

# **AI Driven Smart Tourism Platform for Personalized Safe & Sustainable Travel Planning**



Presenter  
**R25 -006**



**R25 - 006**

# Meet Our Team



**SRIKANTH.S**

IT21821240



**SENEVIRATHNE S.D.C.D**

IT21831768



**Thuwakaran. R**

IT21835728



# Table Of Content

01. BACKGROUND
02. RESEARCH PROBLEM
03. OBJECTIVES
04. OVERALL SYSTEM  
DIAGRAM
05. INDIVIDUAL COMPONENTS



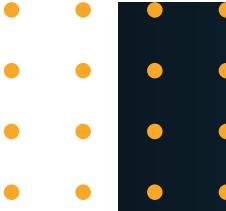
# BACKGROUND



- Modern travelers seek highly personalized experiences and seamless itinerary planning tools.
- Artificial Intelligence and real-time data are revolutionizing travel by enabling smarter and tailored experiences.
- Existing platforms lack advanced features like, collaborative travel platform, flexible itinerary management, and effective budget planning.

# RESEARCH PROBLEM

---



01

Existing platforms fail to combine itinerary optimization, and cost management, making it difficult for travelers to access a comprehensive solution in one place.

02

Travelers have limited options to modify their plans dynamically based on changing preferences, unexpected events, or real-time conditions during their journey.

03

Most platforms do not focus on enabling group-based travel planning or connecting travelers with similar interests to collaborate on trips or events.

04

Current tools lack the ability to provide predictive cost estimates and adjust budgets dynamically, preventing travelers from effectively managing their expenses across different activities

# OBJECTIVES

## OBJECTIVE 01

Provide a Collaborative Travel Companion Platform for group matching and real-time communication

## OBJECTIVE 02

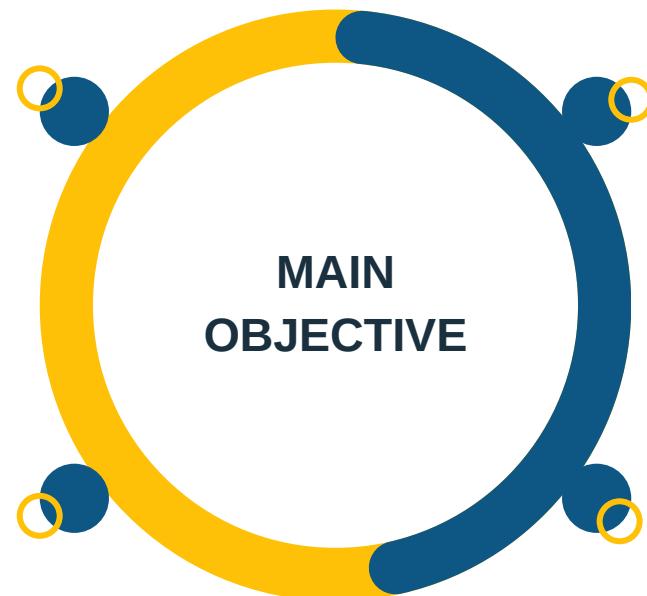
Provide a feature hybrid Personalized Recommendation and itineraries planning system.

