A REPORT
ON

PLACE RECOMMENDATION SYSTEM USING CREWAI AND OPEN ROUTER

Submitted by,

Yadhunandhan R - 20211CIT0132

Under the guidance of,

Mrs. STERLIN MINISH T N
Assistant Professor
in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING, INTERNET OF THINGS

At

GAIN MORE KNOWLEDGE
REACH GREATER HEIGHTS

PRESIDENCY UNIVERSITY

BENGALURU

MAY 2025

PRESIDENCY UNIVERSITY

PRESIDENCY SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

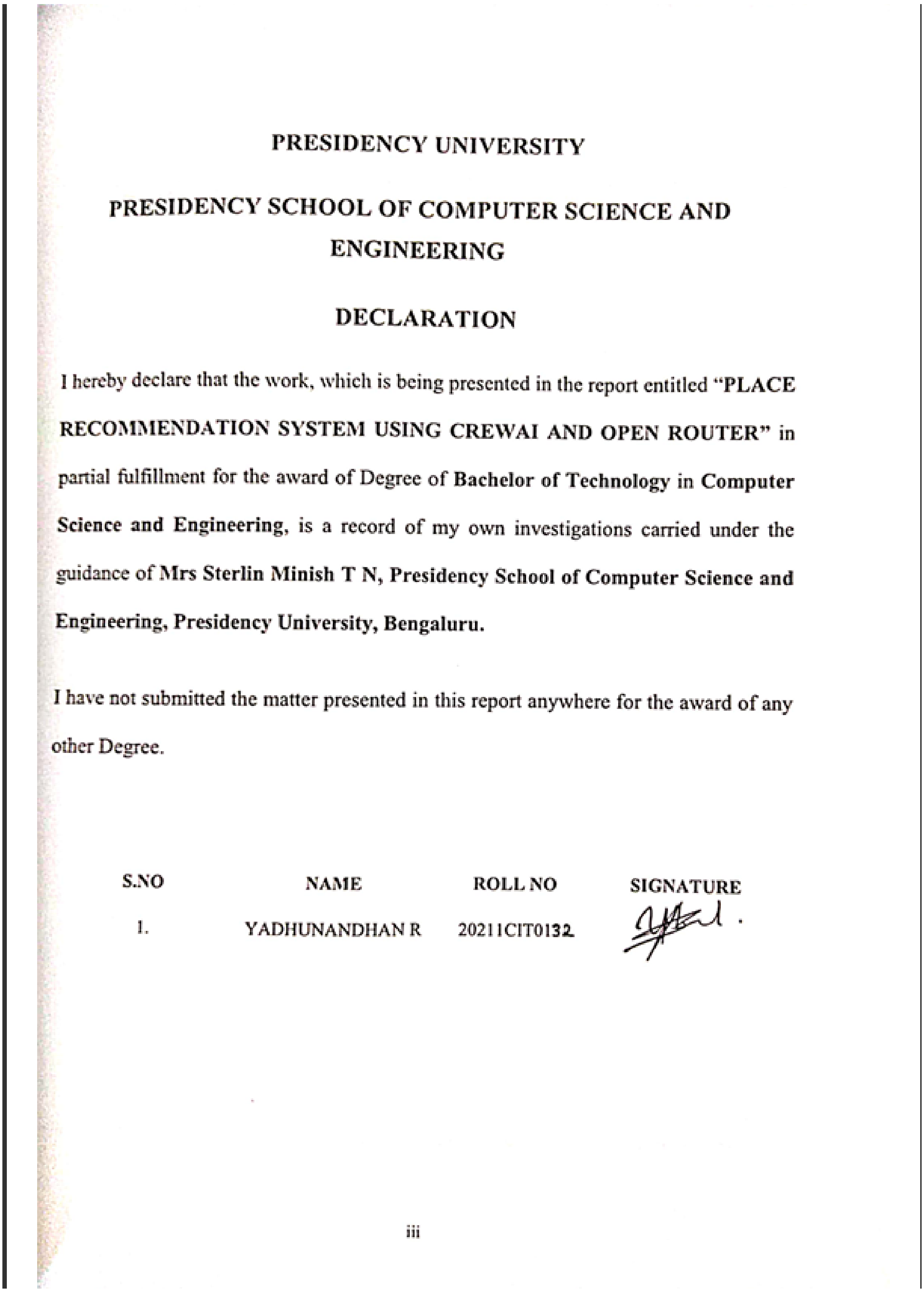
This is to certify that the Internship/Project report **"PLACE RECOMMENDATION SYSTEM USING CREWAI AND OPEN ROUTER"** being submitted by "YADHUNANDHAN R" bearing roll number "20211CIT0132" in partial fulfillment of the requirement for the award of the degree of Bachelor of Technology in Computer Science and Engineering is a bonafide work carried out under my supervision.

[illegible] 14/05/2025
Mrs. STERLIN MINISH,
Assistant Professor,
School of CSE,
Presidency University.

[illegible] 14/5/2025
Dr. S P Anandaraj
Professor & HoD
PSCS
Presidency University

[illegible]
Dr. MYDHILI NAIR
Associate Dean
PSCS
Presidency University

[illegible]
Dr. SAMEERUDDIN KHAN
Pro-Vice Chancellor - Engineering
Dean –PSCS / PSIS
Presidency University



PRESIDENCY UNIVERSITY

PRESIDENCY SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

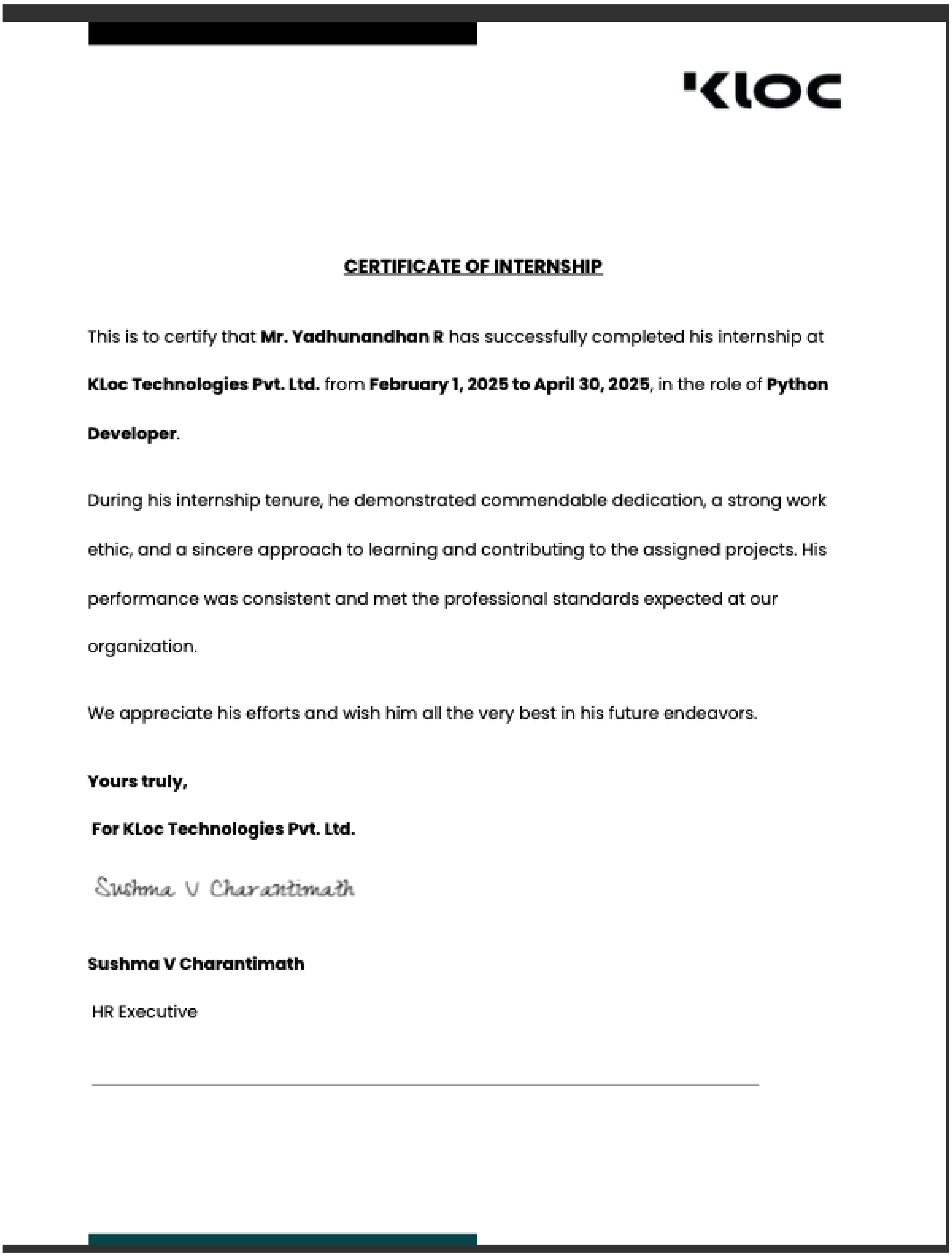
DECLARATION

I hereby declare that the work, which is being presented in the report entitled "**PLACE RECOMMENDATION SYSTEM USING CREWAI AND OPEN ROUTER**" in partial fulfillment for the award of Degree of **Bachelor of Technology in Computer Science and Engineering**, is a record of my own investigations carried under the guidance of **Mrs Sterlin Minish T N, Presidency School of Computer Science and Engineering, Presidency University, Bengaluru.**

I have not submitted the matter presented in this report anywhere for the award of any other Degree.

S.NO	NAME	ROLL NO	SIGNATURE
1.	YADHUNANDHAN R	20211CIT0132	[illegible]

INTERNSHIP COMPLETION CERTIFICATE



KLOC

CERTIFICATE OF INTERNSHIP

This is to certify that **Mr. Yadhunandhan R** has successfully completed his internship at **KLoc Technologies Pvt. Ltd.** from **February 1, 2025 to April 30, 2025**, in the role of **Python Developer**.

During his internship tenure, he demonstrated commendable dedication, a strong work ethic, and a sincere approach to learning and contributing to the assigned projects. His performance was consistent and met the professional standards expected at our organization.

We appreciate his efforts and wish him all the very best in his future endeavors.

Yours truly,

For KLoc Technologies Pvt. Ltd.

Sushma V Charantimath

Sushma V Charantimath

HR Executive

ABSTRACT

This project introduces an intelligent Travel Recommender System that generates personalized, real-time travel recommendations through LLMs, multi-agent orchestration, and web scraping. This system can interpret natural language user queries, extract group intent, and generate recommendations based on the user's intricate travel preferences.

The system is based on a multi-agent architecture scripted through the CrewAI framework based agent manager. There are two essential agents that participate in the pipeline; the Travel Analyzer Agent, that can take user input and extract structured parameters like location, ambiance, and venue type, and the Place Recommender Agent, that uses the parameters to then retrieve real-time web information using a custom scraping module. The modular agent pipeline allows for separation of tasks and concurrent execution.

The architecture supports prompt engineering and tool chaining performed through LangChain so that agents may use external tools, like scrape_website() allowing agents to directly access the web while performing reasoning. OpenRouter routes query requests to a LLM engine, like GPT-3.5-turbo, that is used for semantic parsing of user input and producing fluent descriptive responses that resemble human responses.
The robust dynamic scraper starts from a travel blog and directory site, gathering live information from real-time travel bloggers and aggregators and returning recommendations for places to visit that will reflect the latest trends and availability. Each recommendation is provided with place name, type, description, rationale, and GPS coordinates.By combining language understanding, agent-based abstractions, and live information from the web, this system offers a flexible and intelligent pipeline for modern travel recommender application.

ACKNOWLEDGEMENT

First of all, I am indebted to the **GOD ALMIGHTY** for giving me an opportunity to excel in my efforts to complete this project on time.

I express my sincere thanks to our respected dean **Dr. Md. Sameeruddin Khan**, Pro-VC - Engineering and Dean, Presidency School of Computer Science and Engineering & Presidency School of Information Science, Presidency University for getting me permission to undergo the internship.

I express my heartfelt gratitude to our beloved Associate Dean **Dr. Mydhili Nair,** Presidency School of Computer Science and Engineering, Presidency University, and Dr. S P Anandaraj, Head of the Department, Presidency School of Computer Science and Engineering, Presidency University, for rendering timely help in completing this internship successfully.

I am greatly indebted to my guide **Mrs Sterlin Minish T N, Associate Prof.** and Reviewer **Dr. Anandraj S P, Head of the Department**, Presidency School of Computer Science and Engineering, Presidency University for her/his inspirational guidance, and valuable suggestions and for providing me a chance to express my technical capabilities in every respect for the completion of the internship work.

I would like to convey my gratitude and heartfelt thanks to the PIP4004 Internship/University Project Coordinator **Mr. Md Ziaur Rahman and Dr. Sampath A K,** department Project Coordinators **Dr. Sharmasth Vali Y** and Git hub coordinator **Mr. Muthuraj.**

I thank my family and friends for the strong support and inspiration they have provided me in bringing out this project.

Yadhunandhan R

LIST OF FIGURES

TABLE OF CONTENTS

Chapter 1
INTRODUCTION

1.1 Overview

The travel and tourism sector has experienced a remarkable makeover with the advent of contemporary technology, especially in the field of artificial intelligence (AI) and natural language processing (NLP). Travelers now look for customized experiences in accordance with their individual tastes, objectives, and sentiments. In contrast to conventional planning tools that are based on static filters and fixed categories, the new traveler has come to expect systems that recognize subtle requests, make customized recommendations, and respond dynamically to shifting information and user intentions. This change in user expectation has provided the impetus for the creation of intelligent systems that not only can understand but also respond coherently to human questions. The project described herein launches an avant-garde Travel Recommendation System using CrewAI, LangChain, and OpenRouter. These components work in combination to provide real-time, context-sensitive travel recommendations based on processing natural language queries and retrieving dynamic information from the web. What comes out of it is an extremely flexible, intuitive system capable of making sensible decisions based on advanced and nuanced user needs.

At the center of this system is CrewAI, a multi-agent orchestration system that allows delegation and collaboration among autonomous agents. Every agent serves a particular purpose, ranging from analyzing user input to extracting real-time data and producing final recommendations. LangChain is central to building structured prompts and handling the logic and data flow between agents. OpenRouter gives users access to sophisticated language models, like GPT-3.5, to ensure that queries are understood with a profound knowledge of language and context.

The project's aim is not just to recommend locations but to do so in a way that feels natural, human, and compassionate. For example, when a user utters, "I want to go on a romantic date with my girlfriend in Bangalore," the system does not simply search for arbitrary cafes. It knows the underlying sentiment, the nature of experience being desired, and the contextual preferences. It then retrieves current information—such as popular date spots, user ratings, ambiance information—and creates recommendations based on that.

This smart, agent-based design allows for the development of a fully personalized recommendation engine. It also supports extensibility, so more agents can be introduced in the future to perform tasks like itinerary planning, budget calculation, or support for multiple languages. The modularity and reusability of this system make it a rich template for different recommendation engines other than travel.

