1. **INTRODUCTION**
   1. **Project Overview**

Explore with AI: Custom Itineraries for Your Next Journey is a generative AI–based web application designed to simplify and automate travel planning. The project allows users to generate personalized travel itineraries by providing basic inputs such as destination, number of days, and number of nights through an interactive web interface.

The system leverages a pre-trained generative AI model to create detailed and structured travel itineraries that include daily activities, nearby attractions, food recommendations, and useful travel tips. Instead of relying on manual research or static datasets, the application processes real-time user inputs and dynamically generates customized travel plans.

The application is developed using Python and Streamlit, providing a user-friendly interface for seamless interaction. By integrating generative AI with a lightweight web framework, the project demonstrates the practical application of AI in real-world travel planning scenarios. This solution is beneficial for individual travelers, travel agencies, and travel content creators by reducing planning effort, saving time, and improving the overall travel planning experience.

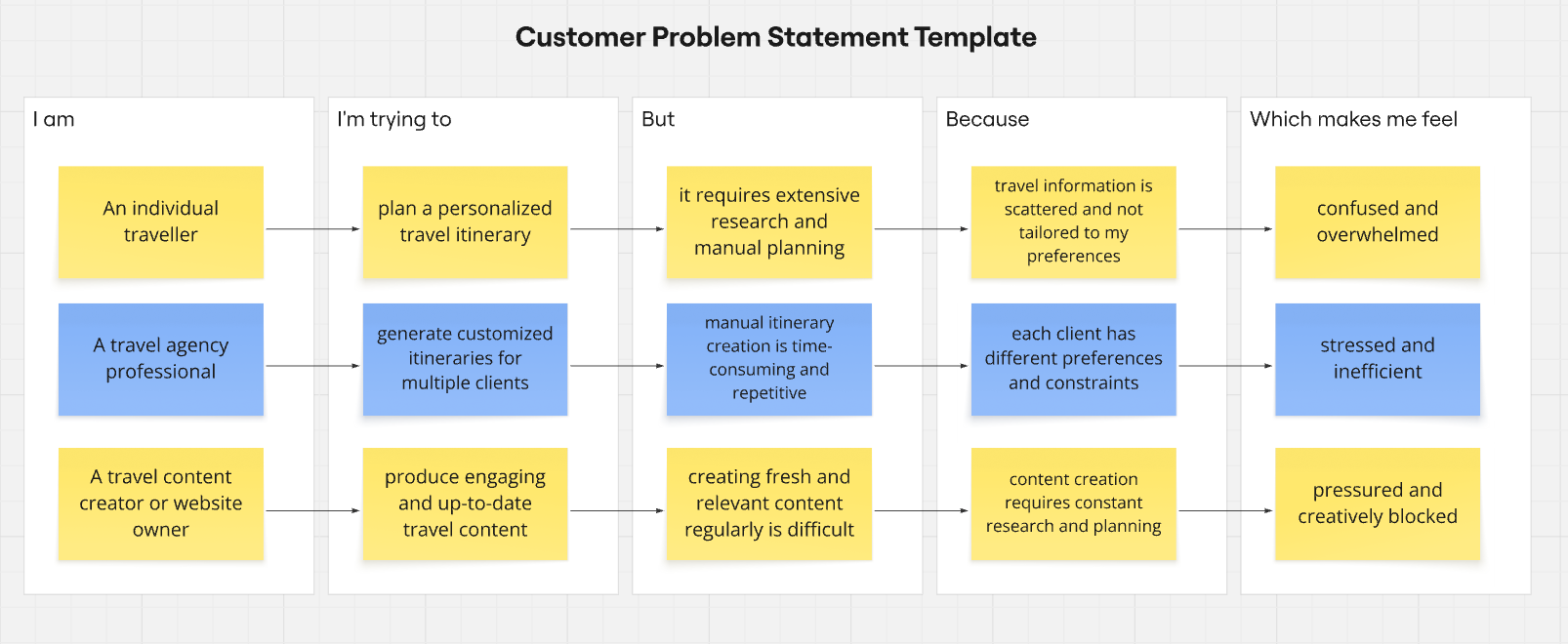
**1.2 Objectives**

* To develop an AI-based application for generating personalized travel itineraries.
* To use a pre-trained generative AI model for automated itinerary creation.
* To design a simple user interface using Streamlit for easy interaction.
* To validate user inputs for accurate and meaningful itinerary generation.
* To deploy the application as a working travel planning solution.

1. **Ideation Phase**

2.1 **Define Problem Statements (Customer Problem Statement Template):**

Travel planning and content creation often take a lot of time and effort because they require continuous research and personalization. Many individual travellers find it difficult to create customized travel itineraries that match their interests and trip duration, which can be confusing and frustrating. In the same way, travel agencies struggle to quickly prepare personalized itineraries for multiple clients due to repetitive manual work. Travel websites and content creators also face challenges in regularly producing engaging and updated travel content. These difficulties create a need for an intelligent and automated solution. The **Explore with AI** system addresses these challenges by using generative AI to automatically generate personalized travel itineraries and relevant travel content with minimal manual effort.



Reference: https://miro.com/templates/customer-problem-statement/

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Problem Statement (PS)** | **I am (Customer)** | **I’m trying to** | **But** | **Because** | **Which makes me feel** |
| **PS-1** | An individual traveller | plan a personalized travel itinerary | it requires extensive research and manual planning | travel information is scattered and not tailored to my preferences | confused and overwhelmed |
| **PS-2** | A travel agency professional | generate customized itineraries for multiple clients | manual itinerary creation is time-consuming and repetitive | each client has different preferences and constraints | stressed and inefficient |
| **PS-3** | A travel content creator or website owner | produce engaging and up-to-date travel content | creating fresh and relevant content regularly is difficult | content creation requires constant research and planning | pressured and creatively blocked |

**2.2 Empathy Map Canvas**

**Empathy Map Canvas:**

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user’s behaviours and attitudes.

It is a useful tool to helps teams better understand their users.

Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user’s perspective along with his or her goals and challenges.