## OBJECTIVE 03

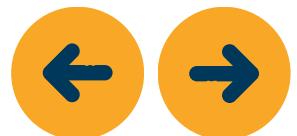
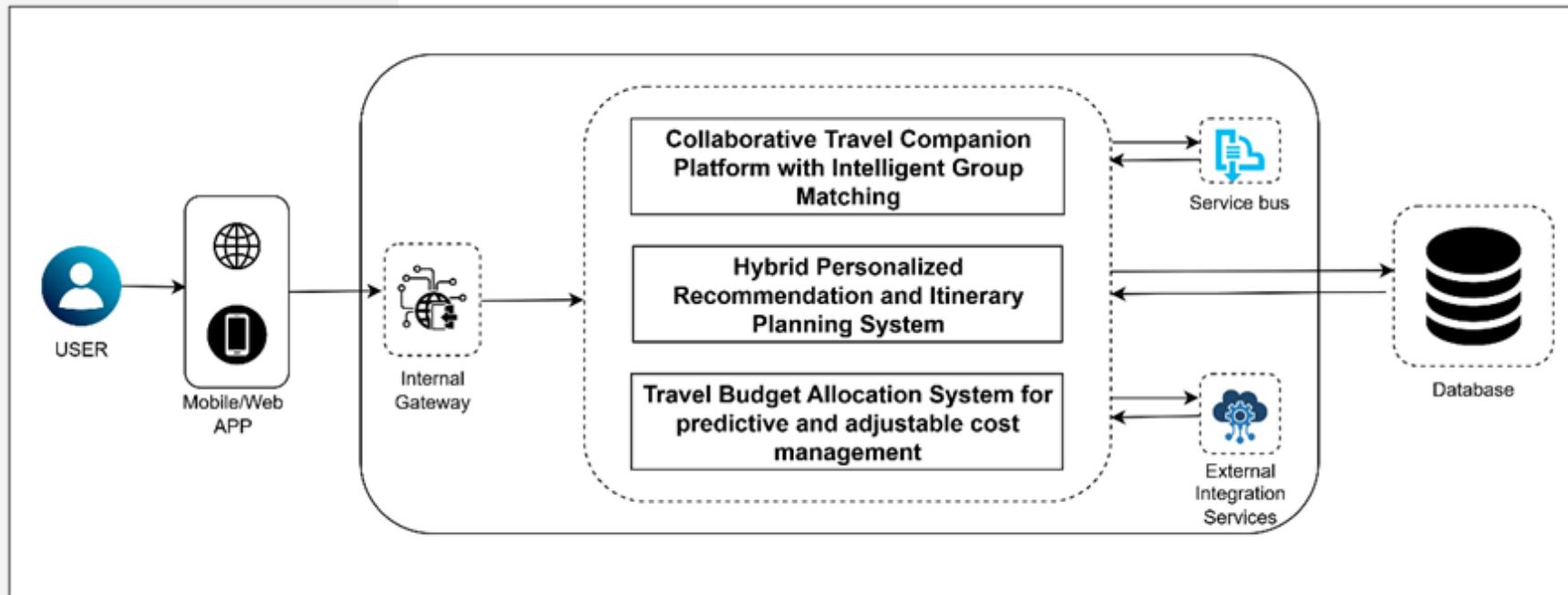
Provide a feature Travel Budget Allocation System for predictive and adjustable cost management.

## OBJECTIVE 04

Providing a web application that ensures performance, reliability, correctness, and excellent UI/UX.



# OVERALL SYSTEM DIAGRAM





# GENERAL DETAILS

01. GANTT CHART
02. COMMERCIALIZATION PLAN
03. GITHUB DETAILS
04. PROJECT MANAGEMENT
05. RISK MANAGEMENT



# GANTT CHART

Task List	Nov	Dec	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct
Research Topic Selection												
Feasibility Study												
Evaluation Feasibility & Background Study												
Topic Evaluation												
Requirement Gathering												
Background Survey												
Literature Review												
Requirement Analysis												
Software Requirement Specification												
Functional & Non-functional Requirements												
Project Charter												
Proposal Presentation												
Project Proposal Report												
Software Design												
Designing Wireframes												
ML Component Development												
Frotend Development												
Progress Presentation 01												
Research Paper												
Unit Testing												
Progress Presentation 02												
Software Integration												
Integration Testing												
Deployment & Maintanance												
Final Presentation & Viva												

# COMMERCIALIZATION



The proposed solution will be integrated with the web application and these features are very unique

By comparing the competitors these features are new to the travel platform.

**SUBSCRIPTION**

**ADVERTISING STRATEGY**

# GITHUB DETAILS

The screenshot shows a GitHub repository page for a project named "Research-Project". The repository is public and has 13 branches. The main branch is "main". There are 42 commits in total, with the most recent being a merge pull request from "SLIIT-Projects-Y4/feature/dashboard" by "last-chaanz". Other commits include updates to "backend", "frontend", and ".gitignore". The repository has 0 stars, 0 forks, and 0 watching. It includes sections for About, Releases, Packages, and Contributors.

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Research-Project Public

Your main branch isn't protected

Protect this branch

main 13 Branches 0 Tags

Go to file Add file Code

last-chaanz Merge pull request #8 from SLIIT-Projects-Y4/feature/dashboard a57a91b - last week 42 Commits

backend model update last week

frontend frontendfiles update last week

.gitignore all new file update with the latest changes of backend and fr... 2 weeks ago

README.md Initial commit 6 months ago

README

Research-Project

About

Final year Research Project

Readme Activity Custom properties

0 stars 0 watching 0 forks

Report repository

Releases

No releases published Create a new release

Packages

No packages published Publish your first package

Contributors 2

# PROJECT MANAGEMENT

The screenshot shows a Jira project management interface for the space "Research Project Management". The board has four columns: TO DO, IN PROGRESS, TESTING, and DONE.