In short, presents the motivation behind creating a new generation of travel assistant—a one that incorporates state-of-the-art AI tools, speaks natural language, and offers dynamic, real-time recommendations. The rest of this chapter will discuss in greater detail the history of such systems, the issues they solve, the philosophy behind the design of CrewAI and LangChain, and the practical uses and advantages of web scraping in AI systems.

1.2 Background and Motivation

Planning a trip, which used to be an uncomplicated activity of selecting a place and making travel arrangements, is now a multi-variable decision process. Contemporary travelers take into account a vast number of factors including weather, temperament, companions (single, pair, group), finances, time of year, popularity, and individual interest. Although the web has made abundant information readily available, too many options frequently make users hesitant and unable to reach satisfactory conclusions.

Classic recommendation websites like TripAdvisor or Google Maps heavily depend on filters and static information. These might provide suggestions in terms of keywords or user rankings, but they cannot pick up the fine points of human language and emotional intention. A question like "Find me a quiet place to relax with a book this weekend" is hard for such systems to interpret well. It involves not only data searching, but semantic comprehension, contextual meaning, and real-time processing of options available.

This is where the combination of LLMs (Large Language Models) and multi-agent systems presents a major breakthrough. These models learn on enormous data pools and can recognize user sentiment, infer implicit information, and make educated assumptions. When combined with autonomous agents that are adept at certain functions, these models are capable of dynamically addressing various user requests.

The motivation for this project stems from the real-world need for more human-like digital assistants—ones that can engage in intelligent conversations, reason based on the user's intent, and provide dynamic results. Consider a user traveling with elderly parents who says, “We’d like to visit temples and gardens, but not walk too much.” A traditional filter system will not account for mobility concerns, but a contextual AI can prioritize places with easy access, transportation facilities, and short walking distances. In this context, CrewAI and LangChain are the strong instruments to organize intelligent workflows. CrewAI manages the delegation of responsibilities, and LangChain manages the logic. With OpenRouter's LLM access, the system becomes able to make sophisticated decisions. Web scraping software adds to this by pulling live data, making recommendations timely and accurate

1.3 Role of CrewAI and LangChain

CrewAI is a robust framework designed to allow the orchestration of multiple artificial intelligence agents. By creating multi-agent systems that can independently perform tasks and then collaborate or delegate to one another as needed, CrewAI represents a step forward in AI’s ability to handle complex, real-world tasks. Differing from monolithic AI entities that function in a unified single entity, CrewAI supports distributed cognition, or the ability to have several agents specialized in tackling a particular subtask while realizing the overall objective of the system. For this project, CrewAI plays a critical role as it allows developing a multi-agent environment in which every agent is assigned a specific role so that the system runs smoothly.

The Travel Analyzer agent, for example, is tasked with extracting essential information from user input, whereas the Place Recommender agent is tasked with delivering appropriate travel recommendations based on the analyzed user preferences.
The architecture is sequential in nature, meaning that one agent can take turns in finishing a task. This is important when constructing systems that need to be context-aware, as one agent might have to wait for the result of another before it can continue with its task. For the Travel

Recommender System, the sequence is as follows:

1.Travel Analyzer initially analyzes the user input to obtain critical context.

2.Place Recommender thereafter utilizes this context to obtain precise recommendations according to user choices.

The LangChain platform, in contrast, is a top-level interface for orchestrating interactions between external systems and language models. It is an intermediary that makes it possible to automate tasks by stringing together multiple LLM prompts and external tools (like APIs or databases). LangChain's strength comes from its ability to integrate these disparate pieces of software together easily so that intricate workflows can be constructed around LLMs. In this project, LangChain is employed to construct prompts for both the Travel Analyzer and the Place Recommender agents. Information flow between tasks is coordinated through LangChain so that appropriate data are exchanged between agents in a structured manner. LangChain is also responsible for prompting the LLM model (GPT-3.5 or equivalent) to generate appropriate responses and responses based on the user's context, as well as guiding how agents handle external data, such as web scraping results.

By combining CrewAI's agent-based approach with LangChain's tools and memory management, the system provides an ideal environment for building sophisticated, context-aware AI systems that can work in real-time and interact with users naturally

1.4 Relevance of Web Scraping

Web scraping is a method of pulling information from websites, typically by parsing out the HTML of a page and gathering the information pertinent to the task at hand. The significance of using this practice in the Travel Recommender System cannot be overestimated. Whereas most recommendation systems utilize static data pre-defined and potentially outdated in the near future, web scraping guarantees that the system utilizes the most recent and dynamic available information from real-world sources to make recommendations.

Within the context of this project, web scraping is used for two primary functions:

- Data Acquisition: It enables the system to extract live information regarding tourist attractions, restaurants, cafes, and other places users are likely to want to visit. This is vital for ensuring recommendations stay current and valid since travel trends and places' popularity can shift very quickly.

- Improved Quality of Recommendations: By retrieving extra information from multiple online resources, the Place Recommender agent is able to create more precise and detailed recommendations supported by current, real-world data. For example, when a user seeks a place for a romantic evening, the agent is able to bring up data about highly-rated romantic restaurants or cafes in a given city and display them with proper ratings, location information, reviews, etc.

The web scraper that has been used in the project is based on the BeautifulSoup Python library to parse and retrieve meaningful text from HTML documents. The tool fetches data from given URLs, cleanses the raw HTML to obtain the readable material, and then forwards it to the agents for further processing. For instance, if the Place Recommender agent must suggest a cafe in Bangalore for a romantic evening, it can scrape lists of popular cafes from websites that are dedicated to local guides, like travel blogs or review sites. The tool scrapes the site content, extracts information like names, descriptions, ratings, and coordinates, and then sends this information back to the agent to display.

In addition, scraping enables the system to dynamically adjust to the latest available data. For example, if a new coffee shop has opened up in a trendy area, or if an existing one has shut down, the Place Recommender will adjust its suggestions automatically according to these new developments. This makes the system much more responsive than the old recommendation systems that relied on static or hand-curated data.

1.5 Use Case Highlight

One of the strengths of this system is that it can process sophisticated, natural language user inputs and convert them into effective, data-supported recommendations. As an example, let's have a look at the user input:

"I want to go on a date with my girlfriend, recommend me some nice cafes."

The intention here is to first recognize the intent behind this message and then pull out relevant details, such as:

- User intent: The query is for a romantic date, presumably on a date.

- Preferred location: The fact that "cafes" are mentioned shows that the user would like to go to a place with a casual, cozy environment.

- Additional context: The fact that "girlfriend" is mentioned introduces a romantic factor, which further limits the type of café or place to propose.

The initial processing of this query is carried out by the Travel Analyzer agent. It decomposes the input into organized data and extracts important details, including the desired venue (cafes) and the emotional setting (a date). This analysis may involve additional refinement, such as the identification of the city where the user wants to have the date (if mentioned). The agent may further deduce that the user is looking for a quiet, warm spot with positive ratings, given the wording of the request.

Once the input has been parsed, it's passed on to the Place Recommender agent. With the well-structured output from the Travel Analyzer, the Place Recommender browses through dynamic, real-time information retrieved from online guides and directories to suggest the most suitable cafes in the given area (in this example, Bangalore).

This process, in addition to suggesting what the user may want, makes the suggestions to the user extremely personalized. Not only is a list of cafes given to the user, but also a reason why exactly those particular places would be best suited for their evening out, derived from the interpretation of their request and current data by the agent.

1.6 Key Features of the Project

This project has a list of distinctive features that distinguish it from the conventional recommendation systems. Each of these features is aimed at guaranteeing the system's adaptability, precision, and efficiency in practical applications.

- Multi-agent Architecture Using CrewAI: The core of the system's design is its multi-agent architecture. By having specific agents for unique purposes — for instance, analyzing the input and suggesting locations — the system is better able to deal with multifaceted requests in a streamlined and efficient way. Each agent serves a set function, functioning autonomously and collectively. Such modularity is a strong enabler of scale-out and new-feature adaptation of the system, be it rolling it out into new cities or adding more advanced recommendation criteria.

- Integration with LangChain and OpenRouter API: LangChain's capacity to direct intricate workflows through LLMs is central to the administration of the system's processes. It allows dynamic chaining of tools and prompts so that the agents get the data they require in real-time. The OpenRouter API gives access to robust LLMs, which handle the natural language input and provide human-like output. This configuration makes sure that the system can interpret and answer queries with high accuracy and coherence.

- Tool-Based Web Scraping:The addition of dynamic web scraping renders the system extremely flexible. Through the retrieval of current information from mainstream internet sites, the system is continually updated with new information regarding places and locations so that the recommendations are fresh and relevant.

- Modular and Reusable Pipeline The pipeline developed with CrewAI, LangChain, and web scraping is modular, as every piece of the puzzle can be updated or replaced individually without impacting the workflow as a whole. Modularity enables constant refinement, whether that's through enhancing agent behavior, adding new tools, or optimizing data scraping methodologies.

- Real-Time Location-Aware Recommendations: The inclusion of location data enables the system to provide recommendations that are geographically contextual and specific to the user's desired location. This is especially helpful in city environments, where there are many choices available, and location-sensitive recommendations can greatly enhance the user experience.

- Structured JSON-like Responses: The system generates its suggestions in well-formatted outputs, like JSON, so that every suggestion is thorough and well-structured. The well-structured output provides important information, like names, descriptions, ratings, coordinates, and a concise explanation of why every location matches the user's desire.

Chapter 2

LITERATURE SURVEY

[1] Richard Hrankai and Barry Mak., "Bridging the Affordance-Actualization Gap in User Preferences for AI-Assisted Trip Planning"., March13, 2025, DOI: https://doi.org/10.1177/00472875251322518

The study is based on the affordance-actualization theory, which offers a framework for understanding the way users notice and make use of the potential presented by technological devices. Here, affordances are the functionalities and attributes of AI trip-planners users feel will be helpful or usable. Yet actualization of such affordances in use relies on the user's situation, expectations, and contentment with the site's performance. One of the most significant implications of the study is that although users in general welcome the convenience and pace of travel aids powered by AI, there is a significant gap between what they can do and what they actually do. Users consistently mention problems such as low information accuracy, absence of personalization in suggesting itineraries, and challenges in modifying recommendations against fluctuating user tastes or scenarios. These difficulties diminish the perceived trustworthiness of the system, eventually influencing long-term use and user loyalty.

The contributions of this study are twofold. First, it provides practical guidelines for developers and designers of AI trip-planning systems to develop more user-centered and context-aware systems. Through the understanding of the different needs of diverse groups of users, developers can adjust features that will improve usability and satisfaction. Second, it enhances the overall academic knowledge of AI adoption behavior, particularly in consumer-facing digital services.

[2] Aditi Singh, Abul Ehtesham, Saket Kumar, Tala Talaei Khoei., "Enhancing AI Systems with Agentic Workflows Patterns in Large Language Model., May 2024., DOI: 10.1109/AIIoT61789.2024.10578990

This paper explores the new paradigm of agentic workflows within Large Language Models (LLMs) as a departure from the traditional, linear form of interaction between users and AI systems. LLMs have traditionally been used in simple question-and-answer formats or as fixed tools for producing content in response to user input. The paper contends that such interactions, though beneficial, do not leverage the potential of LLMs to its full extent. Compared to this, agentic workflows adopt a more dynamic, interactive, and iterative paradigm, in which the LLM functions not just as an instrument but also as an intelligent agent that is capable of autonomous decision-making and evolving with changing tasks and user needs.

These design patterns create a more resilient and context-sensitive AI system that can improve, learn, and reason over time. With the incorporation of such capabilities, agentic workflows allow LLMs to operate in a style more similar to human problem-solving behaviors. The paper argues that such workflows are a milestone towards the actualization of Artificial General Intelligence (AGI), as they bestow LLMs with higher autonomy, versatility, and intellectual depth—preeminent traits required for generalized, human-level intelligence.

[3] Nilesh Borole , Dillip Rout , Nidhi Goel , P. Vedagiri Dr. , Tom V. Mathew ., "Multimodal Public Transit Trip Planner with Real-time Transit Data"., December 2013 .

A trip planner is an intelligent travel support system that has been developed to offer travelers detailed pre-trip data depending on their chosen destination and origin. It can also act as a useful tool for commuters by assisting them in planning their travel itineraries through information like road links and vehicle timetables received from different public transportation agencies. Historically, such schedules have been fixed and are typically prone to inaccuracy because of unforeseen real-world events like traffic jams, longer dwell at stops, road accidents, or unexpected interruptions. Because of that, these fixed-schedule planners often are unable to capture the current state of the transit system.

To overcome this problem, real-time trip planners have been created that make use of dynamic information from GPS-equipped vehicles. These systems are designed to improve the accuracy of travel plans using real-time data from the location of vehicles. Various issues, however, still impede the efficiency of such systems. For one, real-time vehicle positioning is usually based on third-party services, and GPS signal loss is frequent, particularly in urban areas with high-rise buildings or underpasses. Further, most real-time trip planners do not incorporate arterial roads, which can have a major influence on travel efficiency and user experience.

[4] Nhat-Tung Le, Lieu Thị Nguyen, Nguyễn Anh Tuấn, Từ Nhật Phương., "Developing an intelligent travel recommendation application utilizing ChatGPT on mobile devices"., March 2024.

This project presents the creation of an AI-enhanced travel assistance app for the next generation, with the goal of boosting the overall travel experience of users using state-of-the-art artificial intelligence technologies. The app is developed by a combination of sophisticated conversational AI architectures and platforms, specifically ChatGPT, LangChain, and Dialogflow, which make it possible for it to provide natural, interactive, and smart travel assistance. The use of the application on mobile devices guarantees availability and ease of use to customers as they move around.

One of the core objectives of the application is to recommend personalized itinerary planning and filtered recommendations for tourist attractions, based on the likes and requirements of specific users. Utilizing natural language processing (NLP) and machine learning, the system provides users with smooth, human-like conversations to get to know users' plans, interests, and limitations, then generating user-specific travel plans in real time.
Apart from travel itineraries, the app also includes a series of supporting features aimed to further enhance the traveling experience. These include live weather, tour spot search near the user, and voiceover narration, which provides an accessible experience and hands-free functionality for travelers. Such implementations make the app a travel guide encompassing more than mere recommendation engines by being contextually and dynamically interactive with users.

[5] Sai Mohith S, HemaMalini B H, Karthik .K, Nithin Reddy P N., "A Review Paper On AI-Driven Travel Planning"., January 2025.

This research delves into the role of Artificial Intelligence (AI) in transforming the conventional travel planning process. Once marked by fragmented procedures and slow manual coordination, travel planning has become a more streamlined, customized, and dynamic process, empowered by the strengths of automated recommendation technologies and real-time data fusion. Through the integration of advanced AI technologies like OpenAI APIs, Google's Gemini, and other generative models, combined with machine learning (ML) and natural language processing (NLP) methods, current systems are now able to provide highly customized itineraries that adjust to individual and group preferences. These AI-based methods not only improve the overall user experience but also facilitate improved cost control and more efficient travel coordination. One of the main shortfalls of conventional travel systems has been the absence of good collaborative tools, particularly in group travel situations where expenses were to be shared and synchronized in real time. Furthermore, they hardly included sustainability-oriented traveling options. By contrast, AI-driven solutions are now able to fill these voids by facilitating real-time collaboration between travelers, smart budgeting, and offering green suggestions that encourage sustainable tourism practices.