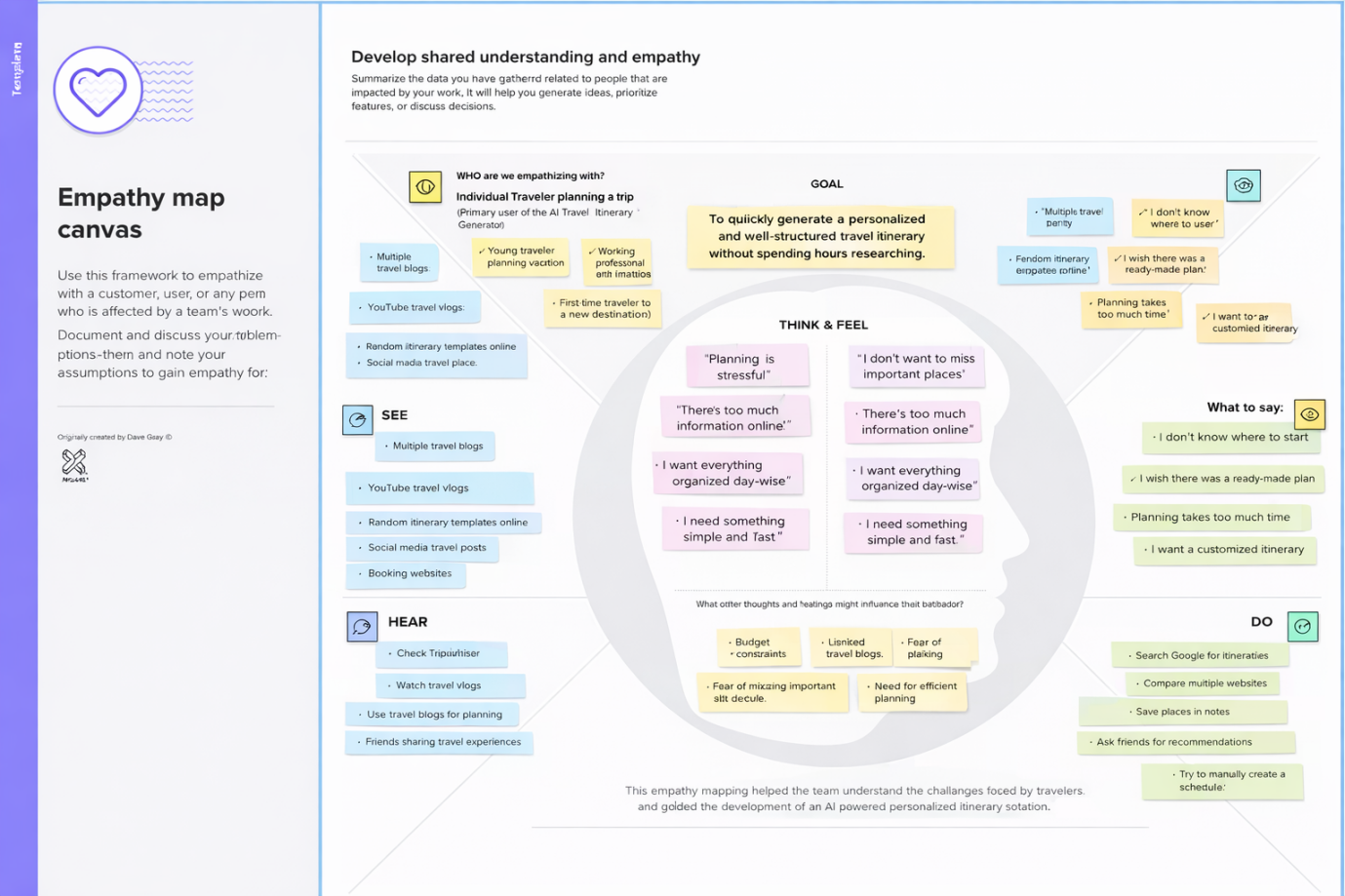
**Example:**

Diagram

Description automatically generated

Reference: <https://www.mural.co/templates/empathy-map-canvas>

**Example: Travel Itinerary Application**



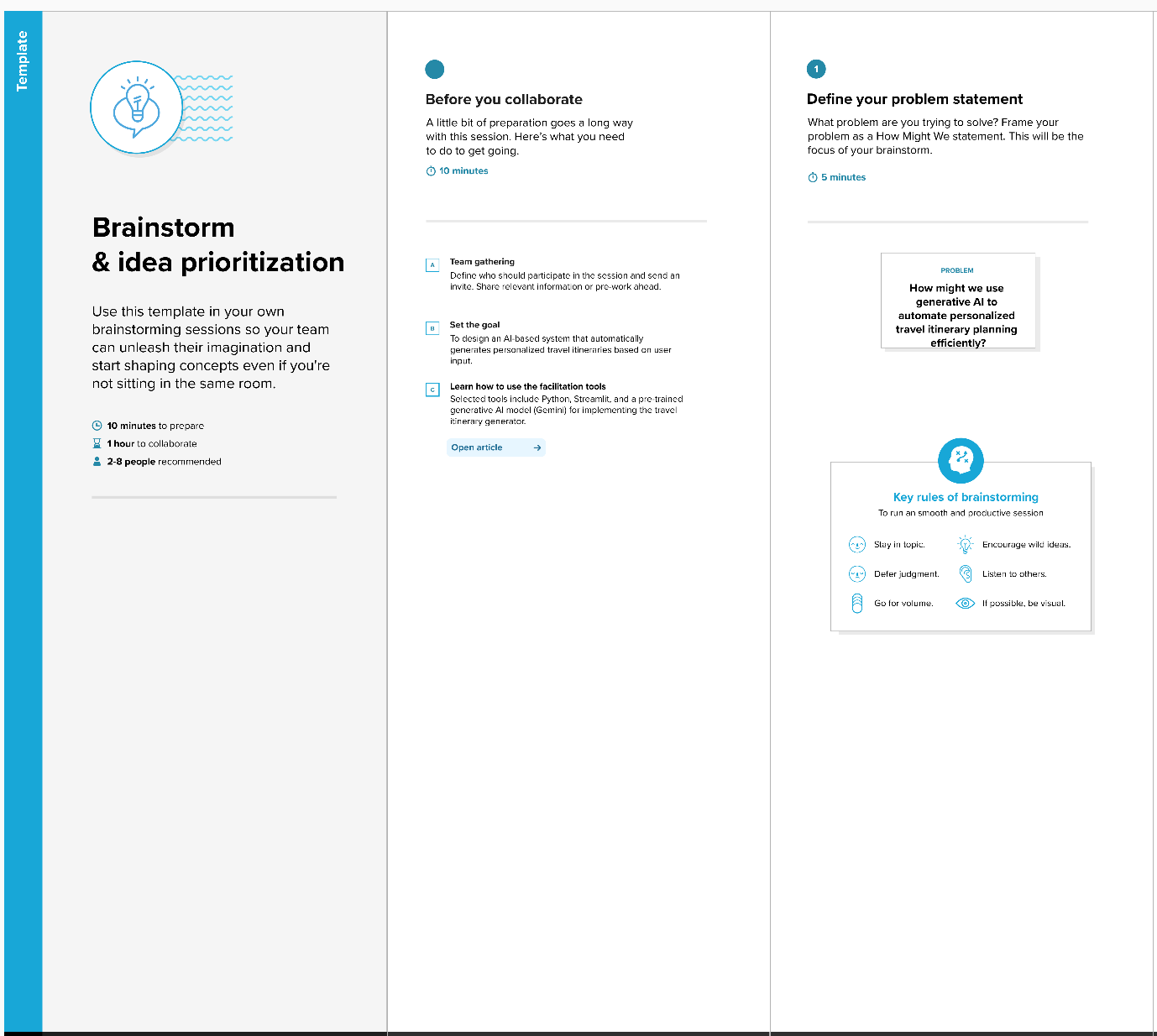
**2.3 Brainstorming**

**Brainstorm & Idea Prioritization:**

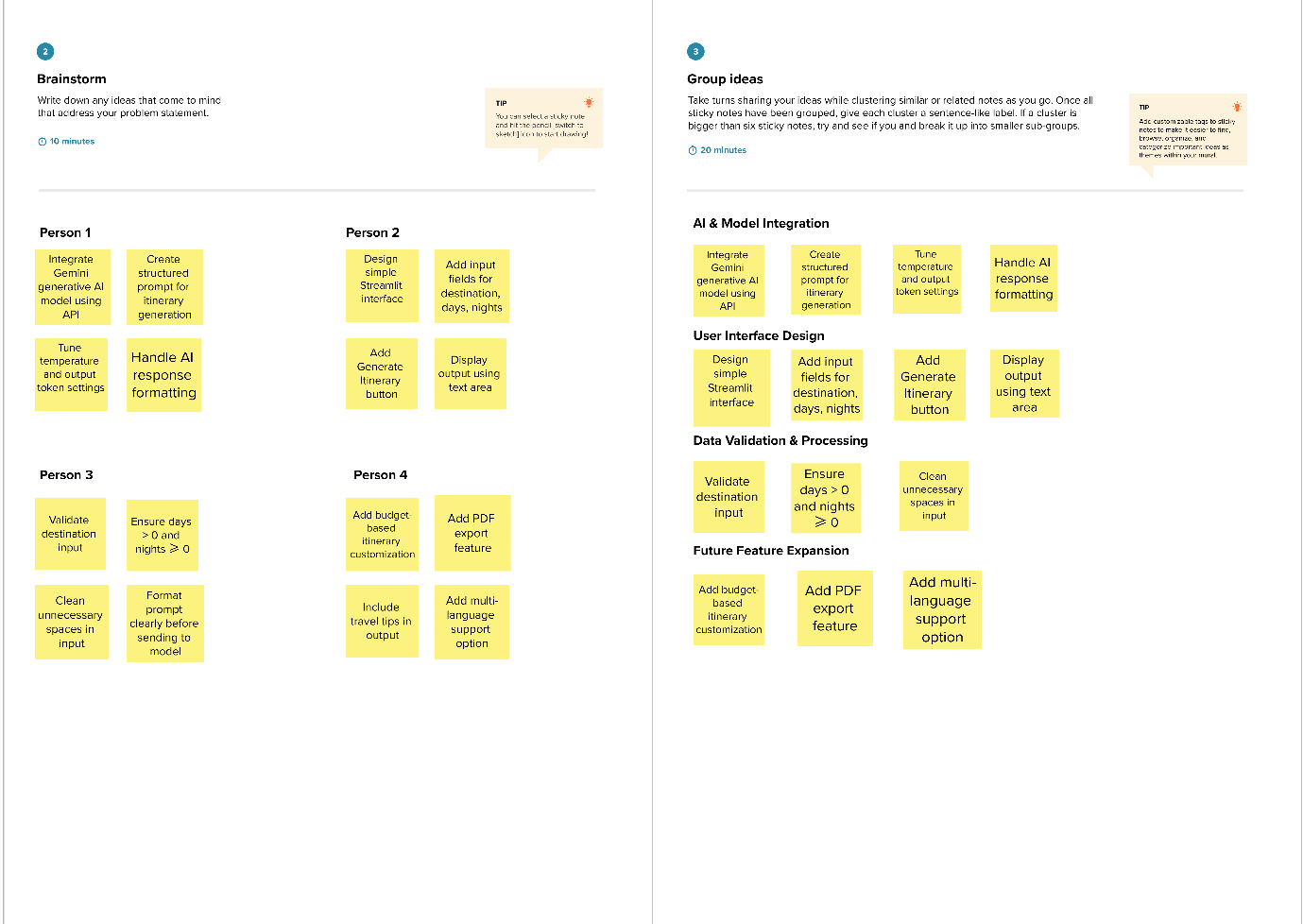
Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.

Reference: <https://www.mural.co/templates/brainstorm-and-idea-prioritization>

**Step-1: Team Gathering, Collaboration and Select the Problem Statement**



**Step-2: Brainstorm, Idea Listing and Grouping**



**Step-3: Idea Prioritization**

**Diagram

Description automatically generated**

1. **Requirement Analysis**

**Functional Requirements:**

Following are the functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Functional Requirement (Epic)** | **Sub Requirement (Story / Sub-Task)** |
| FR-1 | Travel Itinerary Generation | * User can enter destination. * User can enter number of days. * User can enter number of nights. * System generates personalized travel itinerary using AI. |
| FR-2 | Input Validation | * Validate that destination field is not empty. * Ensure number of days is greater than zero. * Ensure number of nights is zero or positive. * Display error message for invalid input. |
| FR-3 | AI Model Integration | * Create structured prompt using user input. * Send prompt to generative AI model. * Receive generated itinerary from AI. * Format AI response properly. |
| FR-4 | Output Display | * Display itinerary in readable format. * Allow user to review generated content. * Show appropriate error message if API fails. |

**Non-functional Requirements:**

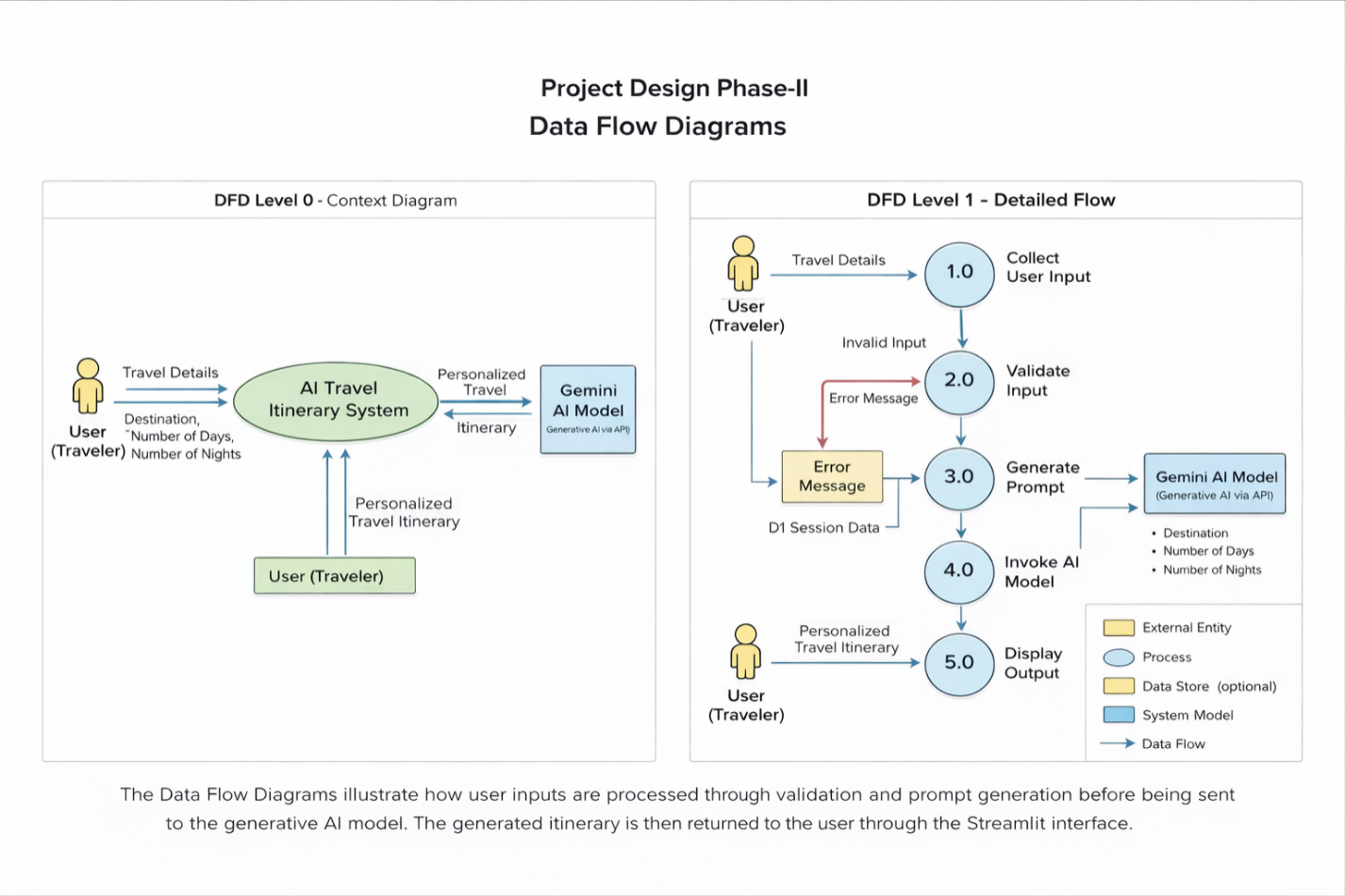
Following are the non-functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Non-Functional Requirement** | **Description** |
| NFR-1 | **Usability** | The application should provide a simple, intuitive, and user-friendly interface using Streamlit. |
| NFR-2 | **Security** | The API key must be securely configured and not exposed publicly. User input should not be stored permanently. |
| NFR-3 | **Reliability** | The system should handle invalid inputs and API errors gracefully without crashing. |
| NFR-4 | **Performance** | The application should generate travel itineraries within a reasonable response time. |
| NFR-5 | **Availability** | The system should be accessible whenever the Streamlit server is running and internet connectivity is available. |

**3.2 Data Flow Diagram & User Stories**

**Data Flow Diagrams:**

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

****

**User Stories**

Use the below template to list all the user stories for the product.