- TO DO:** 9 items
  - Final Presentation (GroupTask, CPG-8)
  - Platform Website (CPG-31)
  - CheckList - II (GroupTask, CPG-33)
  - Final Report - Safety Assistant (CPG-30)
  - Testing the system - Safety Assistant
- IN PROGRESS:** 5 items
  - Implementation - Collaborative Platform (90%) (CPG-17)
  - Implementation - Itinerary Planner (90%) (CPG-18)
  - Implementation - Budget Planner (90%) (CPG-19)
  - Create API for the models. (CPG-43)
  - Research LogBook
- TESTING:** 4 items
  - Testing the system - Collaborative Platform backend (CPG-5)
  - Testing the system - Collaborative Platform Frontend (CPG-52)
  - Testing the system- Itinerary Planner (CPG-14)
  - Testing the system - Budget Planner (CPG-15)
- DONE:** 17 items
  - Final Group Report (GroupTask, EPG-26)
  - Final Report- Itinerary Planner (EPG-28)
  - Final Report - Budget Planner (EPG-29)
  - Research Paper (GROUP TASK, EPG-7)
  - Progress Presentation - 2 (EPG-25)

Each card includes a "Group" icon and a small circular badge with initials (e.g., SS, C, MN, TR). A "Quickstart" button is visible in the bottom right corner of the Done column.

# RISK MITIGATION

Risk	Trigger	Owner	Response	Resource Required
<strong>Risk with respect to the Project Team</strong>				
Minor delays in deliverables	Small miscommunications or changes in priority	Project Leader	Discuss with team to identify any small delays. Adjust timelines as needed.	Updated Project Schedule Plan/Gantt Chart
<strong>Risk with respect to the Panel/Supervisor(s)</strong>				
Slight misalignment on expectations	Minor changes in requirements	Project Leader	Confirm and clarify expectations through quick discussions. Update project documents if necessary.	Meeting Notes Updated Project Plan
Panel preferences shift slightly	Panel's evolving preferences or needs	Project Leader	Seek feedback regularly to stay aligned with expectations. Make small adjustments as needed.	Meeting Logs Updated Deliverables
<strong>Risk with respect to Project Execution</strong>				
Minor technical hiccups	Occasional bugs or troubleshooting	Project Leader	Allocate time for quick debugging and testing. Work with the team to address issues.	Debugging Tools Technical Documentation



**IT21821240 - SRIKANTHAN.S**

**SPECIALIZING - SOFTWARE ENGINEERING**

**COLLABORATIVE TRAVEL COMPANION PLATFORM  
WITH INTELLIGENT GROUP MATCHING**





# INTRODUCTION

01. BACKGROUND
02. RESEARCH QUESTIONS
03. OBJECTIVES



# BACKGROUND



- Travel platforms currently lack personalized group matching based on users' specific preferences.
- Limited tools for real-time coordination of activities and cost-sharing among group members.
- Lack of continuous feedback mechanisms to improve recommendations and user satisfaction.

# RESEARCH PROBLEM

---



01

How can intelligent group matching improve the travel experience for users?

02

What features can facilitate effective real-time coordination among travel group members?

03

How can user-generated itineraries and shared experiences help others with similar preferences?

04

How can user groups support travellers in finding suitable travel buddies and planning cost-effective trips?



# OBJECTIVES

## OBJECTIVE 01

To match users with similar travel interests, styles, destinations, and budget preferences into compatible travel groups.

## OBJECTIVE 03

To facilitate seamless communication between group members through a built-in chat system for planning and coordination.

## OBJECTIVE 02

To recommend existing travel groups that best suit the user's profile, avoiding the need to create new groups unless necessary.