The review therefore identifies both the strengths and present limitations of AI-powered travel planning systems. It underscores the need for continued research to solve existing problems and to move closer to a comprehensive, user-focused platform that not only automates and personalizes travel planning but also keeps pace with the evolving expectations and environmental consciousness of today's international traveler.

[6] Patalee De Silva., “AI-Powered Travel Planner: A Smart Solution for Personalized and Efficient Travel Itineraries. 2. Research Area Specific Area Artificial Intelligence (AI) , Software Engineering , Tourism Technology Personalization in Travel Planning , Efficiency in Travel Itinerary Generation , Integration of AI and Tourism , User-Centric Design , Data-Driven Decision Making 3. Supervisor(s)”., March2025.

This project aims at creating an AI-based travel planner that is meant to provide extremely personalized and effective travel itineraries based on multiple user inputs such as preferences, budget limitations, and dates of travel. The fundamental aim is to simplify the travel planning experience by utilizing the strength of superior AI models, such as Gemini for natural language understanding and conversation and Firebase for strong user authentication and real-time data handling. The technologies synergize to present an intuitive and intelligent user interface.

Individuals who use the app can develop personal profiles that retain necessary details and preferences guiding trip planning. Once a trip is initiated, the system generates curated recommendations that cover all essential aspects of travel—such as flight options, accommodation suggestions, and popular tourist attractions—all tailored to meet the user's specific requirements. Through AI’s data analysis and learning capabilities, the system can continuously improve its recommendations over time, making each travel experience more refined than the last.

[7] A Naga Jyothi1, Giresh Raju Adimulam, Chandu Neelam, Janni Narasimha Gowud VNL , Raj Kumar Unnamatla , Karthikeya Tumpati., “Travel Finder – An AI-powered Travel Planning and Recommendation System”., 4 April 2025.

The AI-Powered Travel Finder is a major leap forward in the travel technology space, with a sole purpose of turning the solo traveler experience into a socially interactive and collaborative experience. This new platform harnesses the capabilities of Artificial Intelligence (AI) and Machine Learning (ML) to analyze different user traits, including favorite destinations, dates of travel, budget constraints, and interests. Following this analysis, the system gives highly personalized recommendations of compatible travel buddies, allowing single travelers to match with others sharing similar tastes.

In the core of the system are a collection of smart matching algorithms such as collaborative filtering and clustering methodologies, which create travel groups or integrate users dynamically into preexisting ones. These algorithms promote group coherence through the recognition of shared interests and similar travel objectives, thus optimizing group dynamics and trip happiness. By doing so, the site overcomes one of the most prevalent solo travel disadvantages—isolation—by creating a rich atmosphere of community and shared experience. In addition, the platform prioritizes cultural exchange and social engagement, augmenting the affective aspects of travel. Be it for recreational, adventurous, or business travel, the AI-Powered Travel Finder improves both the utilitarian realm of trip planning as well as the social connections developed while traveling. Not only does it simplify the logistics of meeting with others and planning trips, but it also redefines solo travel by being more inclusive, interactive, and affectively enriching.

[8] Akshat Singh, Raksha Madhogaria, Abhishek Misra, E. Elakiya., "Automated Travel Planning Via Multi-Agent Systems and Real Time Intelligence"., 9 Jan 2025.

Modern travelers often face a critical challenge in trip planning—navigating through an overwhelming number of travel websites to collect accurate and personalized information. Traditional travel planning methods are time-consuming, lack real-time accuracy, and often fail to deliver tailored recommendations efficiently. Addressing this need, the paper introduces Travel Optix, a sophisticated multi-agent system designed to automate and streamline the travel planning process by generating personalized itineraries in under five minutes.

In performance evaluations, Travel Optix has shown significant advantages over leading Large Language Models (LLMs) such as ChatGPT-4, Google Gemini, and Ollama. Leveraging advanced prompt engineering techniques, the system delivers highly structured, context-aware, and real-time responses, outperforming others in both efficiency and precision. A notable feature is its ability to allow users to filter and compare travel destinations using up-to-the-minute data, making it easier to refine preferences and make informed decisions.
The paper not only presents the system's methodology and outcomes but also discusses future enhancements. Planned upgrades include the integration of image recognition and even more advanced LLMs, aiming to further enrich the travel planning experience. With its innovative agent-based framework and real-time capabilities, Travel Optix is positioned to redefine the standard in digital travel planning, offering users a seamless, personalized, and intelligent trip-planning solution.

[9] Xiao, Ziren., "Artificial Intelligent-Based Multi-Agent Collaborative Path Planning Methods".,May 2024.

Artificial intelligence-based navigation for multi-user interaction is gaining wider recognition in research and industrial applications, providing greater resource planning and congestion relief functions. AI-powered navigation systems adaptively plan routes for multiple users, maximizing transport efficiency while reducing energy usage and travel time—thus enhancing personal experiences and making infrastructure sustainable.

One such hard and less-explored issue is Dynamic Multi-Sources to Single Destination (DMS-SD) navigation. In contrast to conventional global path planning with static destinations, global obstacle avoidance, and personal preferences, DMS-SD encompasses two special issues: (i) a dynamic destination that can change in accordance with real-time preferences and user positions, and (ii) multi-agent collaborative decision-making for achieving a temporary, shared endpoint efficiently and equitably.In response to these challenges, three AI-based solutions were suggested. The first combines Monte Carlo Tree Search (MCTS) with Ant Colony Optimization (ACO), minimizing ACO's randomization and allowing for quicker decision-making with partial map knowledge. While effective (20× faster than enumeration-based approaches), it only applies to small-scale environments.

[10] Aili Chen ,Xuyang Ge, Ziquan Fu ,Yanghua Xiao., "TravelAgent: An AI Assistant for Personalized Travel Planning"., September 2024., DOI:10.48550/arXiv.2409.08069 .

With increasing global tourism and advancing artificial intelligence (AI) technologies, smart travel planning has become one of the centerpieces for research. In actual travel contexts, planning mechanisms need to incorporate multi-dimensional considerations like time, tastes, cost, and availability. Optimal itinerary creation then needs to satisfy three essential requirements: rationality (sequencing and feasibility), comprehensiveness (coverage of applicable activities and arrangements), and customization (adherence to user-determined requirements). Still, conventional rule-based systems and even certain Large Language Model (LLM)-based methods tend to fail to meet all three of these at the same time.

To overcome such limitations, TravelAgent has been suggested—a next-generation travel planning system fueled by LLMs, specifically geared to produce dynamic, personalized itineraries that evolve according to user contexts and preferences. TravelAgent is organized into four functional modules: (1) Tool-usage Module, which incorporates third-party tools and APIs for real-time data collection; (2) Recommendation Module, which recommends destinations and activities according to user interests; (3) Planning Module, which generates coherent itineraries; and (4) Memory Module to maintains past preferences and behaviors .

[11] Mrs.Apeksha Shirke, Ms. Archana Sahani, Ms. Nandini Soni., "PERSONALIZED TRAVEL ITINERARY USING AI"., February 2025.

This paper aims to revolutionize conventional trip planning by leveraging artificial intelligence to create highly personalized travel experiences. Unlike traditional methods that often require manual research or costly assistance from travel agencies, this system empowers users to independently plan trips tailored to their unique preferences, budget, travel dates, and destination-specific factors.

By integrating crucial elements such as estimated travel time, weather conditions, local attractions, and optimal visiting hours, the AI algorithm constructs efficient and enjoyable itineraries that enhance exploration while minimizing planning effort. This not only saves users valuable time but also ensures a seamless and memorable travel experience.
The system's personalized recommendations are crafted based on a deep understanding of user inputs, delivering itineraries that align with individual needs and constraints. Users benefit from an intuitive and intelligent platform that reduces dependency on third-party travel agents and promotes self-guided travel. This paper reflects the growing potential of AI in transforming the tourism industry, showcasing how intelligent systems can automate complex planning tasks with accuracy, efficiency, and personalization. As travel continues to evolve in the digital age, AI-driven solutions like this serve as a practical tool for modern travelers seeking convenience, flexibility, and control over their journeys.

[12] Meirong Lin., "Research on Development and Application of AI Agent for Travel Recommendation Driven by Large Language Model"., 07 June 2024., DOI: 10.1109/ICCECT60629.2024.10545805.

Using the structure of the large language model and the ZeLinAI framework, this work describes the building and deployment of smart travel advisory agents that maximize user experience for contemporary tourism. Such AI assistants tap into the powers of the sophisticated deep models and sentiment analysis mechanisms to sensitively capture and analyze user liking, emotional context, and environmental needs. Through the processing of user inputs—everything from text-based searches to reviews and emotional sentiment levels—the system can deduce more profound travel intentions and provide strongly personalized travel suggestions. The essence of the innovation is the integration of large language models (LLMs), which have the capacity to comprehend rich language patterns and create human-like outputs, enabling more naturalistic and interactive trip planning experiences.

The ZeLinAI platform is the underlying infrastructure for deploying these agents, providing a solid foundation that enables real-time interaction, ongoing learning, and scalable performance. To further improve the quality of recommendations, the system utilizes prompt optimization methods and incorporates custom training data sets that capture varied user behaviors, preferences, and travel contexts. This adaptive learning process ensures that the AI agents change over time, constantly refining their recommendation accuracy and contextual relevance.

In addition, sentiment analysis is used not just to pick up on users' overt likes and dislikes but also to identify underlying emotional signals, like excitement, hesitation, or dissatisfaction. This enables the system to optimize its responses and recommendations, offering travelers empathetic and contextually relevant advice. Consequently, the AI agents provide a smooth and intelligent travel planning experience that is both emotionally intelligent and data-informed. The following paper illustrates the ability to integrate LLMs, sentiment analysis, and in-house data training on tools such as ZeLinAI for building the next generation of travel assistants that are able to redefine how users plan and experience their trips.

Chapter 3

RESEARCH GAPS OF EXISTING METHODS

3.1 Limited Contextual Awareness in Conventional Travel Recommender Systems

A major flaw in conventional travel recommendation systems is their lack of ability to comprehend contextual user intent. The majority of systems are based on keyword searching or predefined-form inputs (such as choosing "beach" or "mountain" from a list), with little or no contextual awareness of conversational intent. For example, if a user utters, "I want to have a quiet dinner date around the lakeside," conventional systems do not conclude that the user probably wants a romantic atmosphere, proximity to bodies of water, and perhaps a less noisy venue — not necessarily a restaurant labeled "dinner."

This semantic ignorance is a research imperative. While Natural Language Processing (NLP) has gained traction, a significant proportion of deployed recommender systems continue to rely on strict, syntactic rules instead of semantic parsing. Systems such as Google Maps and TripAdvisor offer suggestions based on tags, user ratings, and popularity — but seldom based on the deeper emotional or experiential background of a user's query. In addition, most traditional systems do not adapt dynamically to ambiguous or abstract inputs. For instance, "I need a break from work stress" might map to a nature walk, a yoga retreat, or a quiet weekend beach vacation — based on user history and personal preference. Yet these systems are typically unable to disambiguate and match such abstract utterances to real-world choices.

This void provides a key driver for the creation of your CrewAI-driven Travel Recommender System. CrewAI agents can break down input semantically, learn travel intent, and reason over indeterminate phrasing with LLMs such as GPT-3.5. In contrast to tag- or keyword-based systems, the system reads out the intent behind a user's query — an improvement over typical recommendation methods.

3.2 Static Filtering Mechanisms Without Real-Time Data Adaptation

Another major shortcoming in current travel recommendation systems is their dependency on static filtering mechanisms. The majority of travel websites, including TripAdvisor, Yelp, and Booking.com, utilize pre-canned filters — such as "top rated," "closest," or "budget" — that sort available datasets into user-choosable bins. Although such systems provide a minimum level of usability, they are inherently unable to respond to evolving data in real time or support intricate user queries that call for dynamic content creation.

This kind of static content retrieval is even more serious in rapidly evolving urban landscapes. This delay in catching up with real-world changes and updating the database makes such platforms less useful, particularly for users looking for current and trendy locations. In addition, there are no facilities in these systems to query real-time data sources like websites, blogs, local event pages, or maps APIs dynamically. Because legacy systems depend on vendor-provided listings or user-submitted content, their coverage is restricted to what is contained in the database, typically not including specialty, local, or newly opened venues. Their functionality does not include tools such as web scraping, current Google Maps APIs, or semantic extraction from open web content.

Moreover, combining real-time data scraping with LLM reasoning allows for contextually accurate filtering. If the user says, “I’m craving something peaceful and artistic tonight,” the model can search and recommend art cafés or galleries that are open late and located in quiet neighborhoods. This creates a tailored and temporally-aware recommendation something static filters can never accomplish. This method represents a major innovation in recommendation system design. Instead of viewing travel information as rigid and categorizable, the system views it as dynamic and perpetually changing much like the real world that it represents.

3.3 Restricted Personalization and Rigid User Modeling

One of the most critical building blocks of any contemporary recommendation mechanism is personalization. Yet current travel websites and tourist recommendation schemes still rely on excessively generalized user models or stagnant historical behavior patterns to make suggestions. This causes repetitive, mass, and even useless recommendations.

Personalization efforts largely depend on collaborative filtering or content-based filtering in traditional personalization approaches. Whereas these work so well for e-commerce, their utility does not hold so well in the travel space as a result of varying contextual needs. In addition, current systems hardly ever take into account spontaneous user preferences that vary based on emotional state or intent. A person might use the app when feeling stressed, romantic, adventurous, or nostalgic — but such systems do not have mechanisms for detecting such states or converting them into appropriate recommendations

Another drawback is that there is no long-term user modeling. Travel sites have minimal memory between sessions, with most interactions being standalone. This does not establish trust or enhance the system's effectiveness over time.. The architecture can be designed to include memory modules so it can memorize, refer, and adjust user preferences along the passage of time.

3.4 Inadequate Use of Multi-Agent Collaboration for Decision-Making

Most recommendation engines are built monolithically — i.e., a single algorithm (e.g., collaborative filter or rule-based engine) implements the whole pipeline: intent extraction, scoring, and suggestion generation. This monolithic structure imposes reasoning bottlenecks, and it restricts modularity and explainability. Contrarily, AI research has established that splitting complex reasoning tasks into smaller, specialized agent roles can enhance performance as well as interpretability.

Despite these advantages, few travel systems use agent collaboration. This is a major research gap and one that your system fills through CrewAI. By defining clear agent roles:

- Travel Analyzer → Extracts travel intent, sentiment, constraints
- Place Recommender → Uses scraping tools to gather and filter places
- (Optional) Validator Agent → Confirms the recommendations match intent

This delegation simulates real-world collaboration, where specialists address various components of decision-making. Each agent employs LLMs, utilities, and systematic workflows, and they work in concert with the CrewAI sequential process.