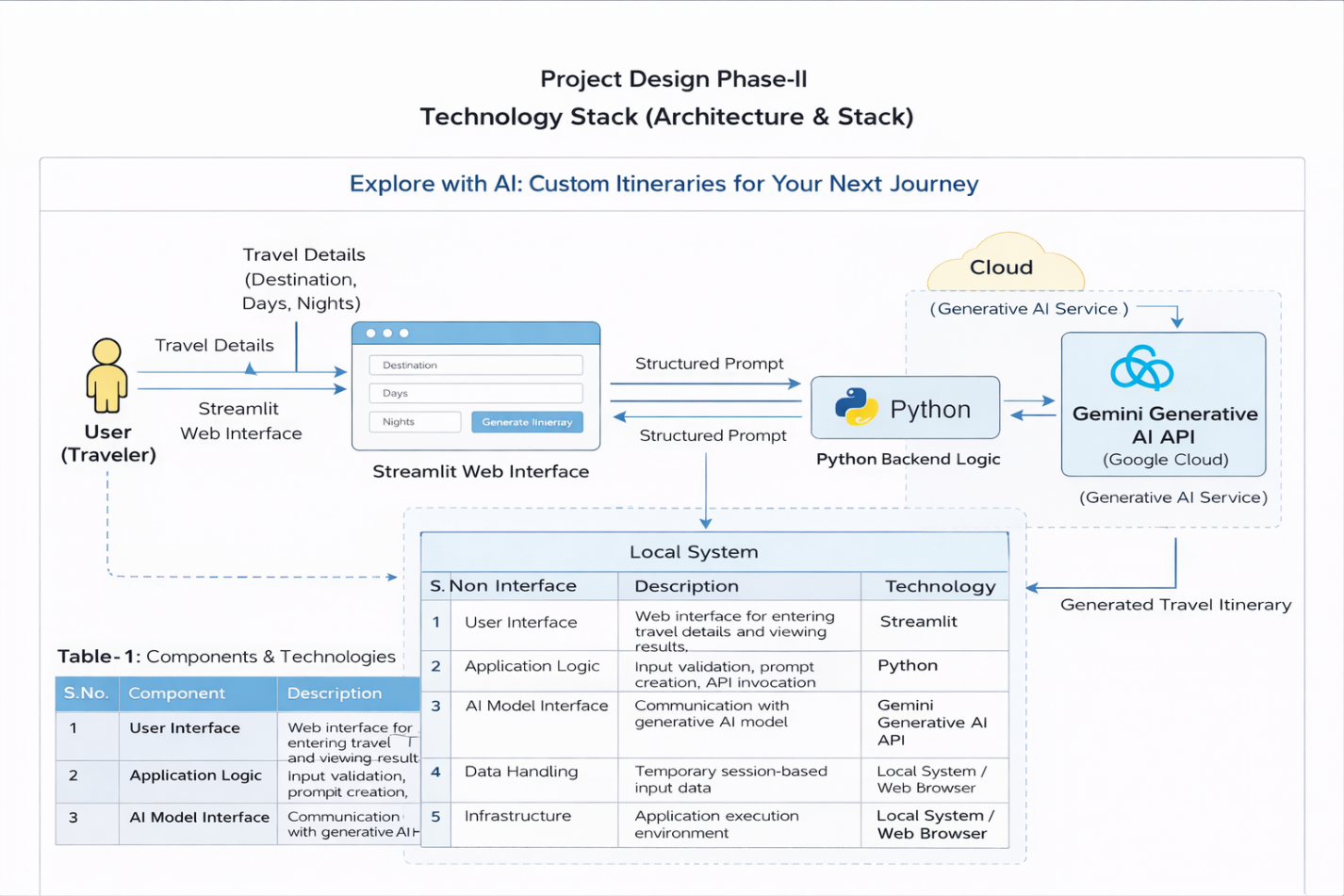
| **User Type** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Acceptance criteria** | **Priority** | **Release** |
| --- | --- | --- | --- | --- | --- | --- |
| Customer (Web user) | Travel Itinerary Generation | USN-1 | As a user, I can enter my destination, number of days, and number of nights to generate a personalized travel itinerary. | The system generates a structured travel itinerary based on the entered details. | High | Sprint-1 |
| Customer (Web user) | Input Validation | USN-2 | As a user, I want the system to validate my input details before generating the itinerary. | If inputs are invalid, the system displays an appropriate error message. | High | Sprint-1 |
| Customer (Web user) | AI Integration | USN-3 | As a user, I want the system to use generative AI to create a detailed travel itinerary automatically. | The AI generates a day-wise itinerary including activities and travel tips. | High | Sprint-2 |
| Customer (Web user) | Output Display | USN-4 | As a user, I want to view the generated itinerary clearly on the web interface. | The itinerary is displayed in a readable format in the application. | Medium | Sprint-2 |
| Customer (Web user) | Error Handling | USN-5 | As a user, I want the system to handle API or runtime errors gracefully. | If an error occurs, the system displays a user-friendly error message without crashing. | Medium | Sprint-3 |

**3.3. Technology Stack**

**Technical Architecture:**

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2

**Reference:** [**https://developer.ibm.com/patterns/ai-powered-backend-system-for-order-processing-during-pandemics/**](https://developer.ibm.com/patterns/ai-powered-backend-system-for-order-processing-during-pandemics/)

****

**Table-1 : Components & Technologies:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
| 1 | User Interface | Web interface for entering travel details and viewing results | Streamlit |
| 2 | Application Logic | Input validation, prompt creation, API invocation | Python |
| 3 | AI Model Interface | Communication with generative AI model | Google Generative AI API (Gemini) |
| 4 | Data Handling | Temporary session-based input data | Streamlit session handling |
| 5 | External API | Generative AI service for itinerary creation | Gemini Flash Model |
| 6 | Infrastructure | Application execution environment | Local System / Web Browser |

**Table-2: Application Characteristics:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristic** | **Description** | **Technology** |
| 1 | Open-Source Frameworks | Used for UI and backend development | Python, Streamlit |
| 2 | Security Implementations | API key protection and input validation | Secure API configuration, Input Validation |
| 3 | Scalable Architecture | Easily extendable to include new features | Modular Python Architecture |
| 4 | Availability | Accessible when application server is running | Streamlit Server |
| 5 | Performance | Real-time AI response generation | Optimized API call handling |

**References:**

[**https://c4model.com/**](https://c4model.com/)

[**https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/**](https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/)

[**https://www.ibm.com/cloud/architecture**](https://www.ibm.com/cloud/architecture)

[**https://aws.amazon.com/architecture**](https://aws.amazon.com/architecture)

[**https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d**](https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d)

1. **PROJECT DESIGN**

**4.1. Problem Solution Fit**

The **Problem–Solution Fit** for *Explore with AI* focuses on addressing the common challenges faced by travelers in planning personalized trips efficiently and without stress.

**Purpose:**

❑ To solve the problem of time-consuming and manual travel planning by providing an AI-powered automated solution.

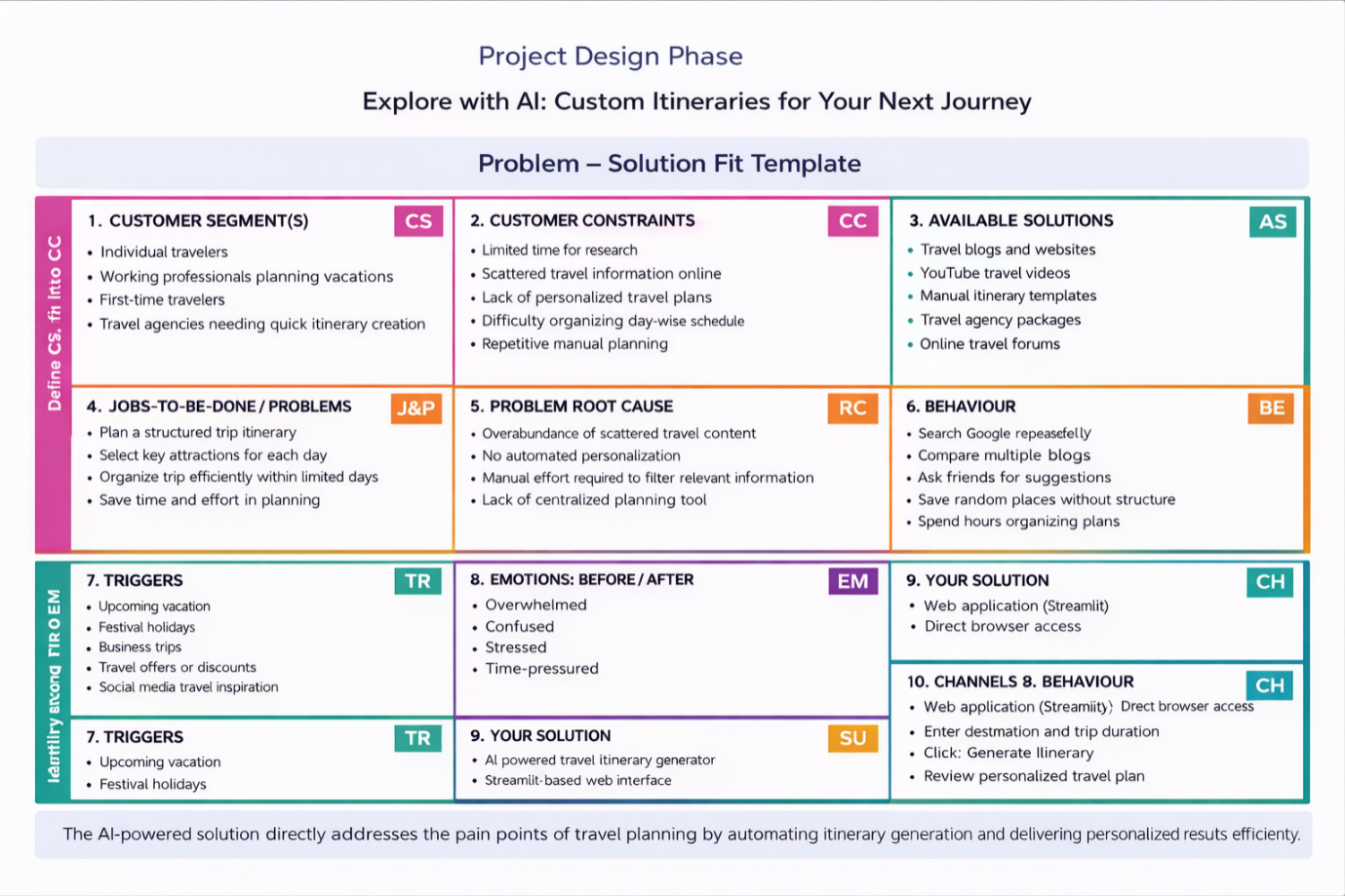
❑ To improve solution adoption by aligning with modern user behavior, where users prefer quick, web-based, and personalized digital tools.

