VIETNAM NATIONAL UNIVERSITY
HO CHI MINH CITY
UNIVERSITY OF SCIENCE
FACULTY OF INFORMATION TECHNOLOGY

REPORT
FINAL

Smart Tourism System

CSC10014 - Computational Thinking

Team members:

24127457 - Nguyen Ho Nam
24127053 - Tran Hoang Minh Khang
24127575 - Nguyen Thi Hong Truc
24127367 - Dang Trung Hien
24127561 - Dang The Tony
24127402 - Phung Quoc Huy
24127324 - Vu Thi Anh

Instructor:

Theoretical Teacher: Dr. Chau Thanh Duc
Laboratory Teacher: Do Duc Hao
Tran Hoang Quan

Ho Chi Minh City, December 16, 2025

Contents

Chapter 1

Introduction

This chapter presents the background for the development of the Smart Tourism System, clarifies the necessity of building such a system, defines the technical objectives of the project, identifies the target users and implementation scope, and analyzes existing solutions and related technologies to highlight the practicality of the proposed system.

1.1 Problem Statement

In the era of digital technology, tourism information has become increasingly abundant, diverse, and rapidly changing. However, this information overload often makes it difficult for users to select destinations and plan suitable travel itineraries. Some common challenges faced by tourists include:

- **Information Overload**: Users must search through hundreds of articles, videos, and reviews before making decisions.
- **Difficulty in personalizing travel plans**: Each traveler has different preferences, time constraints, and budgets, making it challenging to create an optimal itinerary.
- **Lack of integration between services**: Information about attractions, hotels, and food is often fragmented and not organized within a unified intelligent system.
- **Limited distance-based support**: Finding suitable hotels near tourist attractions (within 10 km) is still largely a manual process.

These limitations indicate a clear need for an intelligent system capable of processing data, analyzing user preferences, and automatically recommending suitable options.

1.2 Project Objectives

Based on the identified problems, the project titled *"Smart Tourism System"* is developed with the following objectives:

- **Develop a tourist destination recommendation system** based on user preferences and behavior.
- **Implement food recommendation functionality** according to taste preferences and cuisine types.
- **Design a hotel recommendation module near tourist attractions** within a radius of less than 10 km.
- **Build a smart itinerary generation feature** based on time constraints, user interests, and geographical location.
- **Model and implement recommendation algorithms** using computational thinking approaches.
- **Design an intuitive and user-friendly interface** to enhance user experience.

These objectives are designed to directly address the pain points of travelers and clearly reflect the technical requirements of the project.

1.3 Target Users

The system is designed for multiple user groups who require fast and personalized travel planning, including:

- **Independent travelers**: who want to plan their trips themselves but have limited time.
- **Young travel groups**: who enjoy exploring new destinations and cuisines.
- **Families with children**: who need optimized itineraries and minimized travel time.
- **Business travelers**: who prioritize convenience, nearby hotels, and suitable dining locations.

These user groups have a strong demand for an intelligent, simple, and efficient travel support tool.

1.4 Scope of Implementation

The implementation scope of the project includes:

In-scope features:

- Tourist destination recommendations based on user preferences.
- Food recommendations according to preferred cuisine types.
- Hotel recommendations within a 10 km radius of tourist attractions.
- Automatic generation of basic travel itineraries.
- Design of a demo interface simulating system operations.

Out-of-scope features:

- Integration with real booking or ticketing APIs.
- Processing real-time data from major tourism platforms.
- Development of a full-featured mobile application.
- Real-time GPS navigation support.

Defining the scope helps ensure that the project remains focused on core functionalities and aligns with the available development timeline.

1.5 Overview of Related Solutions and Technologies

Currently, there are several smart tourism applications available on the market, such as Google Travel, TripAdvisor, Klook, and Traveloka. However, each platform has its own limitations:

- **Google Travel**: Strong in mapping and reviews but lacks deep personalization.
- **TripAdvisor**: Rich in community reviews but does not support preference-based recommendations.
- **Klook**: Focuses on tour booking but does not provide smart itinerary planning.
- **Traveloka**: Supports booking services but lacks recommendations for attractions and food.

The proposed system differs in the following aspects:

- Strong emphasis on **personalized travel experiences**.
- Integration of multiple modules: attractions, food, hotels, and itineraries.

- High scalability and potential for future AI integration.
- Optimized for general users and independent travelers.

Therefore, the Smart Tourism System has the potential to become a flexible, modern, and intelligent travel support platform aligned with the trend of personalized tourism.

Chapter 2

System Requirements Analysis

This chapter presents a detailed analysis of the system requirements, including functional requirements, non-functional requirements, and interaction modeling between users and the system through Use Case Diagrams and Use Case Specifications.