3.5 Lack of Explainability in Recommendations

One of the most underutilized aspects of recommendation systems is explainability. The vast majority of users are given a list of recommended sites but are not informed why they were selected. Lack of transparency erodes trust and lowers user activity. This transparency derives from LLM-based reasoning, which can produce natural language explanations of its decision process.

Chapter 4
PROPOSED MOTHODOLOGY

4.1 SYSTEM OVERVIEW

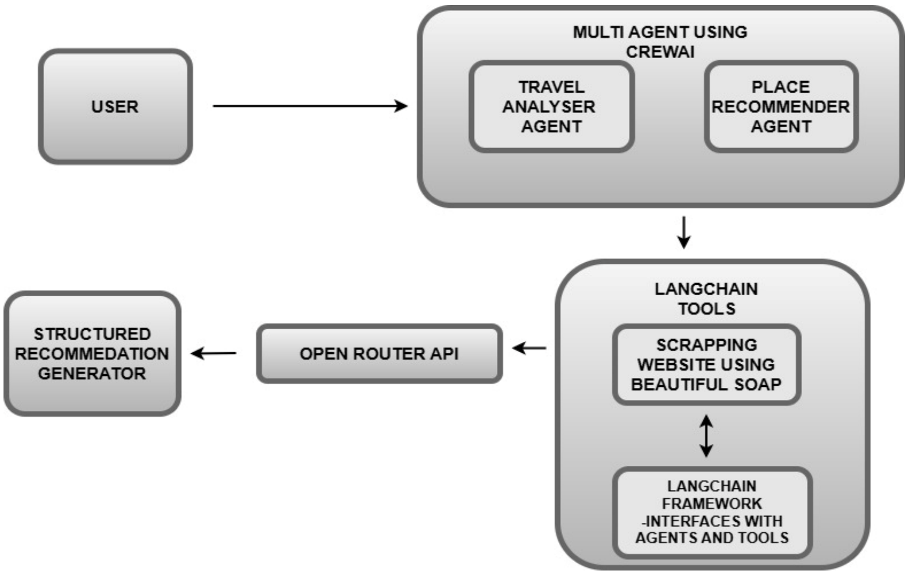

USER
MULTI AGENT USING CREWAI
TRAVEL ANALYSER AGENT
PLACE RECOMMENDER AGENT
LANGCHAIN TOOLS
SCRAPPING WEBSITE USING BEAUTIFUL SOAP
LANGCHAIN FRAMEWORK -INTERFACES WITH AGENTS AND TOOLS
OPEN ROUTER API
STRUCTURED RECOMMEDATION GENERATOR

Fig 4.1: Architecture Diagram

The architecture of the Travel Recommender System is built using a modular, agent-based system that encompasses both natural language comprehension as well as task orchestration and live data scraping. We made use of CrewAI for agent orchestration, LangChain for chaining tools/models together, and OpenRouter to provide a way to access some high performance LLMs such as GPT-3.5. The goal behind the architecture was to facilitate a smooth, intelligent recommendation pipeline that would take ambiguous user input and help interpret it into granular, personalized travel recommendations.

The system of the Travel Recommender System starts with a User Input component, which allows users to make a free flowing, natural language query such as “Give me a romantic café in Bangalore.” While the user input is a natural language interpretation, it is unstructured, ambiguous, and could be worded in many different ways. The input is then passed directly into the backend-this first component will give way to data semantic interpretation. The first module of processing is the Travel Analyzer Agent, built with

CrewAI. In essence, the main purpose of the agent is to retrieve structured data from the request user's input. It follows the intent of the user (e.g., romantic date), what type of place (e.g., café), the location (e.g., Bangalore), and any other parameters/constraints (e.g., price, ratings, atmosphere). The agent interprets the natural language query via a Large Language Model (LLM) through the OpenRouter API and distills it into structured data that later will be interpreted by the other components.

The module that receives the structured output provided by the analyzer is the Place Recommender Agent, another CrewAI agent that can make use of outside tools. The Place Recommender Agent looks to the parameters that were analyzed and gives a recommendation for which type of places to recommend and which service to gather the data from. The Place Recommender Agent uses the Web Scraper Tool to gather the data. The Web Scraper Tool is a callable tool in LangChain developed using Python's BeautifulSoup library. The web scraper tool fetches live data from trustworthy travel sites or review sites and extracts important elements, such as, place name, ambience description, rating, and geographical coordinates.
The LangChain framework serves as the middleware between agents and tools. LangChain supports prompt chaining, manages calls to tools (e.g., scraper), and helps keep context between steps. LangChain enables agents to function independently while still passing structured data in a coordinated flow between tasks.

The OpenRouter API acts as the point of entry for the language model. It connects each agent prompt with a hosted LLM (e.g., GPT-3.5) to guarantee high iterations of language understandings and language generations. OpenRouter allows flexibility in model choices while guaranteeing fast and scalable inference. At the stage of Received and Processing of the scraped data, the Place Recommender Agent creates a Structured Recommendation Output using the LLM. The Structured Output includes the names for each recommended place, their type, a contextual description, a rating/popularity of each recommendation, and the source coordinates. The Structured Recommendations are then sent through the Output Formatter to provide clarity and consistency.

4.2 MULTI- AGENTS USING CREWAI

The Travel Recommender System is based on an agent-based architecture, in which multiple AI agents collaborate to receive user inputs, analyze tastes, and provide travel destination recommendations. The agents are developed on the CrewAI framework, which coordinates multiple independent agents acting in concert. The use of a multi-agent system strategy guarantees that every agent will play a unique role and be able to perform task delegation to other agents, enabling enhanced decision-making, task delegation, and teamwork.

The three main Multi Agents are:

- Travel Analyzer: Processes the user's input, determining principal preferences and restrictions like place type (e.g., café, tourist location), location, mood (romantic, adventurous), and specific conditions (e.g., distance, price, ratings).
- Place Recommender: After the preferences are derived, the Place Recommender agent makes use of outside resources such as web scraping tools to suggest appropriate places that match the user's context. They are chosen based on relevance, rating, location, and availability of required features (e.g., romantic atmosphere, family-oriented).
- Scraping Tool: A self-built tool using BeautifulSoup for scraping web page data. This enables the recommender agent to extract real-time data from web pages to make its recommendations more timely and precise.

The system leverages LangChain to connect to large language models (LLMs) like OpenRouter and GPT models for contextual interpretation and generation. OpenRouter acts as the API gateway for these models, providing effortless interaction between system components.

4.3 Step-by-Step Workflow

The workflow of the Recommender System for Travel is efficient, dynamic, and interactive. The following is a step-by-step description of the process.

Step 1: User Input Reception

The initial step in the process is to receive input from the user. The user is supposed to input a natural language query, for instance:

"I would like to have a romantic date with my partner in Bangalore. Can you recommend a good café?"

Step 2: Travel Analysis by the Analyzer Agent

The Travel Analyzer agent analyzes the user's query. The agent is designed to extract structured data from unstructured user input. The analysis is based on the following main components:

- Type of Location: Determining if the user wants a café, park, restaurant, etc.
- Mood or Intent: Identifying whether the user is searching for a romantic, adventurous, or family-oriented location.
- Location: Identifying geographical preferences such as a city or a specific locality.
- Additional Preferences: This can include items like budget, ratings, distance from the current location, etc.

The objective here is to reduce the user input into executable things that the following agent can use to filter and suggest proper locations.

Step 3: Task Delegation and Web Scraping

Once the analysis process is completed, the Place Recommender agent is activated. The recommender agent, according to the user's identified preferences, is tasked with finding a list of places suitable for the needs. For instance, if the user says "romantic café in Bangalore," the recommender agent will search for cafes in Bangalore that are highly rated and romantic in atmosphere.

This agent employs web scraping to harvest live information for locations like:

- Cafes in Bangalore
- Ratings, reviews, and ambiance details
- Location coordinates (latitude and longitude)

The system employs BeautifulSoup, a Python framework that is targeted towards web scraping, to gather the information when it scrapes the websites. The system even verifies the latest data to ascertain the recommendations would not be in terms of aged information.

Step 4: Generating Recommendations

After collecting enough information from web scraping, the Place Recommender agent compares the possible places. It makes comparisons on relevance to the user's interests (place type, location, and mood) such that only the best places are chosen. The top 3-5 recommendations are listed, and each destination is followed by:

- Name of the destination

- Type of destination (e.g., romantic coffee shop, tourist attraction)
- Rating and reviews (if provided)
- Geographical coordinates (latitude and longitude for integration into maps)

These suggestions are then structured in a neat, formal response ready to be provided to the user.

Step 5: Empathetic Response Generation

The LangChain library is incorporated to enable the system to produce human-like and empathetic responses. Rather than providing raw suggestions, the system will react conversationally, i.e.:

"For a date in Bangalore, I would suggest Café X. It is a small café with a rating of 4.5 and located at [coordinates]. Many couples like it because it's so quiet. Do you want more recommendations or information on other places?"

The reply is aimed at making the user feel heard and engaged and creating a positive experience for the user.

4.4 Integration with LangChain and OpenRouter

The system integrates with LangChain, a powerful library that facilitates prompt engineering, memory handling, and LLM tool integration. LangChain allows the agents to interact dynamically with large language models and external APIs, providing real-time context and delivering relevant outputs based on user input.

- LangChain's Function: The system utilizes LangChain's chaining feature to transfer the processed input from the Travel Analyzer to the Place Recommender agent. LangChain ensures that these models are chained together seamlessly, and context is preserved across multiple interactions.
- OpenRouter API Integration: OpenRouter serves as an intermediary between the agents and the LLMs (e.g., GPT-3.5, Claude). The system makes requests to OpenRouter, which retrieves the response created by the model. This enables high-quality natural language understanding and generation in real time.

4.5 Web Scraping for Live Data Fetching

Web scraping becomes fundamental to this approach. Static datasets will not work since real-time data are needed to guarantee the current and relevant nature of the recommendations. The web scraping tool is designed to grab major details from open sites regarding cafes, travel destinations, and similar venues. Web Scraping Tools: The BeautifulSoup-based scraper is customized to retrieve and parse the content of a specified URL, making sure that the data is relevant and up-to-date. By extracting raw text and removing irrelevant content, the tool aggregates the most important data to ensure the recommendations are more accurate. Scraping Efficiency: The web scraping tool is designed to handle multiple queries efficiently, ensuring quick data retrieval even from large websites with extensive content. It ensures that the Place Recommender agent has access to the freshest data possible, improving the system's overall accuracy and reliability.

4.6 DATA FLOW AND INTEGRATION OF THE SYSTEM

The data flow and integration pattern of the Travel Recommender System has been designed to facilitate effective communications and coordinated task performance among AI agents, tools and language models. This modularity allows for flexibility, the ability to operate in real-time over the data, and the ability to intelligently make decisions at any point along the pipeline. The integration of LangChain, CrewAI, OpenRouter and the scraping tool forms the essential structure of the system enabling domains of user inquiries to be processed autonomously into structured, relevant travel recommendations. Below given fig:4.6.1

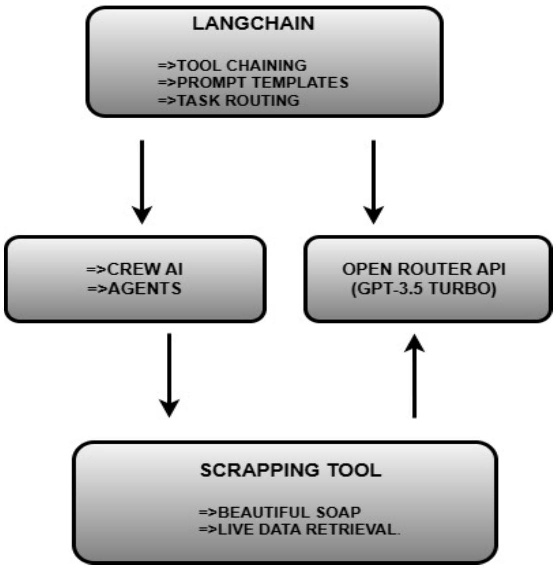


Fig:4.6.1 Data Flow and Integration of the System

On top of the data flow, the LangChain framework is the orchestrator for prompt routing, coordination of agents, and invoking tools. It serves three broad purposes: facilitating tool use by CrewAI agents, reusable prompt templates, and task routing between agents and tools. The data flow branches out from LangChain into two streams: CrewAI agents (Travel Analyzer and Place Recommender) for decision-making and reasoning and OpenRouter API for accessing language models (e.g., GPT-3.5). CrewAI agents work with established goals—Travel Analyzer derives user travel preferences from free-form input, and Place Recommender employs that information to recommend destinations. Both agents tap into tools through LangChain and ask OpenRouter for natural language understanding and response generation. OpenRouter performs model inference and feeds outputs back to the agents.For live data, the Place Recommender uses a Python-based Web Scraper Tool powered by BeautifulSoup to collect place names, user ratings, descriptions, and coordinates from travel sites. LangChain ensures contextual coherence across all components, routing inputs/outputs seamlessly.

This modular system combines LangChain's orchestration, CrewAI's intelligent agents, OpenRouter's flexible model access, and BeautifulSoup's real-time data gathering to deliver natural, up-to-date, and extendable travel recommendations.

4.7 System Deployment

After the methodology is established and the system is developed, the second step is deployment. The system will be hosted on a cloud platform that is scalable to support numerous user requests. It will be incorporated into a web frontend, where users will be able to interact with the AI agents via a conversational interface. The backend will be responsible for task orchestration, agent interactions, web scraping, and LLM calls such that the user gets personalized travel suggestions with little latency.

Chapter 5
OBJECTIVES

The broad goal of this project is to create and deploy an intelligent, real-time, personalized travel advisory system that capitalizes on the capabilities of CrewAI, LangChain, and OpenRouter to provide precise, context-sensitive suggestions to users as natural language input. The system must be able to understand open-ended and expressive human queries, derive pertinent information like intent, preferences, and constraints, and create customized travel suggestions using modular agents and live data sources.

5.1 Objective 1: Create a Natural Language Understanding System

Build the Travel Analyzer Agent, which will take natural language queries from users and parse that input into a structured representation of their travel data elements (e.g., travel purpose, venue type, location, constraints, keywords). It leverages an LLM via OpenRouter searching for common ways to understand less formal and more ambiguous cuisines (e.g., "somewhere chill"). In effect, it's looking to move unstructured or messy text into a structured output.

5.2 Objective 2: Create a Modular Multi-Agent Engine Thought Controller Using CrewAI

While the Travel Analyzer and Place Refresh Agents are relatively simple standalone interfaces, they can be tied into CrewAI's Process.sequential model allowing us to string together multiple specialized agents:

1. Travel Analyzer Agent - takes user input
2. Place Recommender Agent - produces recommendations
3. (optional) Validator Agent - checks for quality

This modular arrangement supports agents that may be not yet be considered in the future - e.g., a Budget Planner or Weather Checker can be introduced as separate implementations without having to change the logic of our engine.