❑ To clearly communicate the value of AI-driven itinerary generation as a faster and more efficient alternative to traditional travel planning methods.

❑ To reduce frequent user frustrations such as scattered travel information, lack of personalization, and repetitive research efforts.

❑ To understand traveler needs and improve the planning experience by delivering structured, day-wise, and customized travel itineraries.

**Template:**



References:

1. <https://www.ideahackers.network/problem-solution-fit-canvas/>
2. <https://medium.com/@epicantus/problem-solution-fit-canvas-aa3dd59cb4fe>

**4.2. Proposed Solution**

Project team shall fill the following information in the proposed solution template.

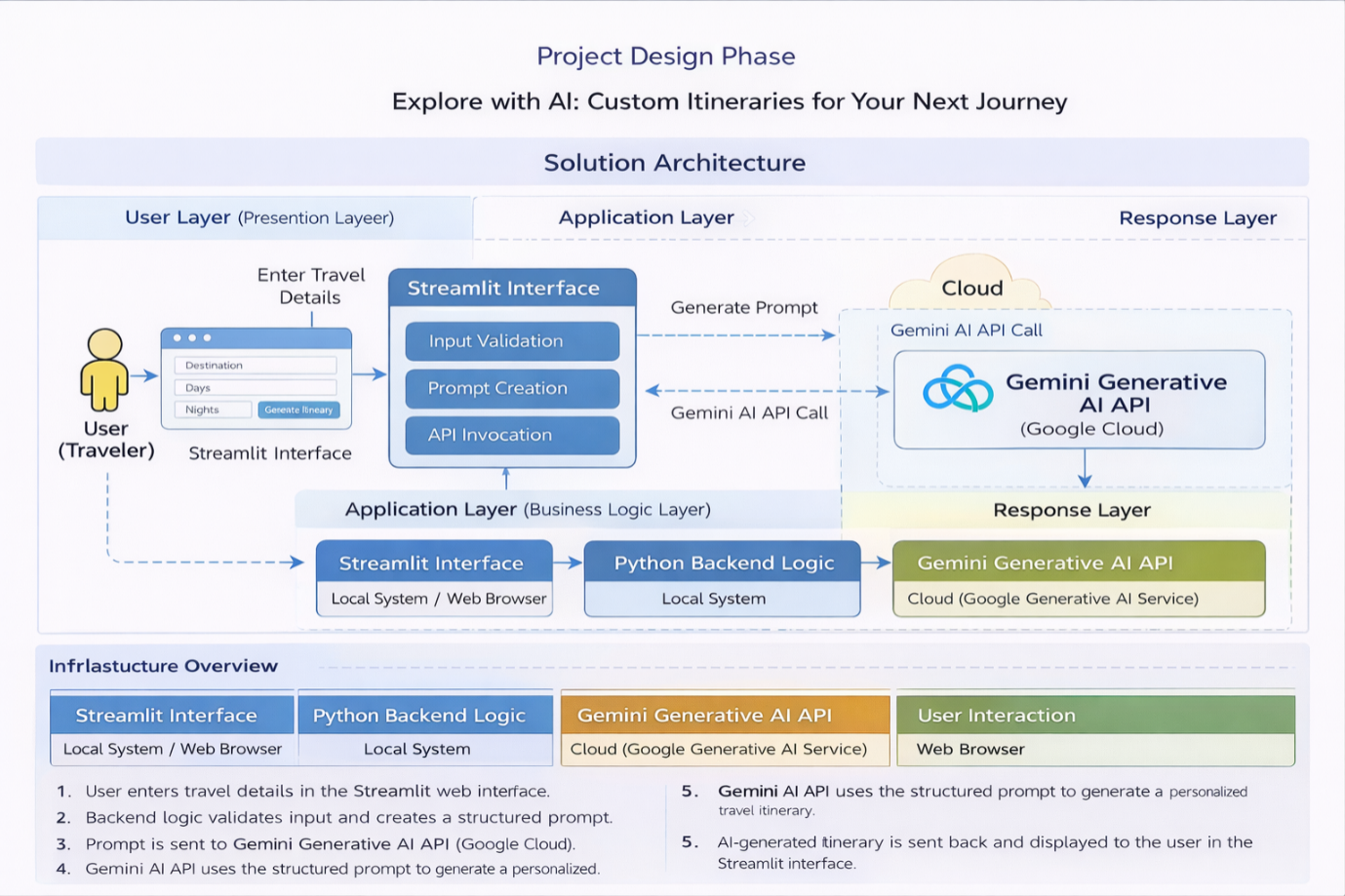
|  |  |  |
| --- | --- | --- |
| **S.No** | **Parameter** | **Description** |
| 1 | **Problem Statement (Problem to be solved)** | Travelers face difficulty in planning personalized trips due to scattered information, time constraints, and manual research efforts. Organizing a structured day-wise itinerary requires significant effort and lacks automation. |
| 2 | **Idea / Solution Description** | Develop an AI-powered web application that generates personalized travel itineraries based on user inputs such as destination, number of days, and number of nights. The system uses a pre-trained generative AI model to create structured and detailed travel plans instantly. |
| 3 | **Novelty / Uniqueness** | The solution integrates generative AI to automate itinerary creation, providing personalized and structured travel plans instantly. Unlike traditional blogs or templates, the system dynamically generates customized itineraries based on user-specific inputs. |
| 4 | **Social Impact / Customer Satisfaction** | The solution reduces stress and saves time for travelers by simplifying trip planning. It improves user experience by offering organized and personalized travel plans, enhancing overall travel satisfaction. |
| 5 | **Business Model (Revenue Model)** | The platform can adopt a freemium model, offering basic itinerary generation for free and premium features such as advanced customization, downloadable PDF export, and travel recommendations through subscription plans or advertisements. |
| 6 | **Scalability of the Solution** | The system can be scaled by deploying it on cloud platforms, integrating additional APIs (weather, hotel booking), adding multi-language support, and handling increased user traffic through cloud infrastructure. |

**4.3 Solution Architecture**

The solution architecture of *Explore with AI* bridges the gap between the problem of manual travel planning and the implementation of an AI-powered automated system.

The architecture is designed to be lightweight, modular, and scalable. It integrates a web-based user interface with backend application logic and a cloud-based generative AI service.

**Solution Architecture Diagram:**

****

*Figure 1: Architecture and data flow of travel itinerary application*

**Reference:** [**https://aws.amazon.com/blogs/industries/voice-applications-in-clinical-research-powered-by-ai-on-aws-part-1-architecture-and-design-considerations/**](https://aws.amazon.com/blogs/industries/voice-applications-in-clinical-research-powered-by-ai-on-aws-part-1-architecture-and-design-considerations/)

1. **PROJECT PLANNING & SCHEDULING**

**5.1 Project Planning**

**Product Backlog, Sprint Schedule, and Estimation (4 Marks)**

Use the below template to create product backlog and sprint schedule

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Story Points** | **Priority** | **Team Members** |
| Sprint-1 | Project Setup | USN-1 | Set up Python environment and install required libraries | 2 | High | S.Afrid |
| Sprint-1 | Project Setup | USN-2 | Configure Gemini API securely | 3 | High | P.Sruthi |
| Sprint-1 | UI Development | USN-3 | Design Streamlit interface for travel details input | 3 | High | K.Jagadesh Reddy |
| Sprint-1 | UI Development | USN-4 | Create input fields (destination, days, nights) | 2 | High | M.Prathima |
| Sprint-1 | UI Development | USN-5 | Add “Generate Itinerary” button | 2 | High | S.Afrid |
| Sprint-2 | Input Validation | USN-6 | Validate destination input | 2 | High | P.Sruthi |
| Sprint-2 | Input Validation | USN-7 | Validate number of days and nights | 2 | High | K.Jagadesh Reddy |
| Sprint-2 | Prompt Engineering | USN-8 | Create structured prompt for AI model | 3 | High | M.Prathima |
| Sprint-2 | AI Integration | USN-9 | Integrate Gemini Generative AI API | 5 | High | S.Afrid, K.Jagadesh Reddy |
| Sprint-2 | Output Handling | USN-10 | Display generated itinerary in readable format | 3 | Medium | P.Sruthi |
| Sprint-2 | Error Handling | USN-11 | Implement try-except for API errors | 2 | Medium | M. Prathima |

**Project Tracker, Velocity & Burndown Chart: (4 Marks)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Total Story Points** | **Duration** | **Sprint Start Date** | **Sprint End Date (Planned)** | **Story Points Completed** | **Sprint Release Date** |
| Sprint-1 | 12 | 7 Days | 01 Feb 2026 | 07 Feb 2025 | 12 | 07 Feb 2026 |
| Sprint-2 | 19 | 7 Days | 08 Feb 2026 | 14 Feb 2026 | 19 | 14 Feb 2026 |

* **Velocity Calculation**

Total Story Points = 12 + 19 = 31

Number of Sprints = 2

Velocity = 31 / 2

= 15.5 ≈ 16 Story Points per Sprint

* **Average Velocity per Day**

If sprint duration = 7 days

Velocity per sprint = 16 story points

Average Velocity per day = 16 / 7

≈ 2.3 Story Points per Day

**Burndown Chart:**

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile[software development](https://www.visual-paradigm.com/scrum/what-is-agile-software-development/) methodologies such as [Scrum](https://www.visual-paradigm.com/scrum/scrum-in-3-minutes/). However, burn down charts can be applied to any project containing measurable progress over time.

