather Manipulation Simulator Agent		
Task 25	Task Automation Assistant Agent		
Total		/250	

Conclusion

Congratulations on completing this journey! Through these 26 tasks, you've explored the limitless potential of AI, from automating workflows to creating futuristic solutions that seemed impossible. This book wasn't just about learning—it was about empowering you to build, innovate, and shape the future.

AI is your tool to solve problems, create impact, and unlock new opportunities. Use the skills you've gained to push boundaries, take on challenges, and continue creating AI agents that make a difference. The future starts with you—keep building, keep innovating!

Coding with AI

Building Ai Agents

Create, Explore, and Have Fun Coding with AI Agents!