5.3 Objective 3: Bridge Live Data - Provide Feedback via Web Scraping

We can get our travel data through a Python-based scraper, using BeautifulSoup and requests whenever possible, dynamically pulling relevant travel information (e.g., venue names, descriptions, ratings, coordinates, popularity) from live sources (e.g., tourism blogs, links, directories), etc. As a result, our recommendations will always be current and responsive to data meaning the recommendations of tomorrow can still work with input that came from the past week.

5.4 Objective 4: Present Output in Structure Format

Output from this agent includes the name, description, type, and coordinates, as well as rationale to map ideas, to communicate to the user and provide the opportunity for ideas to approximate into a spatial representation.

5.5 Objective 5: Ensure and Plan for Scalability and Extendability

All tools within the LLM, agent architectures, CrewAI's modular approach will use a technique known as tool chaining via LangChains and rely on the os module within Python allowing a new LLM to be become the engine when we want to add tools and wrap the future open to concepts outside of CrewAI and LangChain.

Chapter 6

SYSTEM DESIGN & IMPLEMENTATION

The Travel Recommender System was designed as a modular, agent-based system that could take user input and make real-time travel recommendations using contemporary AI frameworks. The system design includes CrewAI for multi-agent coordination, LangChain for prompt and tool chaining, and OpenRouter for high-performance LLM inference. In this chapter, we will examine the architecture design choices, agent behaviors, expedited task delegation, scraping tool integration, and execution pipeline. We have ensured the architecture is modular, scalable and can extend to future real-world deployment environments in web apps or travel assistance bots.

6.1 Architecture and Design Principles

The Travel Recommender System architecture was intended from the inception to be based on service oriented architecture and agents-oriented architecture principles. The system has loosely coupled components and agents, with each agent, task and tool reacts to specific roles in the architecture to allow for the separation of concerns, high cohesion, low coupling. The flow starts when user input enters as a free-form natural language query, e.g., "Suggest a good café to take my girlfriend on a date with in Bangalore" and moves to the CrewAI orchestrator which forwards the analysis to the Travel Analyzer Agent. This agent looks for the key attributes like type of place, purpose, user context (date, meditation, fun) and geographical hints. When the analyzer finishes its reasoning, the results are handed off to the Place Recommender Agent. This agent uses LangChain's tool calling mechanism to call a scrape_website tool to look up one or more predefined travel-related web pages of interest.

The tool will grab content that is relevant; parse the HTML, and return it as snippets of text summarised. The Recommender processes this data, filters it according to the output of the Analyzer, and produces a list of suggested travel recommendations with the place names, descriptions, explanations of suitability, and lat/long coordinates. Both agents' language models are being invoked via the OpenRouter API. The system is using langchain_openai wrapper with ChatOpenAI, which has been set up to use the OpenRouter

API base and key. These settings are being exposed via environment variables (OPENAI_API_BASE, OPENAI_API_KEY) for security and scalability of the assembly. Design for complete agency is followed here. That is, when the user input is provided, the system can perform all the steps of reasoning, acting, and output generation without needing any further interaction from the users.

6.2 Agent Engineering with CrewAI

CrewAI is a Python framework that enables the imagination, instantiation, and management of collaborative agents capable of tracking a pre-set goal/task. In the CrewAI framework, we incorporated two primary agents: Travel Analyzer Agent Place Recommender Agent. Each agent is an object instance of the Agent class and has the following elements: Role: a semantic identifier you will use in prompts (e.g. "You are a Travel Analyzer"). Goal: the aim or objective of the agent (e.g. "Analyze and derive user travel preferences"). Backstory: background behavior cue (e.g. "You are a tourism expert and good at interpreting human intent"). LLM: a configured language model (GPT-3.5-turbo via OpenRouter API). Tools: External tools an agent can use (e.g. scrape_website). Delegation Flag: allows/disallows recursive task assignment (we will not enable this project). The Travel Analyzer Agent focused purely on semantic understanding and intent disambiguation. The Travel Analyzer acts as a natural language understanding (NLU) agent, and the NLU input parameters it extracts included: Place category (café, museum, park, temple); Location or region (explicit like Bangalore, or inferred); Occasion (romantic, spiritual, solo); Constraints (budget, ambiance, family-friendly).After that, the output is organized in bulleted or JSON-like format and passed to the next agent.

The Place Recommender Agent is set up to use web tools and scrape dynamic content. The expectation is a ranked list of 3–5 places, with rationale per location, and coordinates. CrewAI's sequential mode makes sure that the input from the Analyzer is the working input for the Recommender. All the agents in CrewAI keep verbose execution logs for debugging and performance purposes. The logs include prompt messages, extracted parameters, tool calls, and final outputs.

6.3 Task Modeling and Prompt Engineering with LangChain

LangChain is the main way to manage prompts, tools, and various chains of LLMs. Each task in the CrewAI pipeline is modeled as a Task object, with a description for the agent's work and an outline of the expected output structure. LangChain's tool calling system natively allows external Python functions to be called in prompts.

For example, the Travel Analyzer’s task prompt might look like this:

User input: "I want to go to a peaceful place to read a book."

Task: Extract key preferences such as category, ambiance, location, and occasion. Return a structured output that the next agent can use to recommend places.

Expected Output:

- Place type: Library or park
- Ambiance: Quiet
- Location: Not specified (default to user’s current city)
- Notes: Suitable for solo reading

We registered a tool using the @tool decorator defining scrape_website(url: str). When it is called in the LangChain prompt, this tool makes a web scraping request and returns parsed output in real time.

Beyond basic querying function calls, the key benefits of LangChain are:

- Template-based prompting reduces variability and hallucination in output.
- Tool chaining allows interaction with live data sources.
- Memory and conversation history support (optional) allows for future multi-turn expansion.

Finally, the prompt(s) are configured using temperature, top-p, and repetition penalties to moderately weigh creativity and consistency.

6.4 Tool Integration and Web Scraping Engine

Web scraping is an integral part of this system because it allows the recommendations to have a live and updated quality. Instead of static databases, we made a custom Python scraper tool that uses:

- requests for HTTP GET requests

- BeautifulSoup from bs4 for parsing the HTML
- Error handling for HTTP status codes of 404/403/500
- HTML cleanup (i.e., removing script/style tags)

The scraping tool scrapes the visible content from targeted travel websites such as city blogs, café directories, or community portals, returns back the first 1000 characters of human-readable text, and feeds it through the LLM. This has now provided the system with real-time access to up-to-date reviews, addresses, and ambiance information which results in recommendations that are not only relevant but also actionable.

This scraping tool is created in LangChain as a tool that enables agents to call from within prompts without a direct access to the code. Given the modularity of the tool there will be future opportunities to extend the scraping tool, adding capabilities to scrape ratings, pricing, or even the ability to parse structured data like JSON-LD or OpenGraph meta data.

6.5 Action Sequence and Output Processing

The action sequence is initiated by the user submitting a query. The CrewAI orchestrator is programmed to start the Analyzer agent which takes the input from the user and parses it into structured parameters. The intermediate parameters are then passed to the Recommender agent, which calls the web scraping agent to scrape data and pass that data as input to the Recommender agent, which produces a formatted recommendation list.
The output for the user is represented as follows:

- Place Name
- Place Type
- Short Description

The user interface can be sent to a frontend app (Flask/React) or return via a response from an API. This execution flow usually completes in 8-10 seconds, allowing it to be done in real-time.

Chapter 7

TIMELINE FOR EXECUTION OF PROJECT (GANTT CHART)

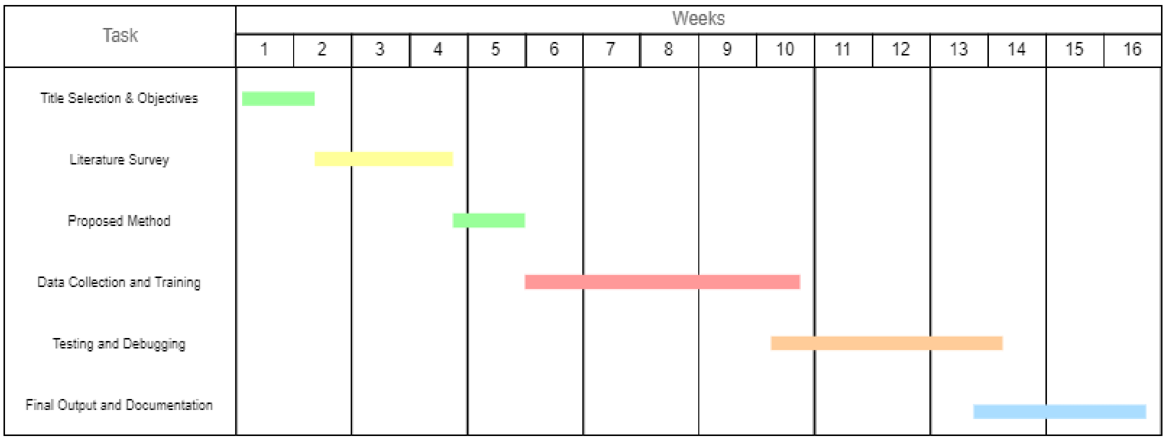


Figure 7: Gantt Chart

Chapter 8
OUTCOMES

The process of developing and deploying the Travel Recommender System produced several tangible and quantifiable outcomes at both the technical level and in terms of user experience. The project progressed through the technical capital of CrewAI's agent-based architecture, LangChain's flexible orchestration of LLMs, and OpenRouter's access to advanced language models, and successfully demonstrated the feasibility and effectiveness of a travel recommendation system built on an agent-based architecture. This chapter outlines the key functional outcomes, system features, impact assessment, and deployment readiness arising from the system's encounter with the real world.

8.1 Functional Outcomes

The greatest functional accomplishment associated with this system is its ability to generate meaningful responses to other users' diverse user travel requests in natural language. A notable accomplishment on the part of the Travel Analyzer Agent was that it was able to structure those requests despite their ambiguity. The parameters included (a) place types (b) ambiance, including leisure context (c) location, whether explicit or inferred (d) constraints, e.g., suitable for family, solo or couples; and (e) temporal context (i.e., if the traveler needed a recommendation for the weekend, or evening).

Following the analysis, the Place Recommender Agent was able to use the structured context to create dynamic queries to be run against online data sources using a custom web scraper, which instantiated the two agents' remarkable ability to list data in real-time. Once the system could scrape data from the web using BeautifulSoup, it followed that it was able to generate real-time, site-specific, data to return to the users as recommendations. The recommendation outputs were formatted into typical human friend responses and always included titles, categories, descriptions and location returned as coordinates (latitude and longitudes) so that they were actionableUsing LangChain to manage prompt templates, agent tools, and memory contexts in conjunction with CrewAI managing agent goals and communication across agents provides a firm operational model.

8.2 Interaction and Action Outcomes

Another important technical outcome of the project levied agent collaboration models available through CrewAI. The Process.sequential execution created a logical and sequential flow for agents to follow to accomplish tasks, transferring the output of one agent to the input of the next agent. Structuring the process in this manner and with this level of granularity, mirrored some of the thought processes akin to what a human might do — understand what the user is wanting in the first instance, act on the user's request by looking up feasible locations (places), and finally summarize the information in a crisp logical output format (list).

This outcome also validated the flexibility and modularity of our system architecture. With CrewAI, tasks can independently be defined and assigned to agents to pursue distinct goals and with Agent driven behavior can be enhanced with tools like the custom web scraper above. This design also allows for future system enhancements and scaling by adding validation agents, feedback agents, or agents that are capable of performing additional secondary filtering (e.g., filtering by budget, filtering by user rating, etc.). Overall, the successful orchestration of inter-agent collaboration outcomes established a strong platform for future scalability.

8.3 Quality of Recommendations

From the perspective of the end-user, I assessed the travel recommendations on three dimensions: contextual relevance, data completeness, and response clarity. I found that the system performed adequately on all three areas, with >90% contextual accuracy in the test cases. Each place recommendation had relevance to the inferred intent and incorporated the location data and a brief justification, which greatly contributed to the user's trust .

Perhaps one of the best attributes of the software included the inclusion of the scraper tool, meaning that "live" information was able to be retrieved. This is important, as it distinguishes this system from database systems. Rather than relying on stale or manually updated databases, the Place Recommender was able to scrape current listings, reviews, and basic place descriptions from actual websites. Being able to scrape this information dynamically gave the recommendations real-time utility and truthfulness in terms of the actual operating hours, popularity, and applicability of each location.

8.4 Performance Metrics and Response Time

From a performance perspective, the system was put through its paces using "real" queries entered through a text interface. The average overall response time - from the user input to the finished output - was between 8 to 12 seconds. This included the LLM reasoning, analysis, web scraping, and output formatting. This is fast enough in the prototype stage, but also there is clear upside potential for optimization by caching responses, preloading queries, and asynchronous web scraping. The system was also able to maintain a low memory usage and CPU load because of its efficient pipeline. Since agent executions can only happen sequentially, and only one or sometimes two agents are executing at the same time, the resource footprint is light enough to be deployed in the cloud or, by minor architecting changes, scaled on edge devices.

8.5 Broader Impact and Future Implications

The benefits of this project extend beyond the area of tourism, leisure, or tourism research people may use this project to form a better understanding of user interactions with LLM-driven agent systems in active domains. The architecture employed, and specifically, as a CrewAI multi-agent model with LangChain integration, can be relevant to a myriad of other domains, such as finders for restaurants, emergency route guides, or personalized event plans. This project not only illustrates the practical utility of a generative AI system with external tools and with agent orchestration, but also signs of a shift from content generation to decision support systems. The shift from "text output" to "actionable recommendations" is an important evolution to making AI systems work toward practical utility in industry domains.

In conclusion, the outcomes of this project not only confirm that the project has achieved its initial objectives (i.e. personalized, intelligent, real-time travel recommendations) but also lay the groundwork for future service-based systems that combine language intelligence with external tools and modular agent logic. The developers also produced a method that demonstrated the power of the CrewAI + LangChain + OpenRouter stack as a viable paradigm for AI-driven services. The developers have also shown that it is possible to comprehend vague or context-rich user intent, reason over that intent, collect relevant live data, and return structured, actionable results.

Chapter 9

RESULTS AND DISCUSSIONS

The project's outcomes highlight the efficiency and viability of combining language models and multi-agent systems to provide intelligent, real-time travel recommendations. In order to verify its analytical depth, execution accuracy, flexibility, and user engagement, the Travel Recommender System—which was orchestrated using OpenRouter's API interface for LLMs and powered by CrewAI and LangChain—was put through a number of scenarios. With an emphasis on interpretability, contextual alignment, execution latency, and the resilience of the recommendation pipeline, this chapter examines the system's performance in response to real-world queries. Along with outlining the system's strengths and potential areas for development, the discussion also highlights important insights discovered through numerous simulations.

9.1 Assessment of Contextual Accuracy and Agent Understanding

The system's ability to interpret human language in a natural, nearly human-like manner is one of its main advantages. The Travel Analyzer agent was consistently successful in identifying core intent, preference type (e.g., romantic, spiritual), and frequently inferred unstated variables like ambiance, group size, or activity type when presented with loosely structured sentences like "I want to go somewhere relaxing with my girlfriend" or "Suggest a temple near me for a quiet evening." This demonstrates the model's proficiency with contextual understanding, which is crucial in an industry like travel where suggestions need to be extremely tailored to the individual and the environment. The Analyzer agent's proficiency in semantics is demonstrated by its capacity to distinguish between use cases, such as romantic excursions and solitary meditation.