[**https://www.visual-paradigm.com/scrum/scrum-burndown-chart/**](https://www.visual-paradigm.com/scrum/scrum-burndown-chart/)

[**https://www.atlassian.com/agile/tutorials/burndown-charts**](https://www.atlassian.com/agile/tutorials/burndown-charts)

**Reference:**

[**https://www.atlassian.com/agile/project-management**](https://www.atlassian.com/agile/project-management)

[**https://www.atlassian.com/agile/tutorials/how-to-do-scrum-with-jira-software**](https://www.atlassian.com/agile/tutorials/how-to-do-scrum-with-jira-software)

[**https://www.atlassian.com/agile/tutorials/epics**](https://www.atlassian.com/agile/tutorials/epics)

[**https://www.atlassian.com/agile/tutorials/sprints**](https://www.atlassian.com/agile/tutorials/sprints)

[**https://www.atlassian.com/agile/project-management/estimation**](https://www.atlassian.com/agile/project-management/estimation)

[**https://www.atlassian.com/agile/tutorials/burndown-charts**](https://www.atlassian.com/agile/tutorials/burndown-charts)

1. **FUNCTIONAL AND PERFORMANCE TESTING**

## **6.1 Performance Testing**

## **Test Scenarios & Results**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Scenario (What to test)** | **Test Steps (How to test)** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| FT-01 | Destination Input Validation | Leave destination empty and click “Generate” | System should show validation error | Error message displayed correctly | Pass |
| FT-02 | Days Input Validation | Enter 0 or negative number of days | System should reject invalid input | Validation message shown | Pass |
| FT-03 | Nights Input Validation | Enter negative value for nights | System should reject invalid input | Validation message shown | Pass |
| FT-04 | AI Content Generation | Enter valid destination, days, nights and click Generate | System generates structured travel itinerary | Itinerary generated correctly | Pass |
| FT-05 | API Connection Check | Verify API key configuration and trigger generation | API should respond successfully | API response received | Pass |
| FT-06 | Error Handling | Temporarily disconnect internet and test generation | System should show user-friendly error | Error handled gracefully | Pass |

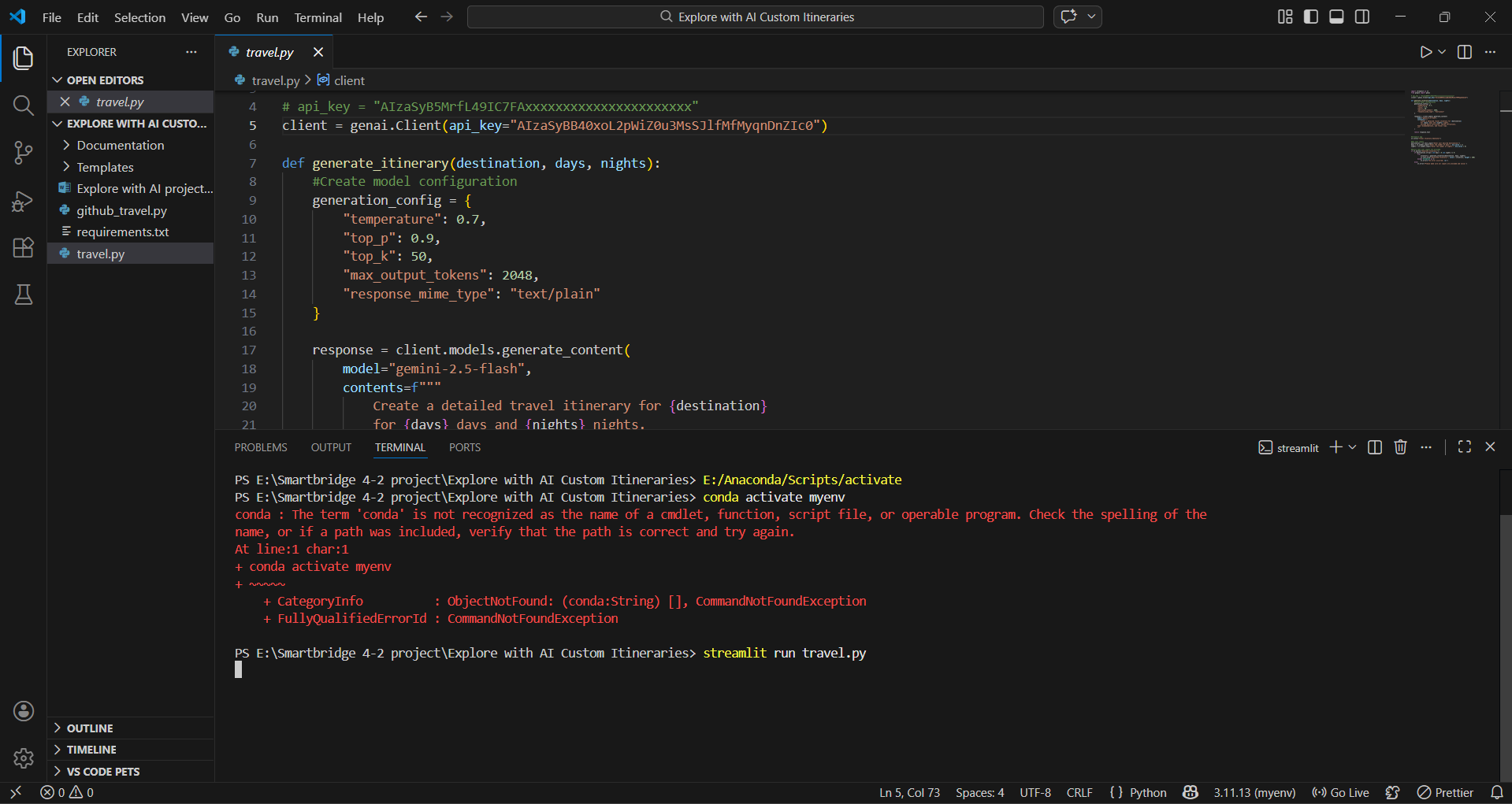
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| PT-01 | Response Time Test | Generate itinerary and measure response time | Should respond within acceptable time (<5 seconds depending on API) | | Response time within limit | Pass |
| PT-02 | Multiple Request Handling | Generate itineraries consecutively | System should process without crashing | Handled smoothly | | Pass |
| PT-03 | Large Input Handling | Enter long destination name and maximum days | System should still generate itinerary properly | Output generated correctly | | Pass |

1. **RESULTS**

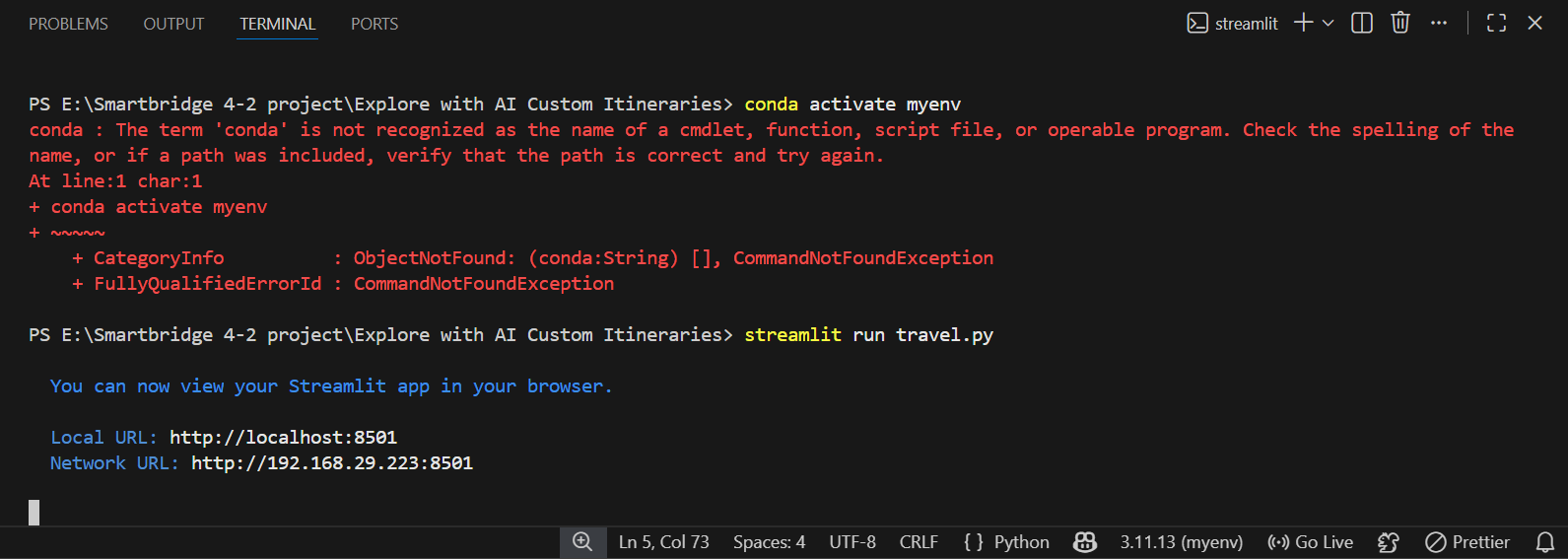
**7.1. Output Screenshots**

The complete execution of Explore with AI: Custom Itineraries for Your Next Journey application is represented step by step in the following screenshots.

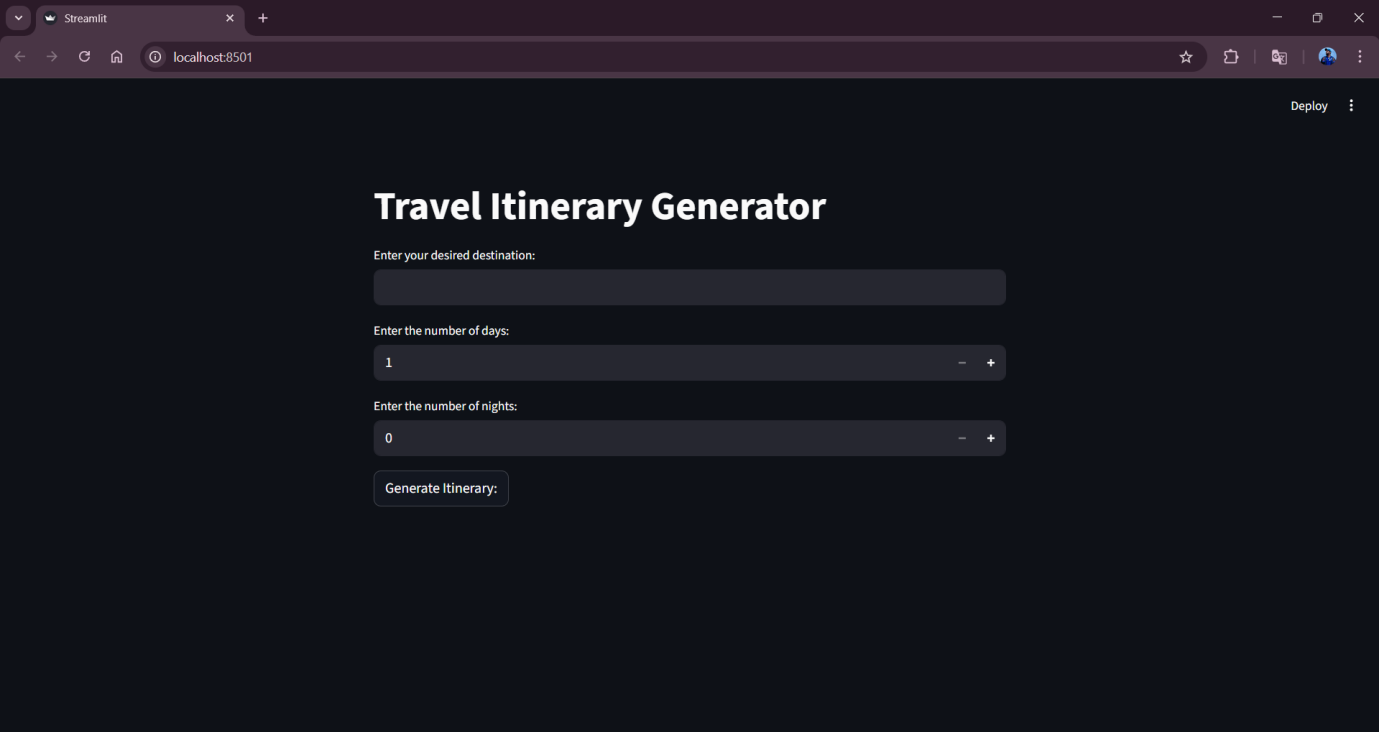
**Step 1:** To run the Streamlit Application we have to use the command streamlit run travel.py in the terminal in path where the travel.py file is located.