2.1 Functional Requirements

The Smart Tourism System must satisfy the following functional requirements:

1. **Tourist Destination Recommendation**
    - Allow users to search destinations by keywords.
    - Recommend destinations based on personal preferences (nature, history, culture, etc.).
    - Display detailed information about destinations (description, opening hours, images, etc.).
2. **Food Recommendation Based on Preferences**
    - Allow users to select preferred tastes or cuisine types.
    - Recommend foods or restaurants based on the selected preferences.
3. **Hotel Recommendation Near Tourist Attractions**
    - Allow users to view hotels near the currently selected destination.
    - Display only hotels within a radius of less than 10 km.
4. **Smart Itinerary Creation and Management**
    - Automatically generate itineraries based on user preferences and available time.
    - Allow users to edit the generated itineraries.
5. **Data Management for Admin**

- Admin can add, edit, and delete tourist destinations.
- Admin can manage hotel and food data.

2.2 Non-functional Requirements

1. **Performance**
   - Search response time must be under 2 seconds.
   - The system must handle at least 100 concurrent requests.
2. **Scalability**
   - Easy integration with third-party APIs (Google Maps, Booking, etc.).
   - Easy expansion of destination and hotel databases.
3. **Reliability**
   - The system operates continuously 24/7.
   - The error rate must be below 1%.

2.3 Use Case Modeling

2.3.1 Overall Use Case Diagram

The system consists of two main actors:

- **User (Tourist)**: Uses recommendation and itinerary features.
- **Admin**: Manages destination, food, and hotel data.

Main Use Case groups include:

- Destination search and recommendation.
- Food recommendation based on preferences.
- Hotel recommendation near destinations.
- Itinerary creation and management.

2.3.2 Use Case Specifications

Use Case 1: Destination Recommendation Based on Preferences

Objective: Personalize the destination list according to user preferences.

Actor: User

Brief Description: The user selects personal preferences (nature, culture, history, entertainment, etc.). The system analyzes the data and displays the most suitable destinations.

Preconditions:

- The user has selected a province on the Vietnam map interface.
- The user has configured a preference profile or selected preferences at the current step.

Postconditions:

- A list of suitable destinations is displayed.

Main Flow:

1. The user selects a province on the Vietnam map interface.
2. The system displays a list of available preferences.
3. The user selects desired preferences.
4. The system analyzes destination data.
5. The system returns a list of suitable destinations.

Alternative Flows:

- No preference selected → The system displays all destinations by default.
- No suitable destination found → The system displays a "No results found" message.

Use Case 2: Hotel Recommendation Near Destination (¡10km)

Objective: Assist users in finding suitable hotels near tourist destinations.

Actor: User

Brief Description: The user selects a tourist destination. The system searches for hotels located within a radius of less than 10 km and displays a list with detailed information.

Preconditions:

- The user is viewing the details of a destination.

Postconditions:

- A list of nearby hotels is displayed.

Main Flow:

1. The user selects the "Nearby Hotels" function.
2. The system retrieves the destination location.
3. The system filters hotels within a radius of less than 10 km.
4. The system displays the hotel list.

Alternative Flows:

- No hotels found within the area → The system displays "No nearby hotels found".

Use Case 3: Local Food Recommendation at Destination

Objective: Help users explore local specialty foods at a destination.

Actor: User

Brief Description: The user selects a tourist destination. The system analyzes the user's food preferences to recommend suitable local specialty dishes.

Preconditions:

- The user is viewing the details of a destination.
- The user has selected food preferences.

Postconditions:

- A list of suitable local dishes is displayed.

Main Flow:

1. The user selects the "Food Recommendation" function.
2. The system retrieves the destination location.
3. The system filters food items based on user preferences.
4. The system displays a list of suitable local foods.

Alternative Flows:

- No matching food preferences → The system displays a "No matching food found" message.

Use Case 4: Smart Itinerary Generation

Objective: Automatically generate an optimized travel itinerary.

Actor: User

Brief Description: The user enters available time, preferences, and desired number of destinations. The system generates an optimized itinerary based on recommended destinations.

Preconditions:

- The user has received destination recommendations.
- The user has selected preferences.

Postconditions:

- The itinerary is generated and displayed to the user.

Main Flow:

1. The user selects "Generate Smart Itinerary".
2. The user selects duration (number of days), preferences, and number of destinations.
3. The system analyzes preferences and location data.
4. The system constructs the itinerary.
5. The system displays the itinerary.

Alternative Flows:

- Insufficient destinations to build an itinerary $\rightarrow$ The system displays a warning.
- At least one destination must be selected.