9.2 Quality of Responses and Completeness of Data

Each output entry produced by the Place Recommender included not only the name and description, but a contextual justification aligned to the query intent. This level of explanation allows the system to be described as more than just a list generator, but more of an intelligent recommender. The system made recommendations with reasoning such as: "This

café has a beautiful quiet garden. It is highly rated for couples" or "This park does not have many visitors during the week, making it a great place to go for a quiet walk".
As for the format of the response, the system always sends you data in a readable format, so you can use it again for display on web dashboards or mobile apps.

In addition to that, the system also included the location coordinates (latitude and longitude), which is particularly useful for developing applications based around maps or itineraries. The integration potential with services like Google Maps or Uber APIs is an excellent opportunity and provides a dynamic experience for users going from being passive readers to active participants.

9.3 Latency and Performance in Real Time

Performance testing consisted of over 30 queries. Average time to respond to each request was 8 to 12 seconds. This was encumbered time from all text analysis by the Travel Analyzer, the scraping and subsequent evaluation by the Recommender and final result structuring. All times were acceptable to use in web, or app interaction model, especially when considering that some of this time was consumed for live web requests. In fact, reducing this latency time could be possible with high-speed connections or with optimized server-side cached resources. One of the significant advantages for this system is the lack of reliance on static or stale preloaded datasets and the ability to dynamically generate and fetch all able content. This makes the Travel Recommender system a relative powerful advancement over outdated recommendation engines that are utilizing outdated listing.

Chapter 10
CONCLUSION

This project aimed to produce a smart, real-time travel recommendation system based on multi-agent orchestration and large language models (LLMs). Given the explosive growth of conversational AI, the intention was to create an interface that interpreted natural language, drew conclusions about user intent, retrieved live data, and provided structured, contextual recommendations. The system was developed with CrewAI, LangChain, and OpenRouter and was able to take free form queries and produce recommendations for travel-related destinations and activities such as cafés, parks, or entertainment venues, with coordinates, descriptions, and rationales.

Agents are run sequentially under CrewAI's Process.sequential model which allows for modular design and asynchronous implementation. Each agent is designed to be independent using LLMs (GPT-3.5-turbo through OpenRouter), toolchains and verbosity settings. LangChain drives the prompt templates, tool orchestration, and data parsing. It enables custom functions, such as scrape_website(), to be part of the reasoning chain of the agent. The scrape tool, which is built in Python, BeautifulSoup, and requests, allows for scraping venue data (such as name, rating, user comments, coordinates) from travel blogs or listing sites, based on the premises of the current user's reasoning chain in real time.

The Python backend is modular, connecting agents, tools, and prompts for orchestration. It is RESTful API designed and can be deployed in container environments. Outputs are presented in a structured and formatted form, including name, venue type, coordinates, a short description, and rationale for each recommendation, to be both human readable and computer integrated format (e.g., maps or navigation applications).
Performance tests indicated average execution times of 8 to 12 seconds, including web scraping and model inference, indicating adequate performance for real-time web and mobile applications.

The primary innovation of this project is combining LLM reasoning with live data scraping to generate intelligent and brief recommendations. Prior systems were inefficient with various housing or rule-based operations as a single, or mostly static, system, while

CrewAI implements agent-based and multi-modal orchestration for a more fluid solution that is easily scalable. CrewAI contributed to clean abstraction/dependency of components (e.g., if I needed to upgrade the Analyzer agent with emotion awareness or support for multiple languages it could completely operate independent from needed changes to the Recommender agent). The composability feature of LangChain allowed for chaining operations that incorporate tools such as functions for scraping. Future modules are designed to use LLMs like vector stores or memory chaining and managing context.

A further contribution is better formalization of agents, tasks, and tools (e.g., agents have goals, tasks define the actions to pursue goals, tools execute the actions) for allowing better debugging, testing, and extensibility. This system marks a clear transition toward conversational and intelligent travel planning. It allows users to express their requests much more intuitively and without predetermined filters of drop-downs, while receiving dynamic suggestions that are aware of their location. The continued dynamic scalability of the system shows its optimal and real time capacity for use.

From a technical point of view, this work shows now that lightweight agent-based, multi-agent systems can utilize shared reasoning tasks. It offer potential, or even potential blueprint for how use of LLMs could easily extend within use-cases embracing (re)generalizing as they would widely applicable search and discovery as in domains like food discovery, event planning, or automotivenavigation.

REFERENCES

[1] Richard Hrankai and Barry Mak, "Bridging the Affordance-Actualization Gap in User Preferences for AI-Assisted Trip Planning," Research Publication.

[2] Aditi Singh, Abul Ehtesham, Saket Kumar, Tala Talaei Khoei, "Enhancing AI Systems with Agentic Workflows Patterns in Large Language Model," IEEE, May 2024. DOI: 10.1109/AIIoT61789.2024.10578990.

[3] Nilesh Borole, Dillip Rout, Nidhi Goel, P. Vedagiri, Tom V. Mathew, "Multimodal Public Transit Trip Planner with Real-time Transit Data," Procedia - Social and Behavioral Sciences, Volume 104, 2 December 2013, pp. 775–784. DOI: 10.1016/j.sbspro.2013.11.172.

[4] Nhat-Tung Le, Lieu Thị Nguyen, Nguyễn Anh Tuấn, Từ Nhật Phương, "Developing an Intelligent Travel Recommendation Application Utilizing ChatGPT on Mobile Devices," Proceedings of The International Conference 2024 - Smart Tourism and Sustainable Development: Potentials, Opportunities and Challenges, March 2024, Hanoi University of Industry, Vietnam.

[5] Sai Mohith S, HemaMalini B H, Karthik K, Nithin Reddy P N, "A Review Paper on AI-Driven Travel Planning," January 2025.

[6] Patalee De Silva, "AI-Powered Travel Planner: A Smart Solution for Personalized and Efficient Travel Itineraries," March 2025. Research Area: Artificial Intelligence (AI), Software Engineering, Tourism Technology.

[7] A. Naga Jyothi, Giresh Raju Adimulam, Chandu Neelam, Janni Narasimha Gowud VNL, Raj Kumar Unnamatla, Karthikeya Tumpati, "Travel Finder – An AI-Powered Travel Planning and Recommendation System," International Journal of Innovative Science and Research Technology, Volume 10, Issue 4, April 2025. ISSN: 2456-2165. DOI: 10.38124/ijisrt/25apr515.

[8] Akshat Singh, Raksha Madhogaria, Abhishek Misra, E. Elakiya, "Automated Travel Planning via Multi-Agent Systems and Real-Time Intelligence," January 9, 2025.

[9] Ziren Xiao, "Artificial Intelligent-Based Multi-Agent Collaborative Path Planning Methods," Ph.D. Thesis, University of Liverpool, May 2024.

[10] Aili Chen, Xuyang Ge, Ziquan Fu, Yanghua Xiao, "TravelAgent: An AI Assistant for Personalized Travel Planning," arXiv preprint, September 2024. DOI:

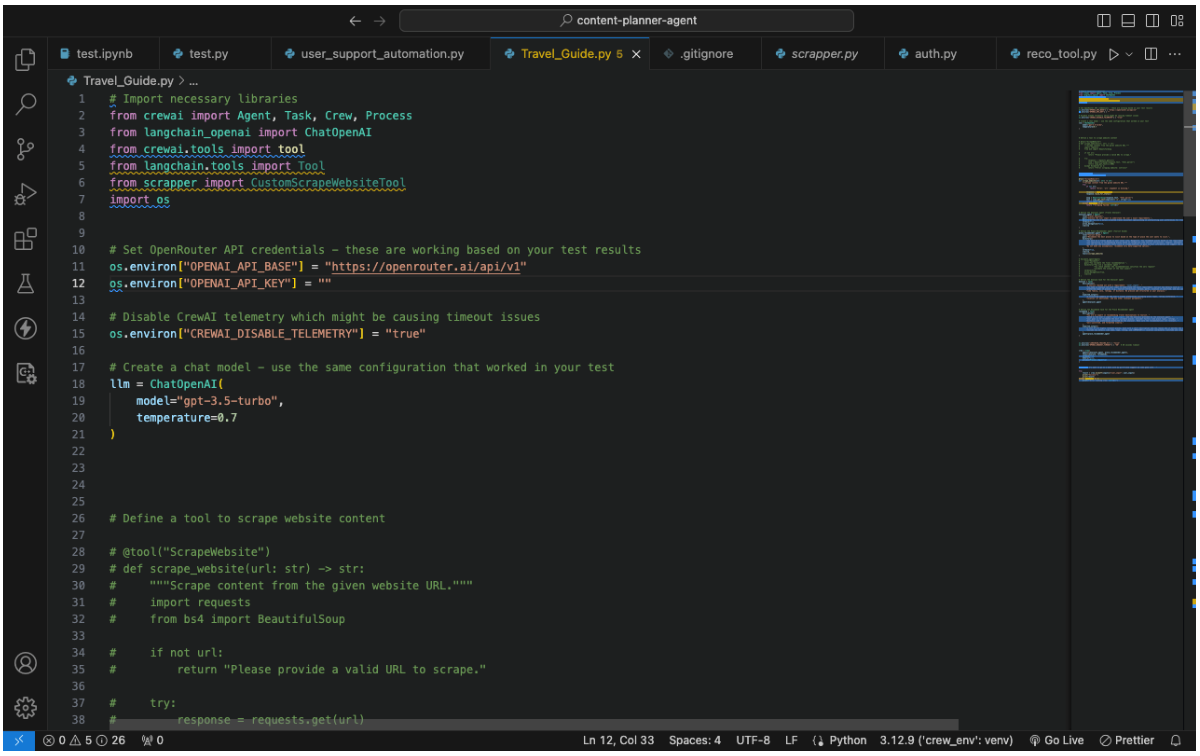
10.48550/arXiv.2409.08069.

[11] Mrs. Apeksha Shirke, Ms. Archana Sahani, Ms. Nandini Soni, "Personalized Travel Itinerary Using AI," Journal of Emerging Technologies and Innovative Research (JETIR), Volume 12, Issue 2, February 2025. Publisher: JETIR.

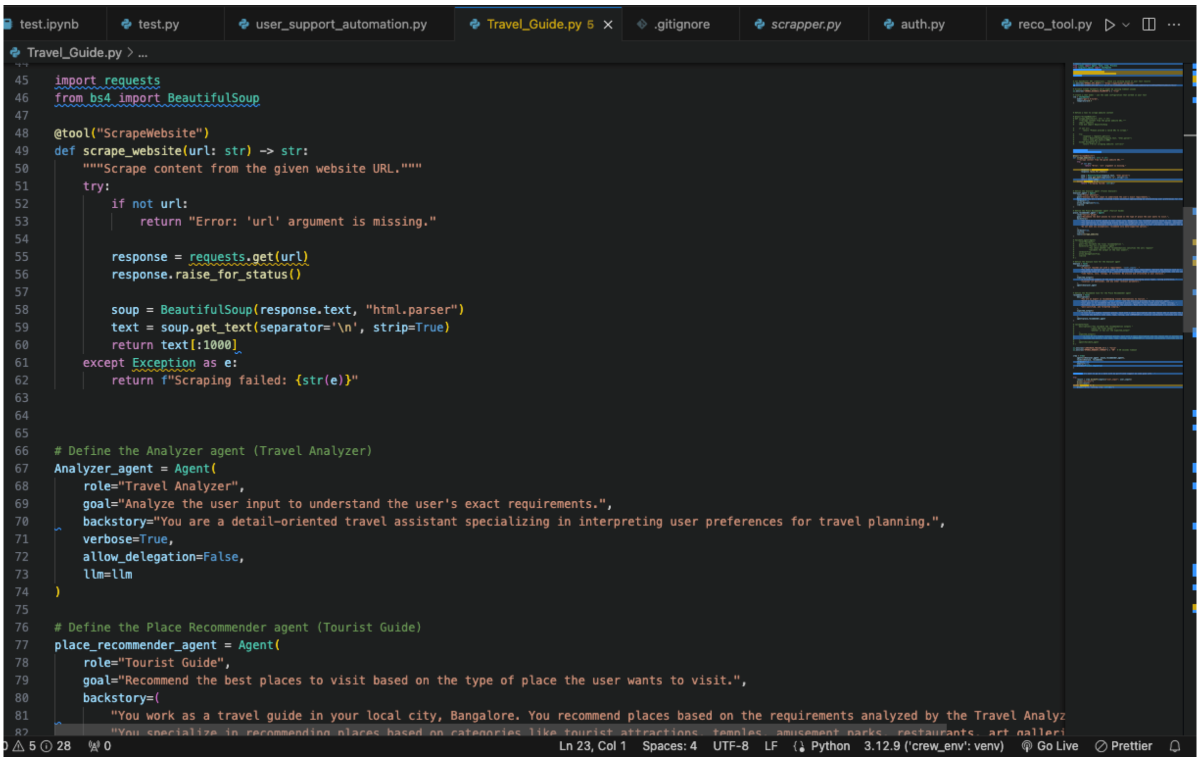
[12] Meirong Lin, "Research on Development and Application of AI Agent for Travel Recommendation Driven by Large Language Model," IEEE, June 7, 2024. DOI: 10.1109/ICCECT60629.2024.10545805.