****

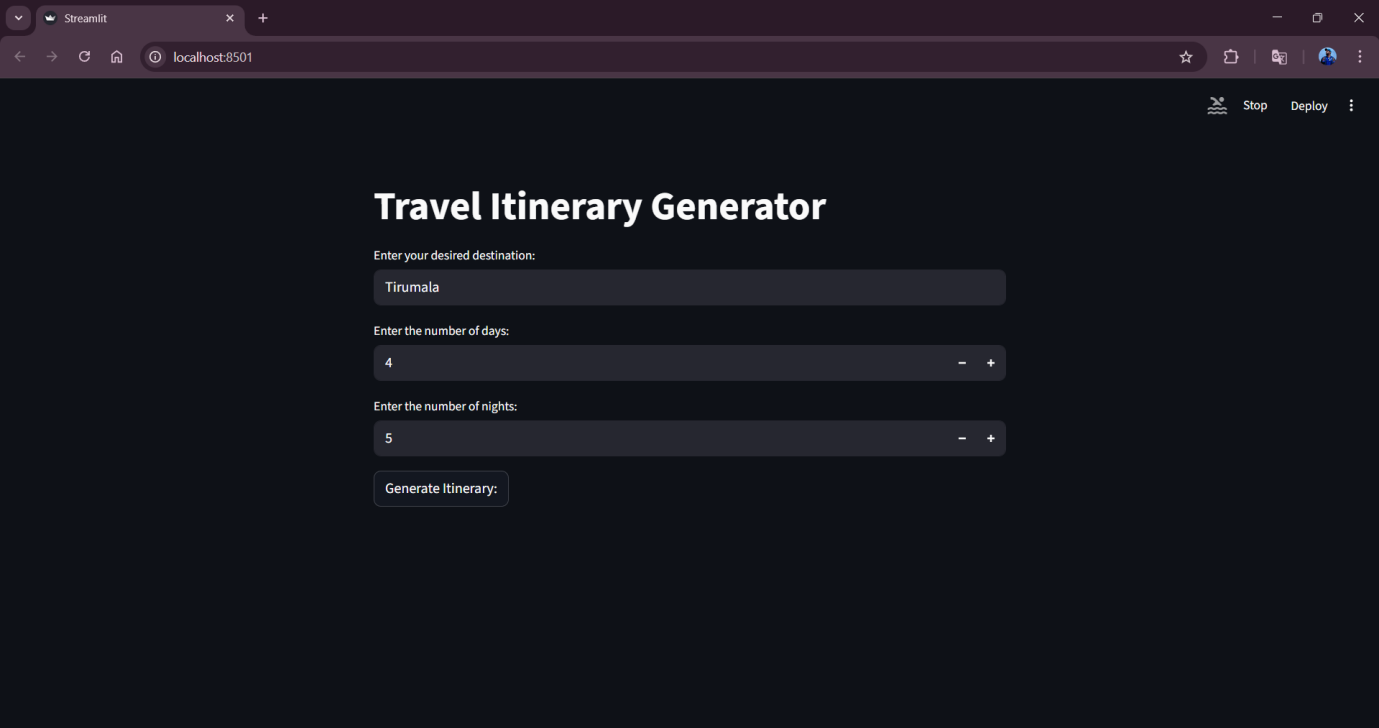
**Step 2:** After running the command in terminal, the code will get executed and the webpage will open directly. Another way to open webpage is that a localhost link will get generated in the terminal, we can access the webpage using that link.



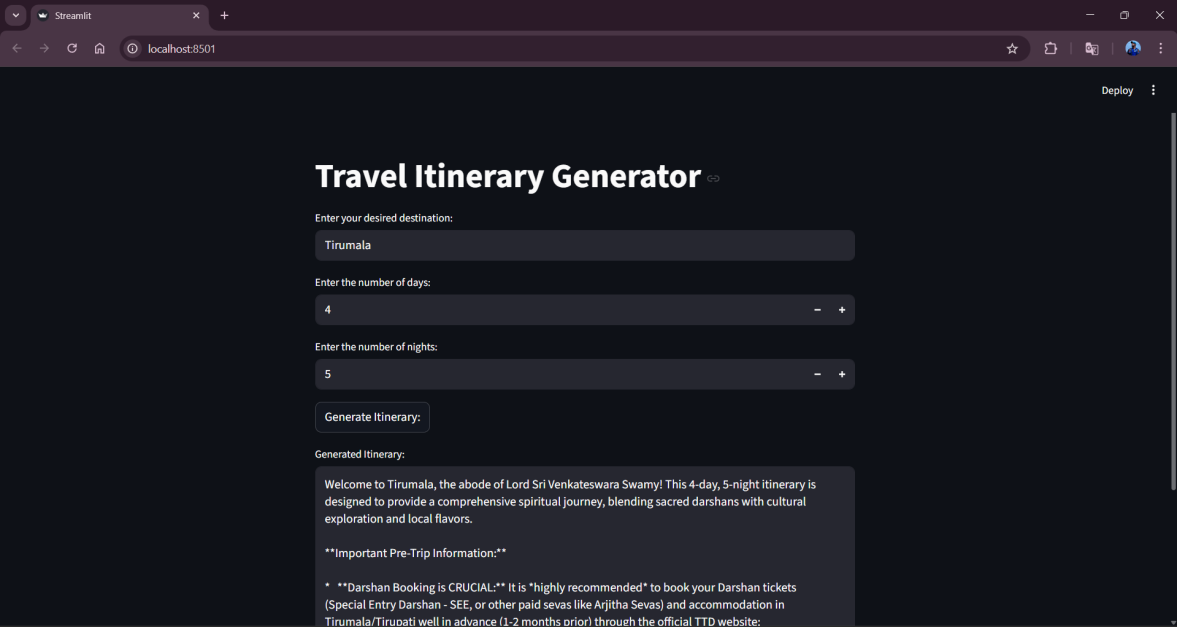
**Step 3:** The Streamlit webpage opens as shown in the figure given below. This is an automated webpage. No secondary HTML codes required to build this webpage. Python code itself consists the webpage building code.



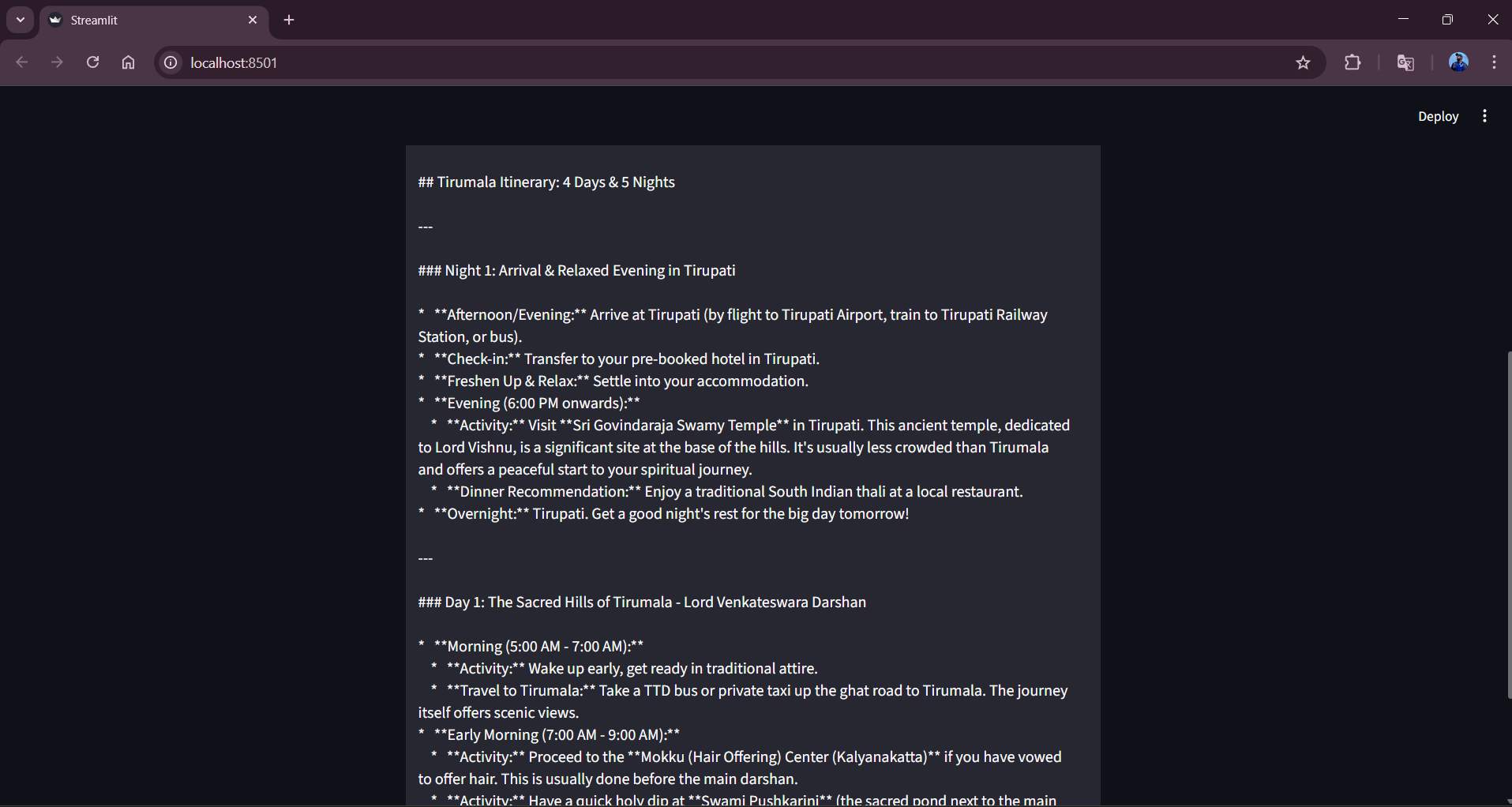
**Step 3:** The user has to give inputs in the website such as desired Destination Name, Number of Days and Number of Nights. The Number of Days and Nights mean in how many days and nights, the user wants their travel planning to be made. After entering the required details, the user should click on Generate Itinerary button to generate the travel plan. Here I chose the destination as Tirumala, days as 4 and nights as 5.