Chapter 3

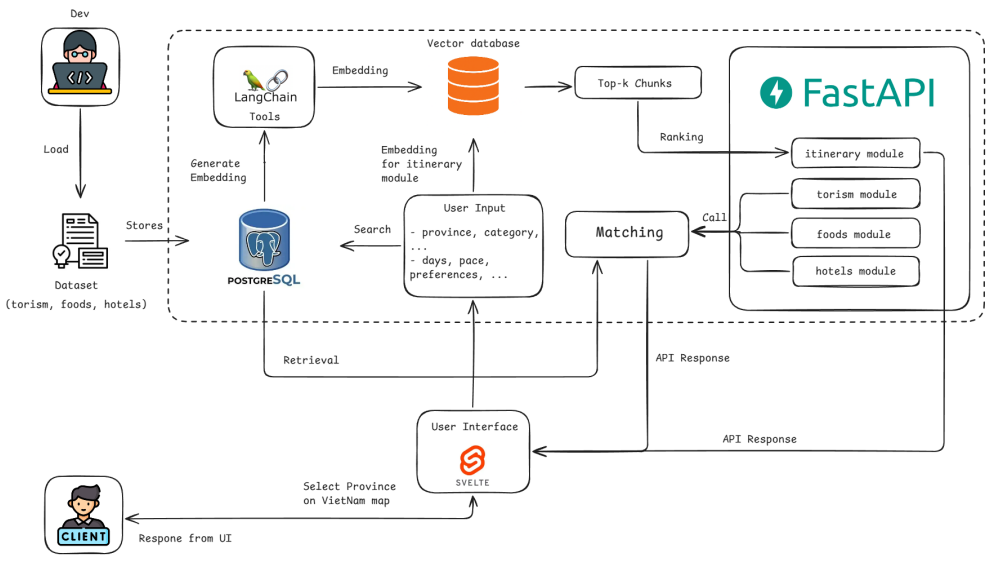
System Design

3.1 Overall System Architecture

The system is designed following a **Client–Server** model with a **modular architecture** to ensure scalability and extensibility (individual modules can be migrated to microservices when necessary).

The main components include:

- **Frontend (Web / Mobile)**: User interface responsible for presentation, user interaction, and API calls to the backend.
- **Backend / API Server**: Provides REST APIs, handles authentication, manages business logic, and orchestrates communication between components.
- **Recommendation Engine**: A dedicated module for recommendation tasks (combining vector search / RAG and rule-based logic).
- **Database (PostgreSQL)**: Stores structured data (Users, Destinations, Hotels, Itineraries) and supports geospatial queries.



3.2 Detailed Design

3.2.1 Class Diagram

The system consists of the following main class groups:

(1) User & UI Group

- **User**: Stores user input information, including: province, category, days, pace, and preferences.
- **UserInterface**: Collects user input and displays system outputs.

(2) Backend API Group (FastAPI)

- Responsible for providing APIs:
  - `getTourismRecommendations()`
  - `getFoodRecommendations()`
  - `getHotelRecommendations()`
  - `getItinerary()`
- Acts as the orchestration layer between system modules.