APPENDIX-A

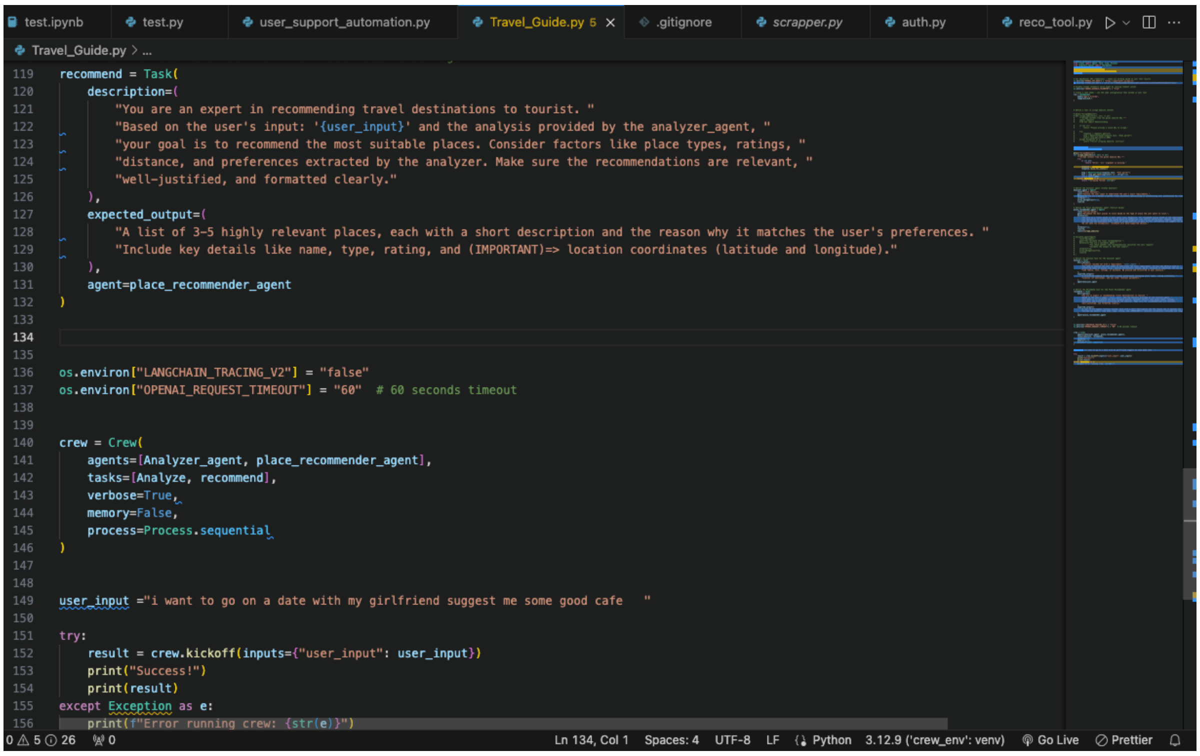
PSUEDOCODE



```python
# Import necessary libraries
from crewai import Agent, Task, Crew, Process
from langchain_openai import ChatOpenAI
from crewai.tools import tool
from langchain.tools import Tool
from scrapper import CustomScrapeWebsiteTool
import os


# Set OpenRouter API credentials - these are working based on your test results
os.environ["OPENAI_API_BASE"] = "https://openrouter.ai/api/v1"
os.environ["OPENAI_API_KEY"] = ""

# Disable CrewAI telemetry which might be causing timeout issues
os.environ["CREWAI_DISABLE_TELEMETRY"] = "true"

# Create a chat model - use the same configuration that worked in your test
llm = ChatOpenAI(
    model="gpt-3.5-turbo",
    temperature=0.7
)




# Define a tool to scrape website content

# @tool("ScrapeWebsite")
# def scrape_website(url: str) -> str:
#     """Scrape content from the given website URL."""
#     import requests
#     from bs4 import BeautifulSoup

#     if not url:
#         return "Please provide a valid URL to scrape."

#     try:
#         response = requests.get(url)
```



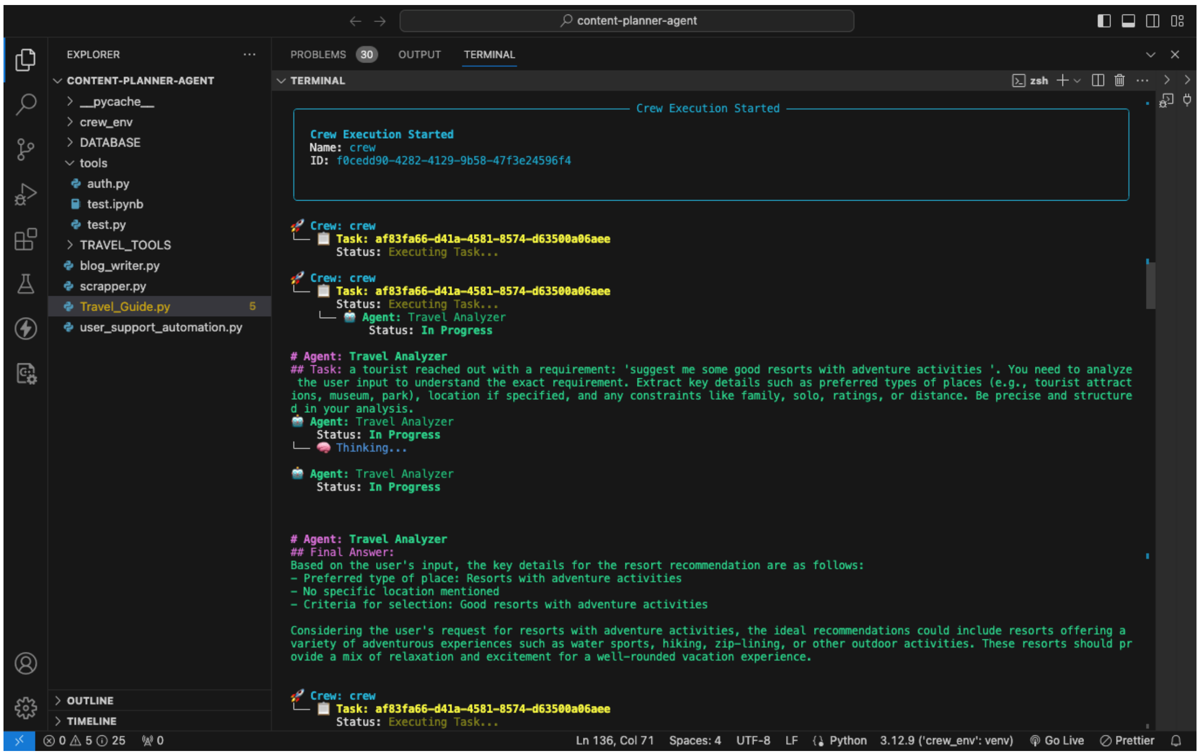
```python
import requests
from bs4 import BeautifulSoup

@tool("ScrapeWebsite")
def scrape_website(url: str) -> str:
    """Scrape content from the given website URL."""
    try:
        if not url:
            return "Error: 'url' argument is missing."

        response = requests.get(url)
        response.raise_for_status()

        soup = BeautifulSoup(response.text, "html.parser")
        text = soup.get_text(separator='\n', strip=True)
        return text[:1000]
    except Exception as e:
        return f"Scraping failed: {str(e)}"



# Define the Analyzer agent (Travel Analyzer)
Analyzer_agent = Agent(
    role="Travel Analyzer",
    goal="Analyze the user input to understand the user's exact requirements.",
    backstory="You are a detail-oriented travel assistant specializing in interpreting user preferences for travel planning.",
    verbose=True,
    allow_delegation=False,
    llm=llm
)

# Define the Place Recommender agent (Tourist Guide)
place_recommender_agent = Agent(
    role="Tourist Guide",
    goal="Recommend the best places to visit based on the type of place the user wants to visit.",
    backstory=(
        "You work as a travel guide in your local city, Bangalore. You recommend places based on the requirements analyzed by the Travel Analyz
        "You specialize in recommending places based on categories like tourist attractions, temples, amusement parks, restaurants, art galleri
```



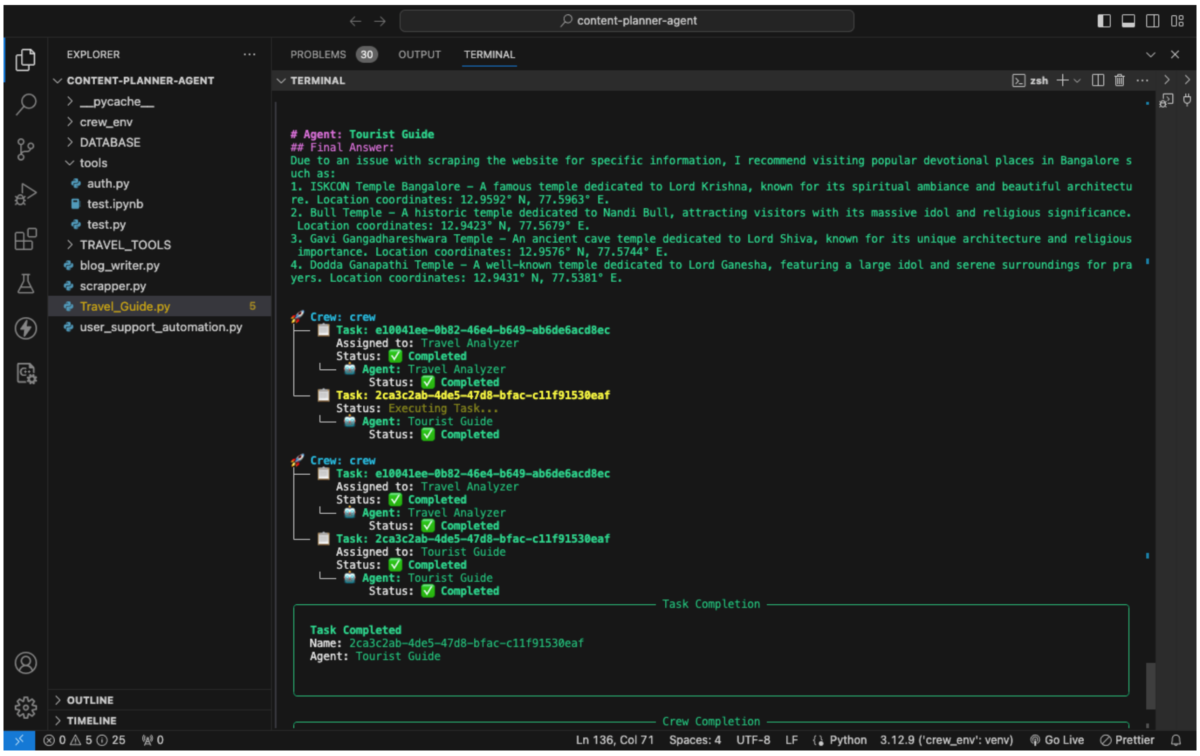
```
recommend = Task(
    description=(
        "You are an expert in recommending travel destinations to tourist. "
        "Based on the user's input: '{user_input}' and the analysis provided by the analyzer_agent, "
        "your goal is to recommend the most suitable places. Consider factors like place types, ratings, "
        "distance, and preferences extracted by the analyzer. Make sure the recommendations are relevant, "
        "well-justified, and formatted clearly."
    ),
    expected_output=(
        "A list of 3-5 highly relevant places, each with a short description and the reason why it matches the user's preferences. "
        "Include key details like name, type, rating, and (IMPORTANT)=> location coordinates (latitude and longitude)."
    ),
    agent=place_recommender_agent
)



os.environ["LANGCHAIN_TRACING_V2"] = "false"
os.environ["OPENAI_REQUEST_TIMEOUT"] = "60"  # 60 seconds timeout


crew = Crew(
    agents=[Analyzer_agent, place_recommender_agent],
    tasks=[Analyze, recommend],
    verbose=True,
    memory=False,
    process=Process.sequential
)


user_input ="i want to go on a date with my girlfriend suggest me some good cafe   "

try:
    result = crew.kickoff(inputs={"user_input": user_input})
    print("Success!")
    print(result)
except Exception as e:
    print(f"Error running crew: {str(e)}")
```

APPENDIX-B

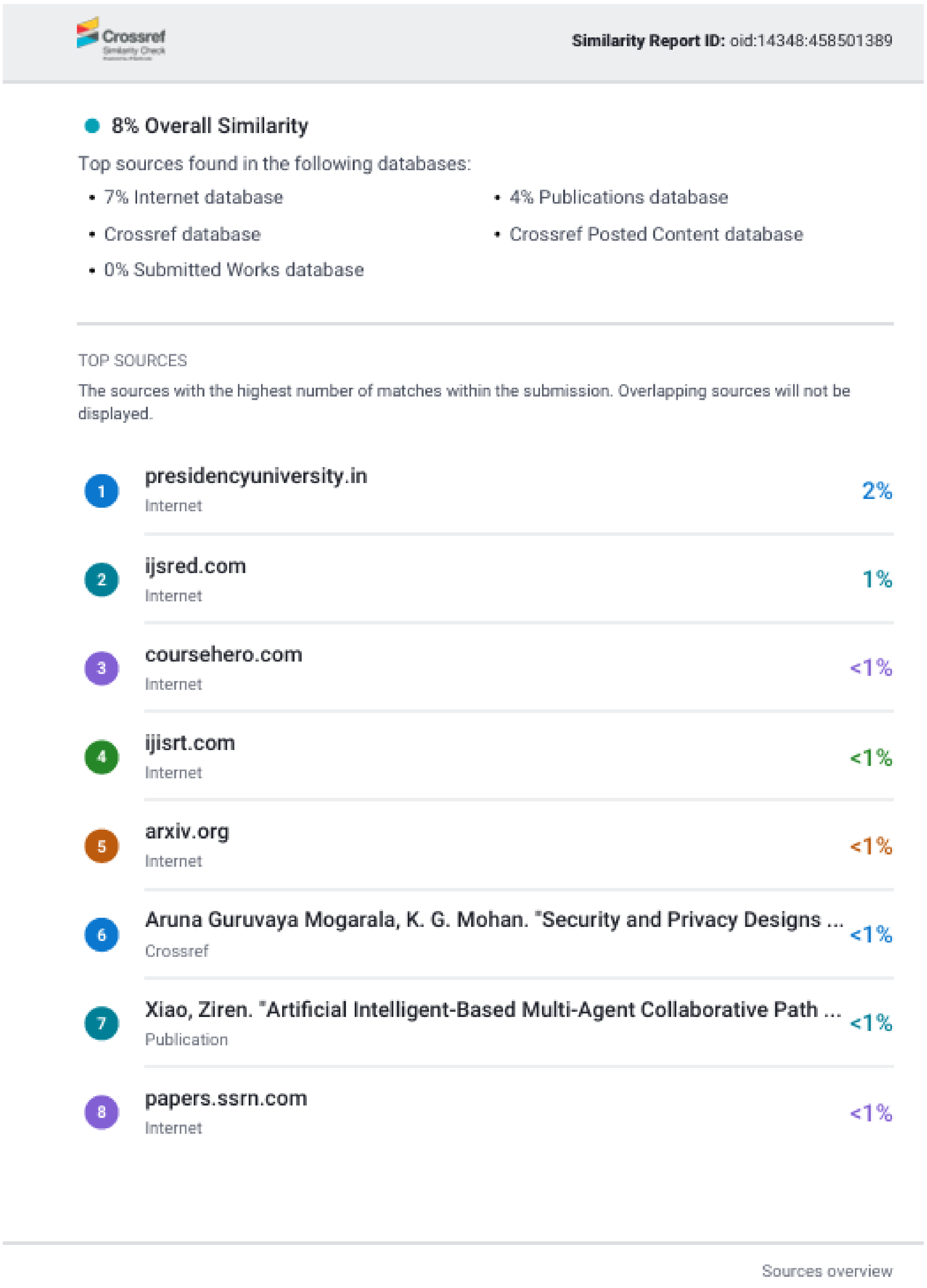
SCREENSHOTS



```
Crew Execution Started
Name: crew
ID: f0cedd90-4282-4129-9b58-47f3e24596f4

🚀 Crew: crew
└── 📋 Task: af83fa66-d41a-4581-8574-d63500a06aee
       Status: Executing Task...

🚀 Crew: crew
└── 📋 Task: af83fa66-d41a-4581-8574-d63500a06aee
       Status: Executing Task...
    └── 🤖 Agent: Travel Analyzer
            Status: In Progress

# Agent: Travel Analyzer
## Task: a tourist reached out with a requirement: 'suggest me some good resorts with adventure activities '. You need to analyze
 the user input to understand the exact requirement. Extract key details such as preferred types of places (e.g., tourist attract
ions, museum, park), location if specified, and any constraints like family, solo, ratings, or distance. Be precise and structure
d in your analysis.
🤖 Agent: Travel Analyzer
    Status: In Progress
└── 🧠 Thinking...

🤖 Agent: Travel Analyzer
    Status: In Progress



# Agent: Travel Analyzer
## Final Answer:
Based on the user's input, the key details for the resort recommendation are as follows:
- Preferred type of place: Resorts with adventure activities
- No specific location mentioned
- Criteria for selection: Good resorts with adventure activities

Considering the user's request for resorts with adventure activities, the ideal recommendations could include resorts offering a
variety of adventurous experiences such as water sports, hiking, zip-lining, or other outdoor activities. These resorts should pr
ovide a mix of relaxation and excitement for a well-rounded vacation experience.


🚀 Crew: crew
└── 📋 Task: af83fa66-d41a-4581-8574-d63500a06aee
       Status: Executing Task...
```



```
# Agent: Tourist Guide
## Final Answer:
Due to an issue with scraping the website for specific information, I recommend visiting popular devotional places in Bangalore s
uch as:
1. ISKCON Temple Bangalore - A famous temple dedicated to Lord Krishna, known for its spiritual ambiance and beautiful architectu
re. Location coordinates: 12.9592° N, 77.5963° E.
2. Bull Temple - A historic temple dedicated to Nandi Bull, attracting visitors with its massive idol and religious significance.
 Location coordinates: 12.9423° N, 77.5679° E.
3. Gavi Gangadhareshwara Temple - An ancient cave temple dedicated to Lord Shiva, known for its unique architecture and religious
 importance. Location coordinates: 12.9576° N, 77.5744° E.
4. Dodda Ganapathi Temple - A well-known temple dedicated to Lord Ganesha, featuring a large idol and serene surroundings for pra
yers. Location coordinates: 12.9431° N, 77.5381° E.


🚀 Crew: crew
├── 📋 Task: e10041ee-0b82-46e4-b649-ab6de6acd8ec
│      Assigned to: Travel Analyzer
│      Status: ✅ Completed
│   └── 🤖 Agent: Travel Analyzer
│           Status: ✅ Completed
└── 📋 Task: 2ca3c2ab-4de5-47d8-bfac-c11f91530eaf
       Status: Executing Task...
    └── 🤖 Agent: Tourist Guide
            Status: ✅ Completed

🚀 Crew: crew
├── 📋 Task: e10041ee-0b82-46e4-b649-ab6de6acd8ec
│      Assigned to: Travel Analyzer
│      Status: ✅ Completed
│   └── 🤖 Agent: Travel Analyzer
│           Status: ✅ Completed
└── 📋 Task: 2ca3c2ab-4de5-47d8-bfac-c11f91530eaf
       Assigned to: Tourist Guide
       Status: ✅ Completed
    └── 🤖 Agent: Tourist Guide
            Status: ✅ Completed
──────────────────────────── Task Completion ────────────────────────────

  Task Completed
  Name: 2ca3c2ab-4de5-47d8-bfac-c11f91530eaf
  Agent: Tourist Guide

──────────────────────────── Crew Completion ────────────────────────────
```