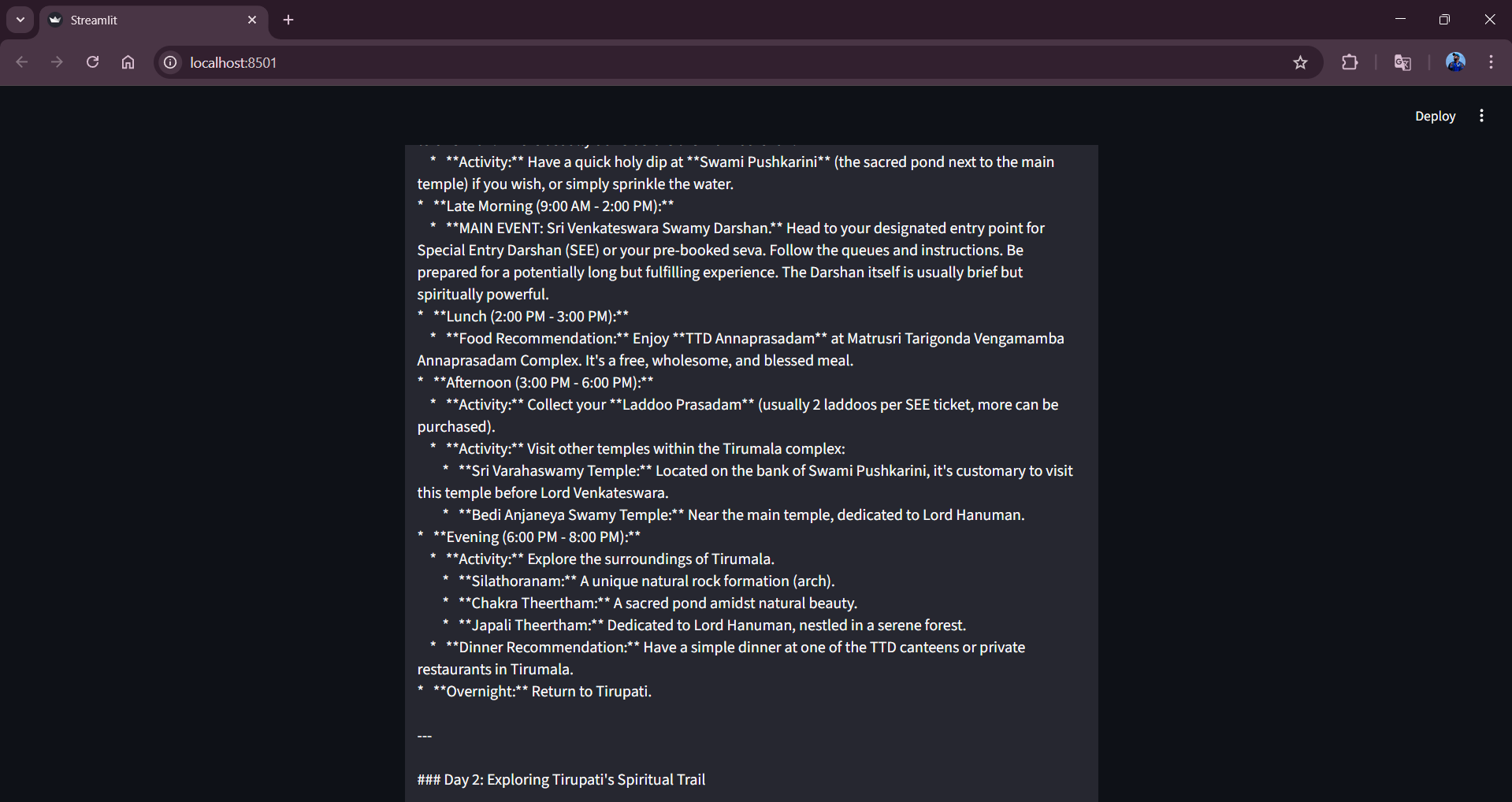


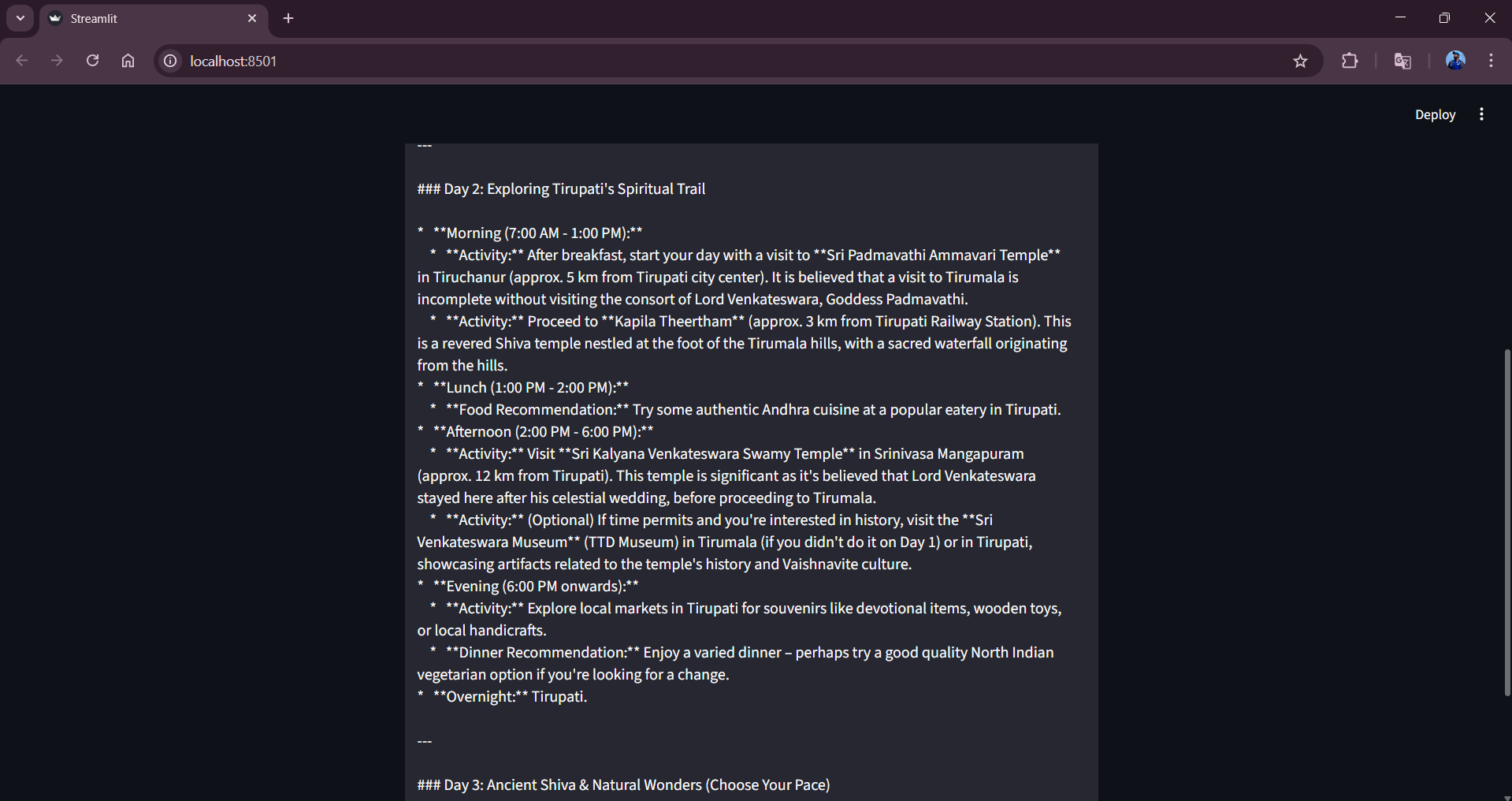
**Step 4:** After clicking the Generate Itinerary button,in fraction of seconds the itinerary will be generated based on the user input as shown in the following images.