(3) Recommendation Modules Group

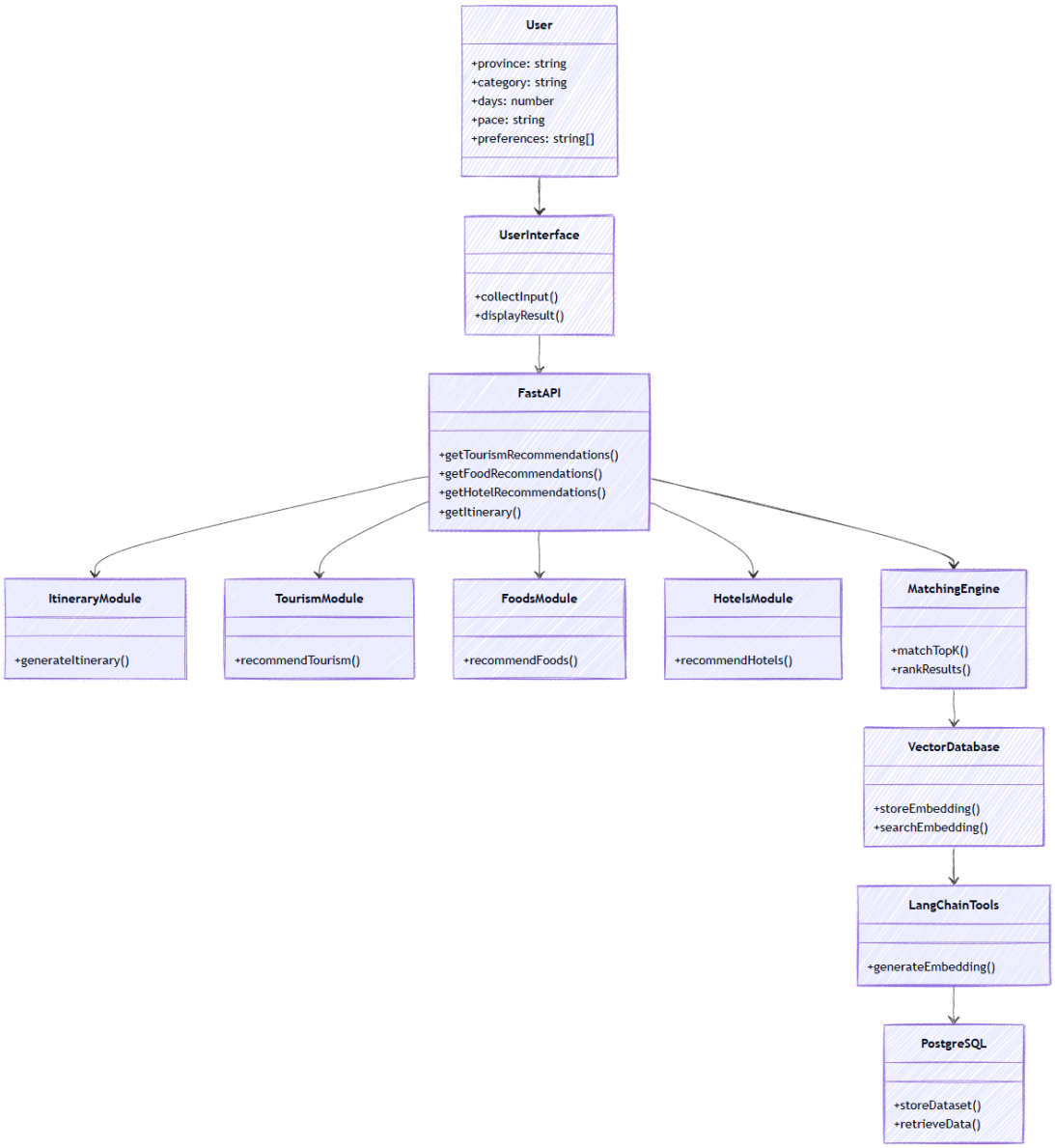
- **TourismModule**: Recommends tourist destinations.
- **FoodsModule**: Recommends foods and restaurants.
- **HotelsModule**: Recommends nearby hotels (within ¡ 10 km).
- **ItineraryModule**: Constructs travel itineraries.

(4) Matching Engine

- Executes `matchTopK()` and `rankResults()` operations.

(5) VectorDB & LangChain Tools

- VectorDB: Stores and searches embeddings.
- LangChainTools: Generates embeddings.
- PostgreSQL: Stores tourism, food, and hotel datasets.

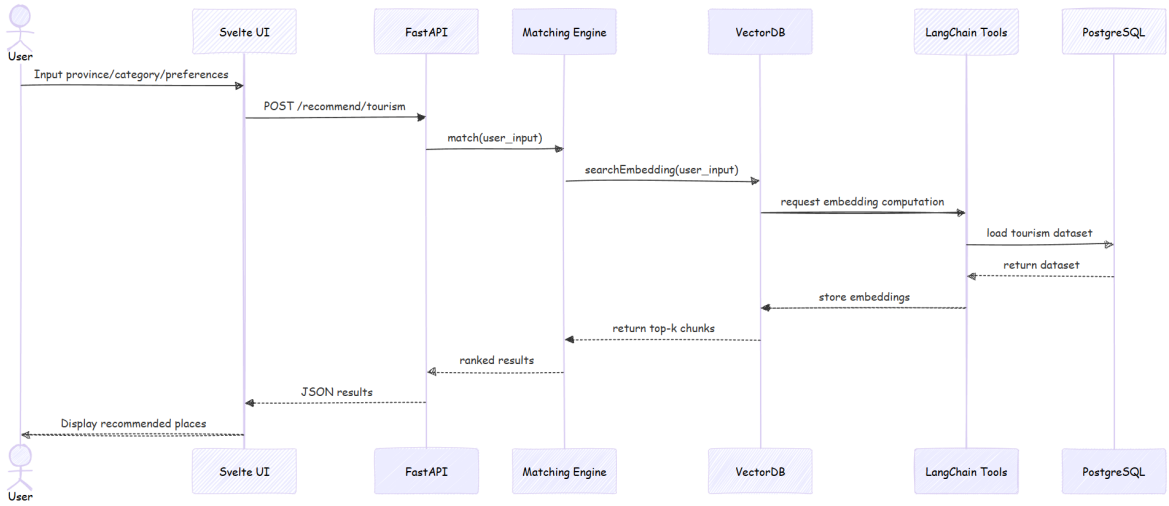


3.2.2 Sequence Diagram

The following sequence diagram illustrates the processing flow of the use case **"User Requests Tourist Destination Recommendations"**:

1. The user inputs data: province, preferences, and time.
2. The user interface sends a request to the API Server.
3. The backend invokes the TourismModule.
4. The TourismModule queries the MatchingEngine.
5. The MatchingEngine queries the VectorDB/Database.
6. Results are returned to the backend.