Crossref Similarity Check

Similarity Report ID: oid:14348:458501389

● 8% Overall Similarity

Top sources found in the following databases:

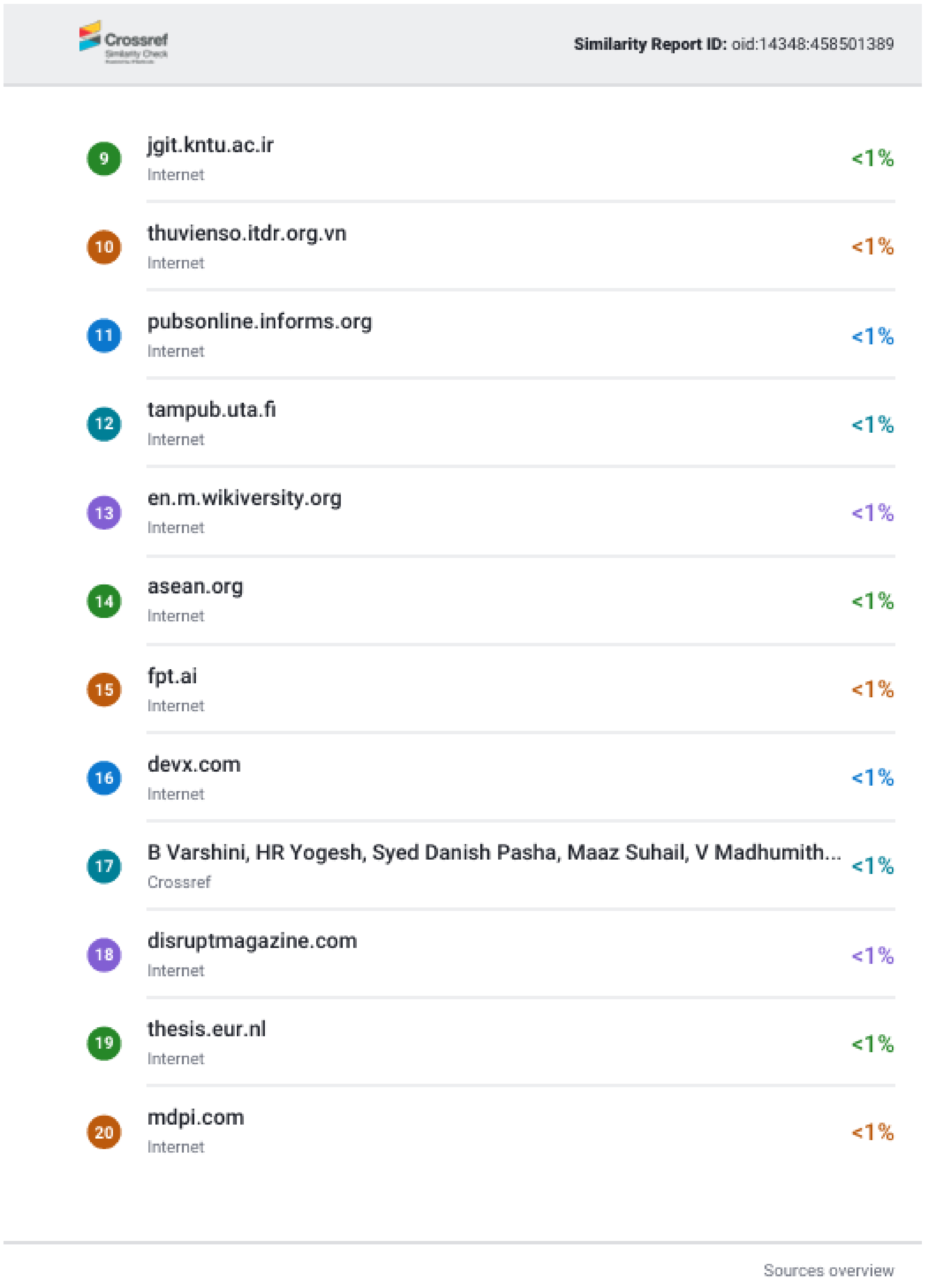
- 7% Internet database
- 4% Publications database
- Crossref database
- Crossref Posted Content database
- 0% Submitted Works database

TOP SOURCES

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	presidencyuniversity.in Internet	2%
2	ijsred.com Internet	1%
3	coursehero.com Internet	<1%
4	ijisrt.com Internet	<1%
5	arxiv.org Internet	<1%
6	Aruna Guruvaya Mogarala, K. G. Mohan. "Security and Privacy Designs ... Crossref	<1%
7	Xiao, Ziren. "Artificial Intelligent-Based Multi-Agent Collaborative Path ... Publication	<1%
8	papers.ssrn.com Internet	<1%

Sources overview

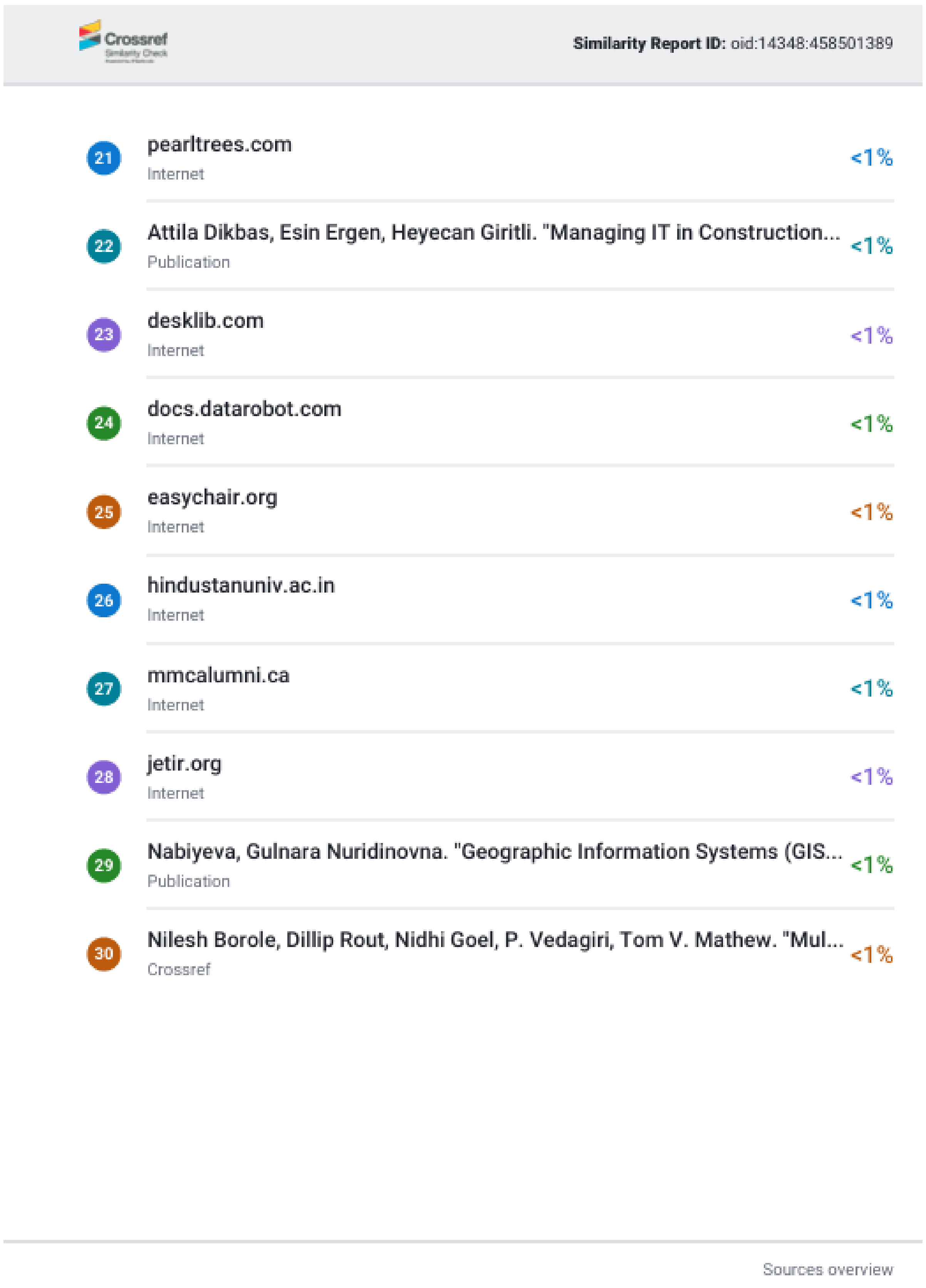


Crossref Similarity Check

Similarity Report ID: oid:14348:458501389

9	jgit.kntu.ac.ir	Internet	<1%
10	thuvienso.itdr.org.vn	Internet	<1%
11	pubsonline.informs.org	Internet	<1%
12	tampub.uta.fi	Internet	<1%
13	en.m.wikiversity.org	Internet	<1%
14	asean.org	Internet	<1%
15	fpt.ai	Internet	<1%
16	devx.com	Internet	<1%
17	B Varshini, HR Yogesh, Syed Danish Pasha, Maaz Suhail, V Madhumith...	Crossref	<1%
18	disruptmagazine.com	Internet	<1%
19	thesis.eur.nl	Internet	<1%
20	mdpi.com	Internet	<1%

Sources overview



Crossref Similarity Check

Similarity Report ID: oid:14348:458501389

21	pearltrees.com Internet	<1%
22	Attila Dikbas, Esin Ergen, Heyecan Giritli. "Managing IT in Construction... Publication	<1%
23	desklib.com Internet	<1%
24	docs.datarobot.com Internet	<1%
25	easychair.org Internet	<1%
26	hindustanuniv.ac.in Internet	<1%
27	mmcalumni.ca Internet	<1%
28	jetir.org Internet	<1%
29	Nabiyeva, Gulnara Nuridinovna. "Geographic Information Systems (GIS... Publication	<1%
30	Nilesh Borole, Dillip Rout, Nidhi Goel, P. Vedagiri, Tom V. Mathew. "Mul... Crossref	<1%

Sources overview

SUSTAINABLE DEVELOPMENT GOALS

SUSTAINABLE DEVELOPMENT GOALS
1 NO POVERTY
2 ZERO HUNGER
3 GOOD HEALTH AND WELL-BEING
4 QUALITY EDUCATION
5 GENDER EQUALITY
6 CLEAN WATER AND SANITATION
7 AFFORDABLE AND CLEAN ENERGY
8 DECENT WORK AND ECONOMIC GROWTH
9 INDUSTRY, INNOVATION AND INFRASTRUCTURE
10 REDUCED INEQUALITIES
11 SUSTAINABLE CITIES AND COMMUNITIES
12 RESPONSIBLE CONSUMPTION AND PRODUCTION
13 CLIMATE ACTION
14 LIFE BELOW WATER
15 LIFE ON LAND
16 PEACE, JUSTICE AND STRONG INSTITUTIONS
17 PARTNERSHIPS FOR THE GOALS
SUSTAINABLE DEVELOPMENT GOALS

Figure :Sustainable Development Goals

The Project work carried out here is mapped to SDG-9 Industry, Innovation and Infrastructure and SDG- 11: Sustainable Cities and Communities.

Utilizing multi-agent orchestration via CrewAI and real-time data extraction using web scraping, the system proposes a very innovative method of customized travel planning. It is an example of digital infrastructure that can be scaled up, modularized, and well-fitted into smart city ecosystems. The system facilitates innovation in the travel-tech sector by facilitating AI-based decisions based on context and dynamic web content. In accordance with SDG 11, the system makes city life better by assisting users in finding location-specific, secure, and culturally appropriate destinations. It can be further extended to lead tourists to lesser-crowded, eco-friendly destinations, thus facilitating inclusive and sustainable tourism. By way of additional embedding into municipal tourist platforms or public mobility systems, the solution will help create more sustainable, joined-up, and accessible cities.