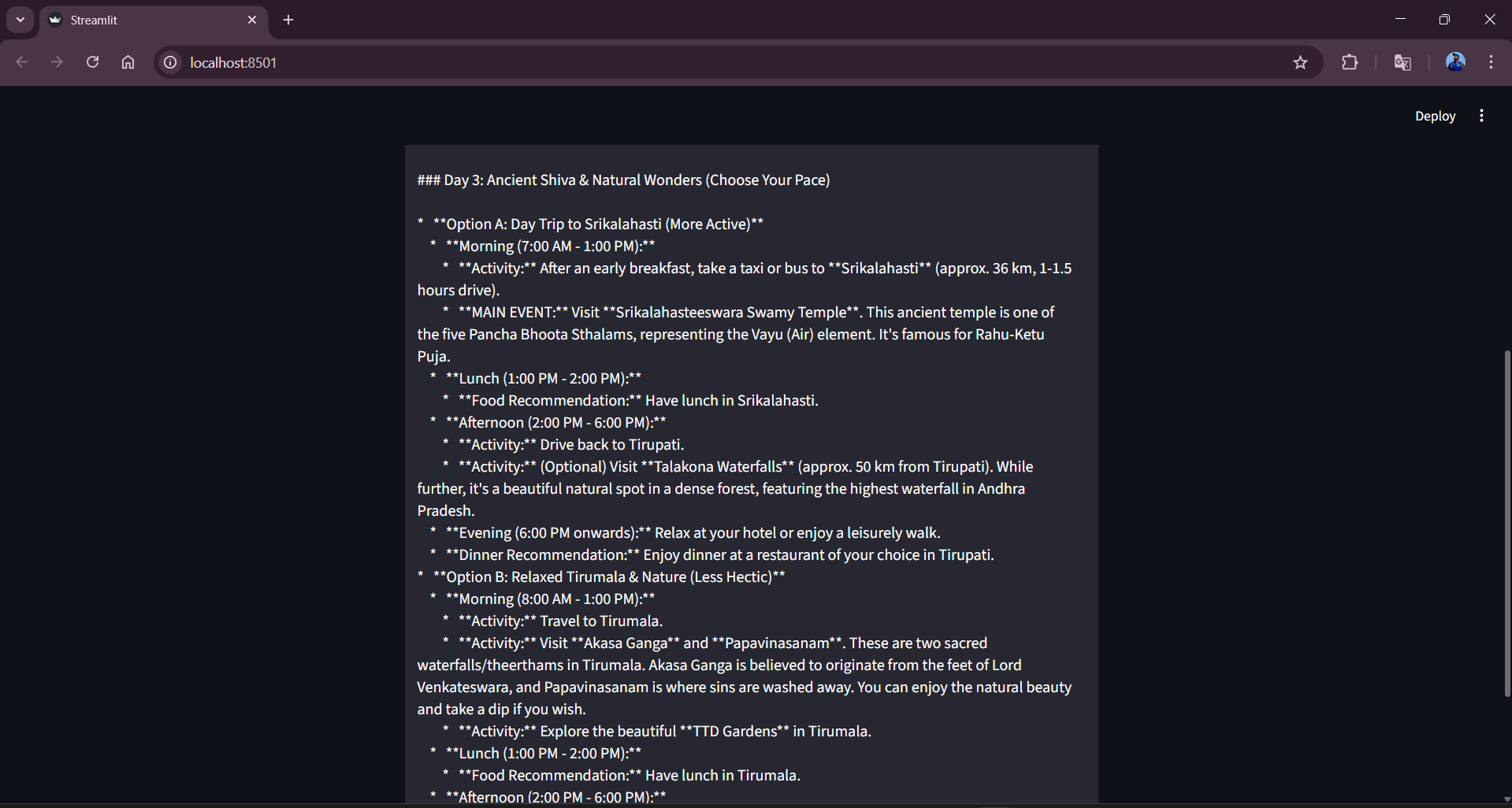


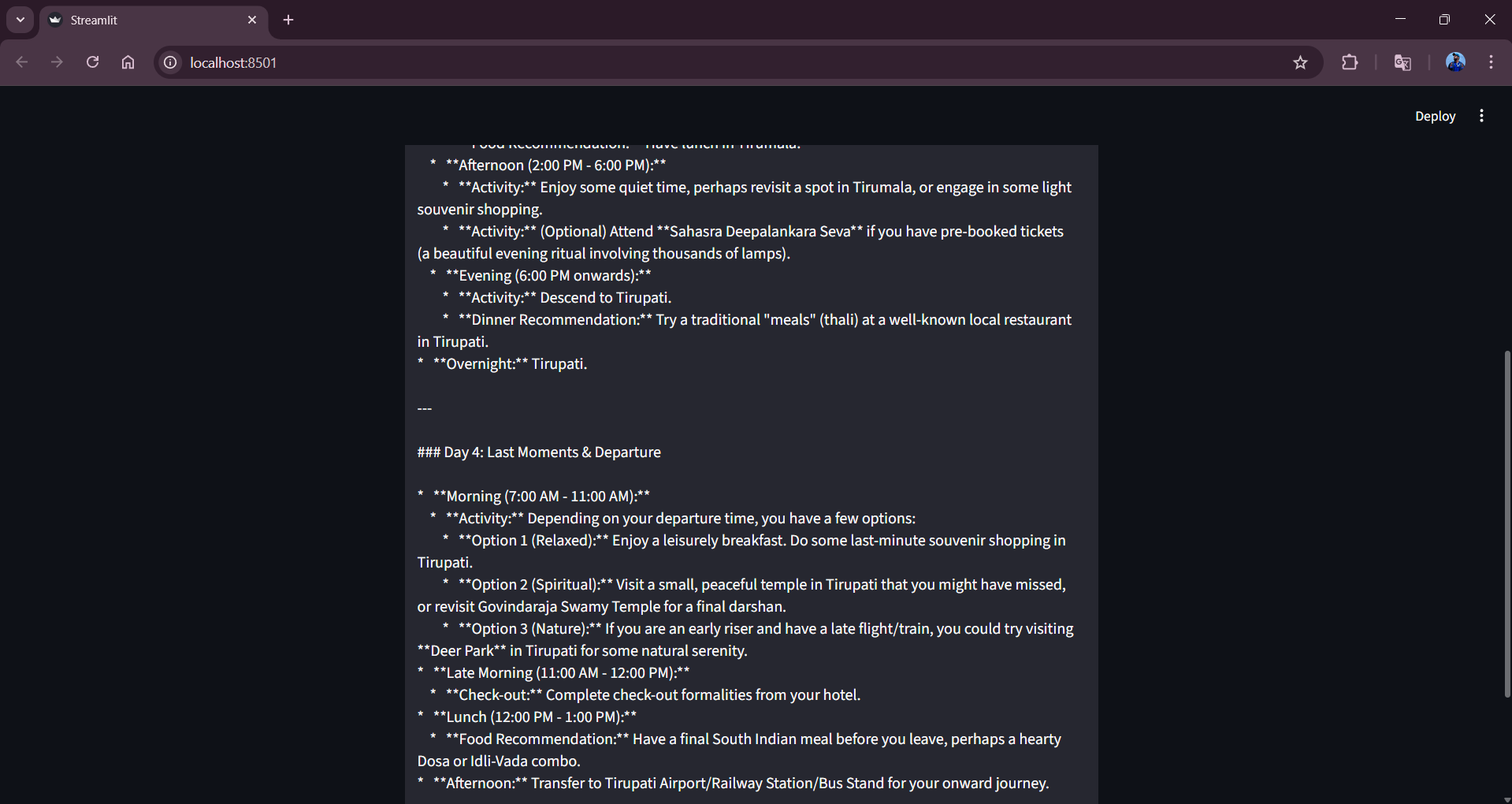


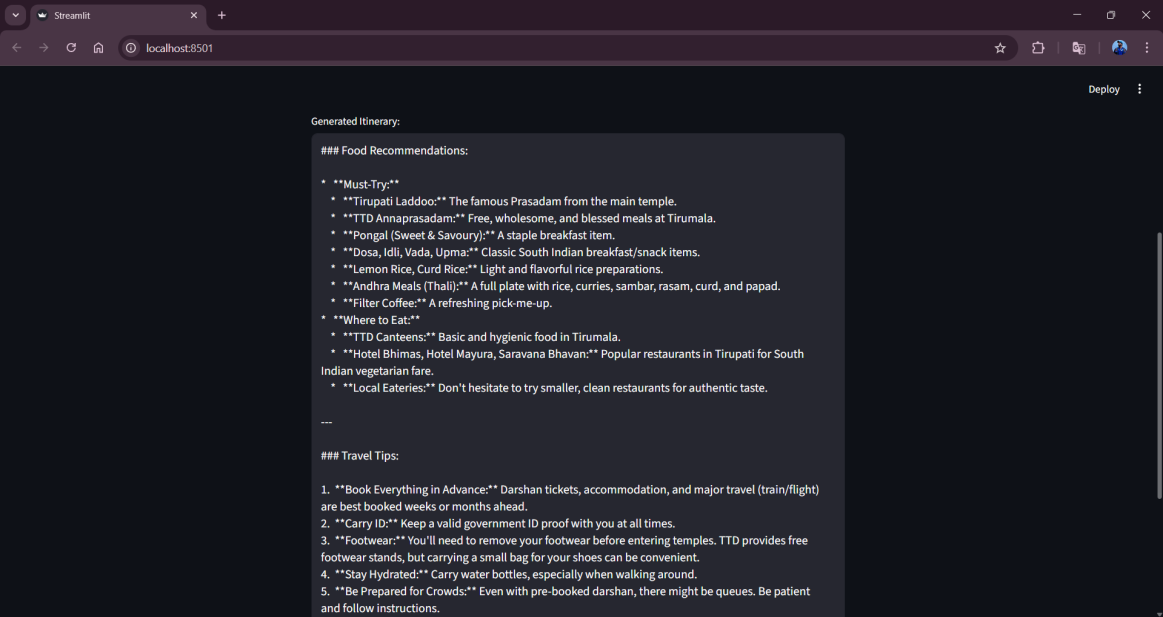


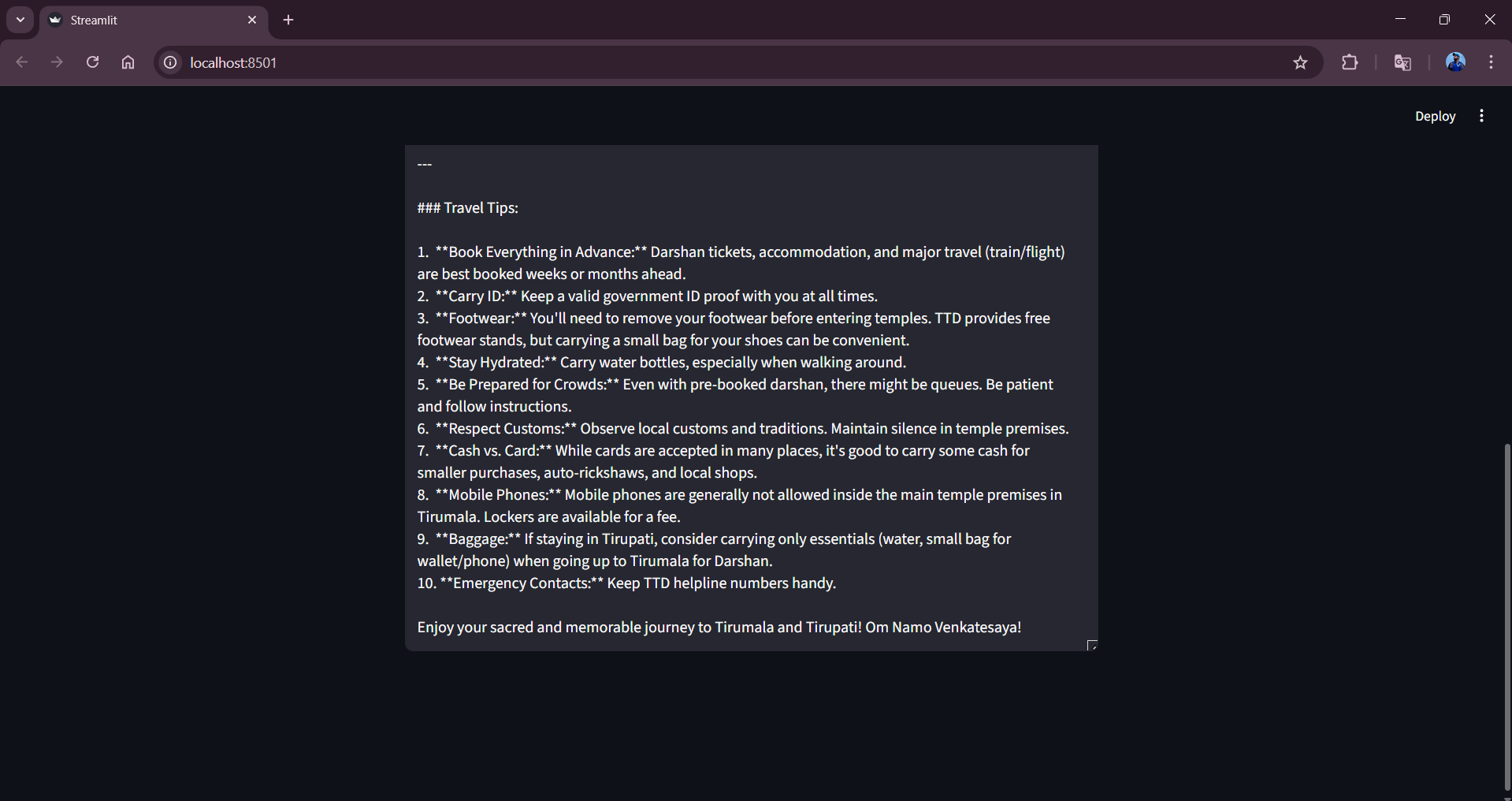












**8. ADVANTAGES AND DISADVANTAGES**

**Advantages**

* Automates travel itinerary creation, saving time and effort for users.
* Generates personalized travel plans based on user preferences and trip duration.
* Easy-to-use interface built with Streamlit, suitable for non-technical users.
* Uses a pre-trained generative AI model, eliminating the need for model training.
* Can be extended to support travel agencies and content creators.

**Disadvantages**

* Depends on internet connectivity and availability of the AI API.
* Generated itineraries may not always reflect real-time travel conditions.
* Limited customization options in the current version of the application.
* Does not include booking, cost estimation, or real-time availability features.
* API usage may incur cost limitations based on usage quotas.

**9. CONCLUSION**

The *Explore with AI* project demonstrates how generative AI can be used to simplify travel planning by automatically generating personalized travel itineraries. The application integrates a pre-trained AI model with a Streamlit interface to provide quick and user-friendly itinerary generation. This project highlights the practical use of AI in real-world applications and serves as a foundation for future enhancements in intelligent travel planning. The developed application serves as a functional prototype that can be further expanded with additional features, making it a valuable tool for travelers, travel agencies, and content creators.

**10. FUTURE SCOPE**

The application can be enhanced by allowing users to specify additional preferences such as budget, travel style, and accommodation type. Real-time data such as weather updates, local events, and travel advisories can be integrated for more accurate itineraries. Booking features for hotels, transportation, and tourist attractions can be added to make the system more comprehensive. Multi-language support can be introduced to serve users from different regions. The system can be deployed on cloud platforms and scaled to support a larger number of users.

**11. APPENDIX**

**11.1. Source Code**

The source code for the Explore with AI: Custom Itineraries for Your Next Journey project includes the implementation of the Streamlit user interface, integration of the Gemini Flash model using the Google Generative AI API, travel plan/ itinerary generation logic, and plan upgrading based on the number of days and nights given by the user feature. The code is written in Python and follows a modular and readable structure.

**11.2. Github & Project Demo Link**

* **Demo Video Link:**

<https://drive.google.com/file/d/1nyrbCrYjuLKzrgCEq3VMfWrk_3KKTpHM/view?usp=drive_link>

* **Github Repository:**

<https://github.com/>NarasimhaRoyal/Explore-with-AI-Custom-Itineraries-For-Your-Next-Journey