7. The backend sends the recommendation list back to the user.



3.2.3 3.3 User Interface Design

The user interface (UI) is designed to be simple, intuitive, and easy to use for general users. The mockups are created using Figma with the goal of enabling users to:

- Quickly input personal information and travel requirements.
- Clearly visualize recommendations for destinations, hotels, and foods.
- View detailed information for each destination and manage personal itineraries.

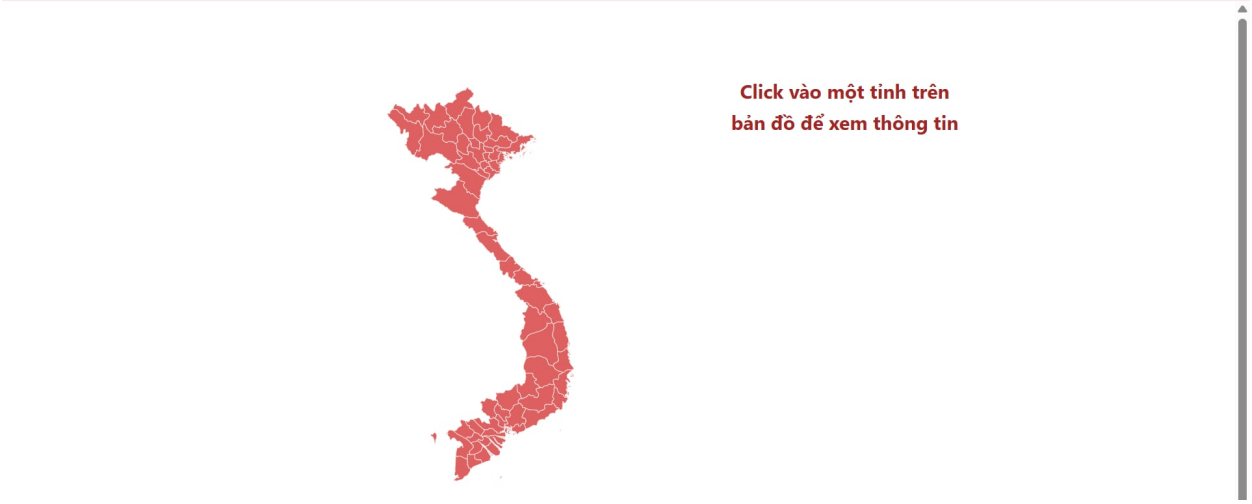
Below is a detailed description of the main interfaces.

3.3.1 Main Screens

This section presents the UI designs (Figma mockups) of the system's key screens. Currently, images have not been fully embedded; therefore, the report describes the functional layout of each interface and reserves space for future screenshots.

(1) Search Screen This screen allows users to input necessary information for generating recommendations:

- Travel province / city.
- Category (Tourism, Food, Hotels, Itinerary).
- Planned duration (number of days).
- Travel pace (Fast, Medium, Slow).
- Personal preferences (beach, mountains, culture, cuisine, etc.).

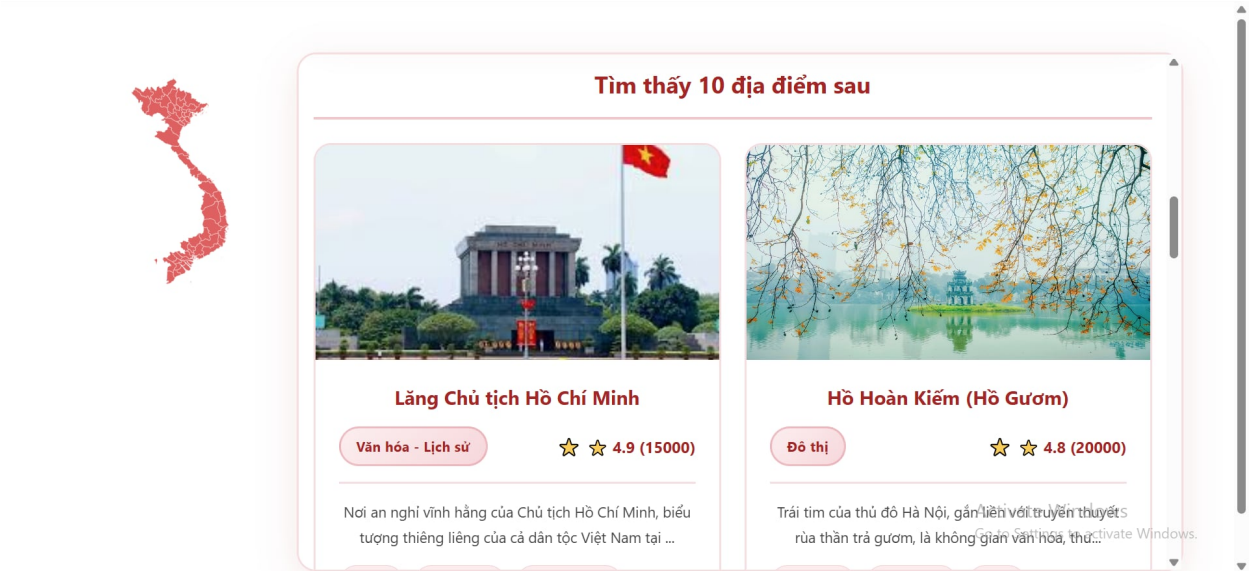


(2) Recommendation Screen After submitting input, users are redirected to the recommendation screen:

- A list of recommended destinations, hotels, or foods.
- Each item includes:
  - Name.
  - Illustrative image.
  - Distance (for hotels).
  - Matching score.
- Filtering options based on relevance score, distance, or category.

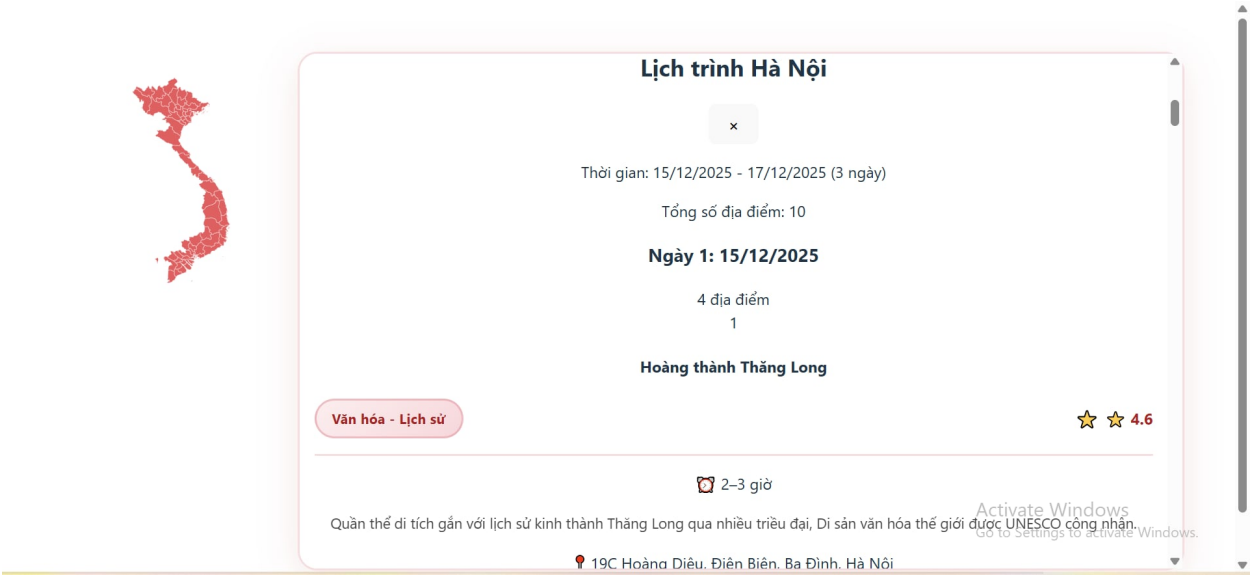
(3) Destination Detail Screen When a user selects a destination from the recommendation list, the detail screen is displayed:

- Large destination images.
- Full description (from dataset).
- Mini map (if available).
- Nearby places list (prioritizing hotels within 10 km).
- "Add to itinerary" button.



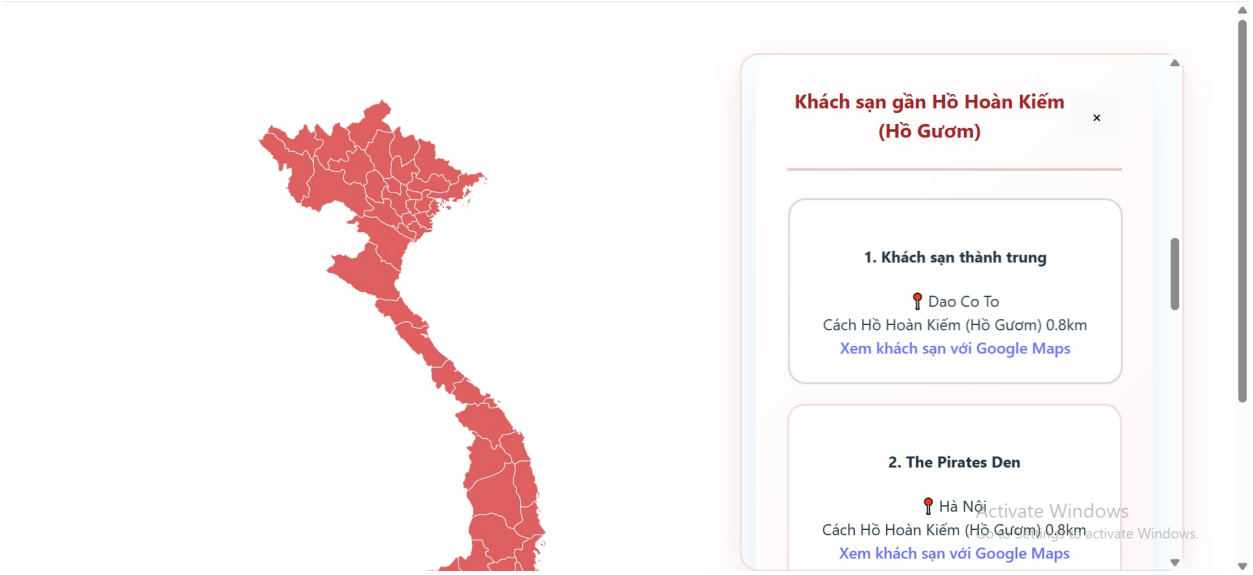
(4) Itinerary Management Screen This screen displays either automatically generated itineraries or user-customized itineraries:

- List of days (Day 1, Day 2, Day 3, etc.).
- Each day contains a list of activities (destination → time → activity).



(5) Nearby Hotels Screen This screen focuses on displaying hotels near the destination currently viewed:

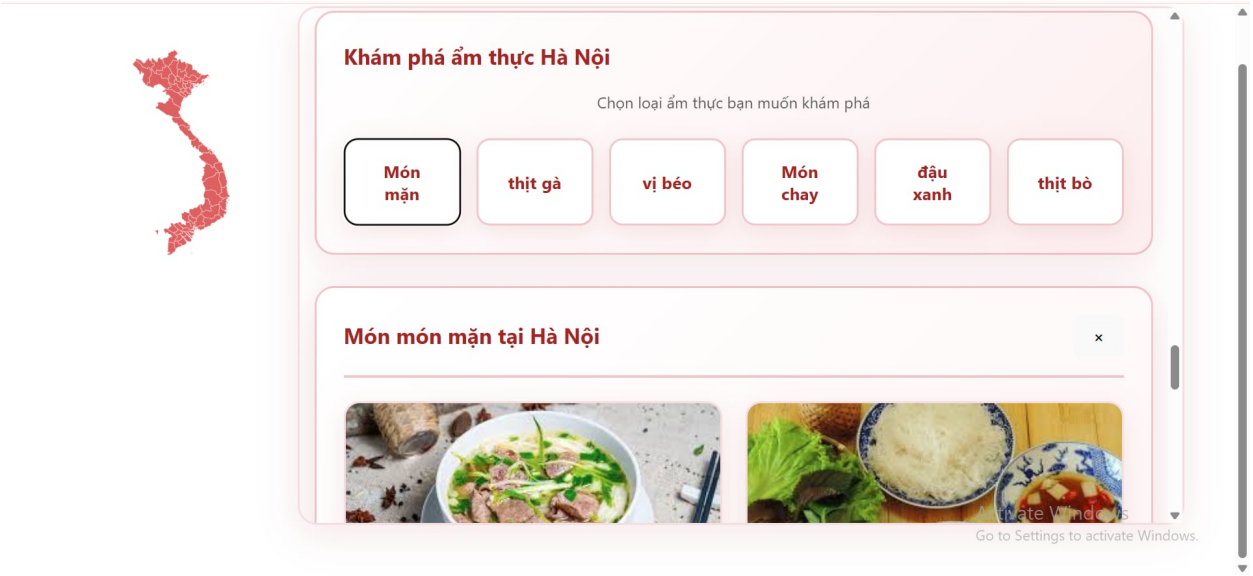
- List of hotels sorted by distance (ascending).
- Displays name, image, estimated price, and distance (km).
- "Select Hotel" or "View Details" button.



(6) Food Recommendation Screen This screen displays a list of local specialty foods suitable for the selected province/city. It is a key interface that helps users explore local culinary culture and add dining locations to their itineraries.

Main components of the screen include:

- **Recommended Food List**: The system displays the most suitable dishes based on:
  - Local specialties of the selected province/city.
  - User personal preferences.
- **Food Information**:
  - Dish name.
  - Illustrative image.
  - Short description.
- **Filters**:
  - By dish type (noodles, rice, seafood, etc.).



Chapter 4

Development Process

4.1 Task Assignment

Throughout the development of the Smart Tourism System, the team assigned tasks based on each member's skills and strengths to optimize overall efficiency. The table below describes the roles and main modules handled by each member:

- **Nam – Backend Developer, System Architect:** Responsible for system architecture design, building APIs using FastAPI, implementing the Recommendation Engine module, and integrating the PostgreSQL database.
- **Truc, Khang, Anh – Frontend Developers (including UI/UX Design):** Developed the user interface using Svelte (the framework agreed upon by the team), designed UI flows, handled API integration, and implemented data visualization for the Tourism, Foods, Hotels, and Itinerary screens. They also designed UI mockups using Figma, standardized screen layouts, and ensured an intuitive and user-friendly experience.
- **Tony, Hien – Data Engineers, Backend Developers:** Collected, preprocessed, and standardized data, including tourist destinations, foods, hotels, latitude/longitude, and distance calculations. Optimized geospatial queries and supported backend development by building FastAPI endpoints for food recommendations and nearby hotel recommendations (¡10 km).
- **Huy – Tester, Backend Developer:** Responsible for testing individual modules (unit testing and system testing), bug tracking, API testing, and implementing the RAG (Retrieval-Augmented Generation) mechanism for the itinerary recommendation module.

Contribution ratio: All members contributed equally (100%).

4.2 Achieved Results

As of the current stage, the team has completed the core modules in accordance with the objectives defined in Chapter 1. The achieved results include:

- **Completed Backend APIs**
  - APIs for recommending tourist destinations based on province/city, user preferences, and popularity.
  - APIs for recommending local specialty foods based on province/city and preferences.
  - APIs for listing hotels near tourist attractions (radius ¡ 10 km).
  - APIs for generating itineraries based on travel duration and user travel pace.
- **Completed Recommendation Engine:** The rule-based recommendation logic operates stably with high accuracy and provides faster response times compared to vector-database-based approaches.
- **Completed PostgreSQL Database Structure:** Including tables such as `tourism_places`, `vietnam_foods`, `vietnam_hotels`, and related relational tables.
- **Completed Basic Frontend**
  - Tourist recommendation screen.
  - Food recommendation screen.
  - Nearby hotels display screen.
  - Itinerary screen.
- **API-level and Workflow-level Testing:** Key endpoints were tested using Postman and functioned correctly according to requirements.

Overall, the team has completed more than 70% of the total planned workload.

4.3 Next Phase Plan

As the project has entered the final week of the academic term, the next phase will primarily focus on data refinement, recommendation logic optimization, and system deployment in preparation for the final presentation. The main tasks include:

- **Data Expansion and Cleaning** Expanding datasets for tourist destinations, local specialty foods, and hotels across multiple provinces/cities to improve recommendation diversity and accuracy. Data will also be standardized by removing duplicates, handling missing fields, and correcting formatting issues.

- **Improving Recommendation Algorithms** Refining recommendation logic based on enriched datasets and optimizing filters for distance, preferences, and popularity.
- **Finalizing APIs and Feature Testing** Conducting end-to-end testing for all major APIs, including destination recommendations, itinerary generation, food recommendations, and nearby hotel recommendations. Fixing issues identified during testing.
- **System Deployment**
  - Deploying the backend to cloud services such as Render, Railway, or AWS.
  - Deploying the frontend to Vercel or Netlify.
  - Configuring domain names, HTTPS, and accessibility checks.
- **Final Report Preparation** Completing the written report, presentation slides, and demo video for the final defense.

4.4 Challenges and Risks

During the development process, the team encountered several challenges and potential risks, including:

- **Lack of Real-world Tourism Data** Data from many provinces/cities was inconsistent, requiring manual collection and standardization.
- **Recommendation Logic Optimization** The initial recommendation algorithms were not sufficiently accurate, requiring improvements to rule-based logic and data cleaning.
- **Frontend–Backend Integration Issues** Some APIs returned complex data structures (such as multi-day itineraries), making UI rendering challenging and requiring additional adjustments.
- **System Scalability Risks** Future integration of vector databases or microservices could increase deployment and maintenance costs.

The team continues to address these issues to ensure the system achieves the highest possible quality upon delivery.

References

[1] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, M. Wang, and H. Wang, *"Retrieval-Augmented Generation for Large Language Models: A Survey,"* Shanghai Research Institute for Intelligent Autonomous Systems, Tongji University; School of Computer Science, Fudan University; College of Design and Innovation, Tongji University.

[2] Keivalya Pandya, Prof. Dr. Mehfuza Holia, *"Automating Customer Service using LangChain, Building custom open-source GPT Chatbot for organizations"*, Birla Vishvakarma Mahavidyalaya, Gujarat, India.

[3] N. Toyaad Kumar Reddy, Manas Ranjan Patra, Brojo Kishore Mishra, *"Design and Implementation of an AI-Based Chatbot Framework with Retrieval Augmented Generation and Integrated Recommender System for Interactive User Support"*, Dept. of Computer Science and Engineering NIST University, Institute Park, Berhampur.

[4] Verena Traubinger, Martin Gaedke, *"Interaction Design Patterns of Web Chatbots"*, Faculty of Computer Science, Chemnitz University of Technology, Germany.