## OBJECTIVE 04

Ensure user-friendly interface and accessible for diverse traveler groups



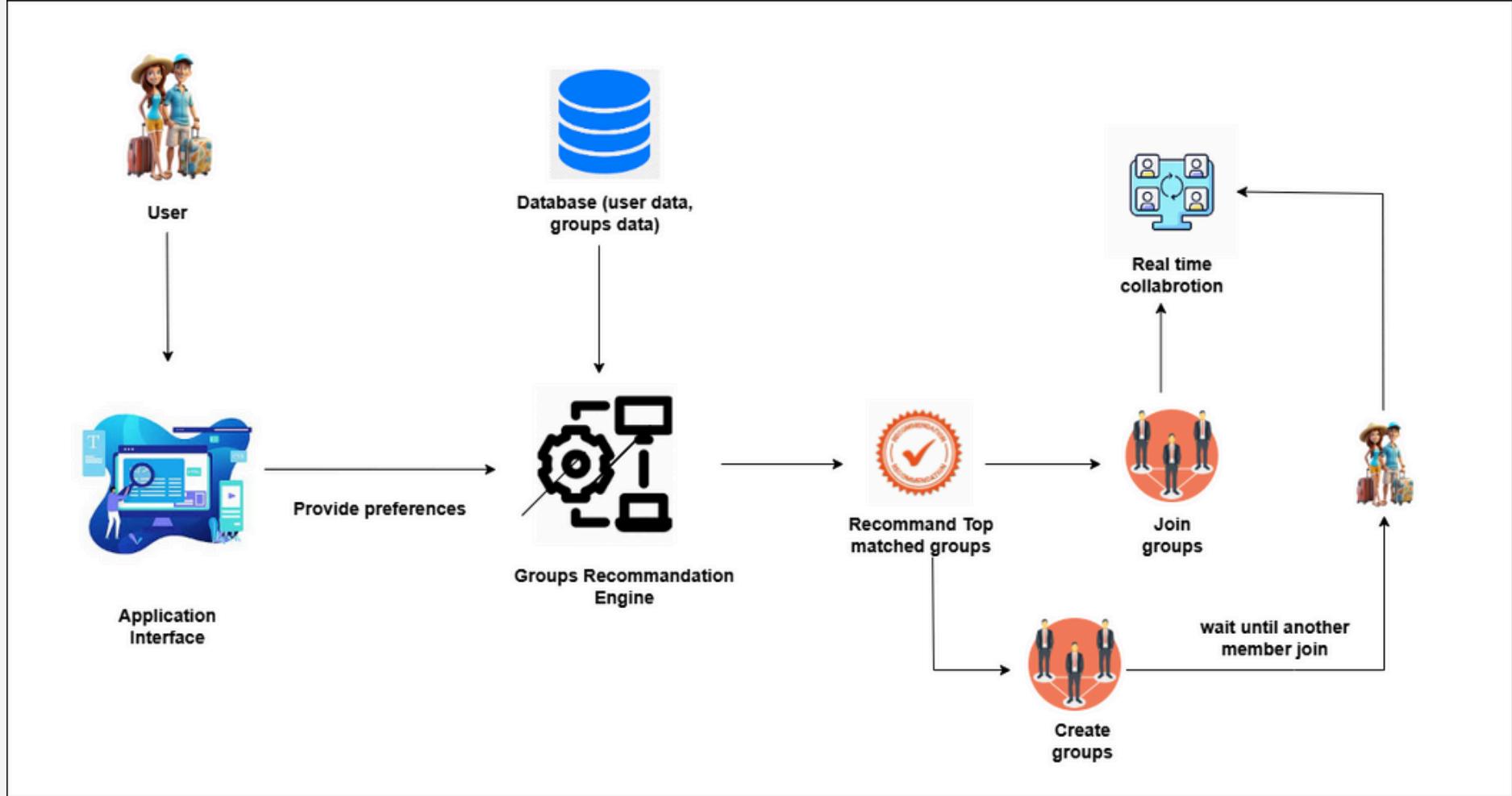


# METHODOLOGY

01. SYSTEM DIAGRAM
02. TOOLS AND TECHNOLOGIES
03. REQUIREMENTS
04. GITHUB COMMITS



# SYSTEM DIAGRAM



# TOOL AND TECHNOLOGIES



**Programming Languages**  
Python



**Project Management**  
Jira



**Database**  
MongoDB



**Frameworks & Libraries**  
lightgbm, pandas, numpy, fastapi, Scikit-learn, sentence-transformers, React



**Other Tools**  
Git , Draw.io  
Postman, Docker & Docker Compose

# REQUIREMENTS

## FUNCTIONAL

- Data Collection and Preprocessing
- Model Training
- User-Group Matching and Recommendations
- Real-Time Communication
- Trip Planning and Group Collaboration
- Platform Implementation

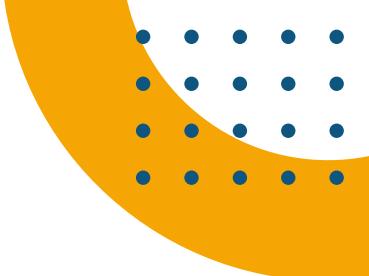
## NON-FUNCTIONAL

- Performance
- Scalability
- Reliability
- Usability
- Security
- Maintainability

## Standards and best practices

- Code quality
- Web security
- Modal quality
- Data Privacy

# GITHUB COMMITS



Screenshot of a GitHub repository named "SLIIT-Projects-Y4 / Research-Project". The "Commits" tab is selected. A dropdown menu shows the branch "feature/develop...". The search bar contains "Type ⌘ to search". The user "IT21821240" is selected in the dropdown, and the time range is set to "All time".

The commits are listed by date:

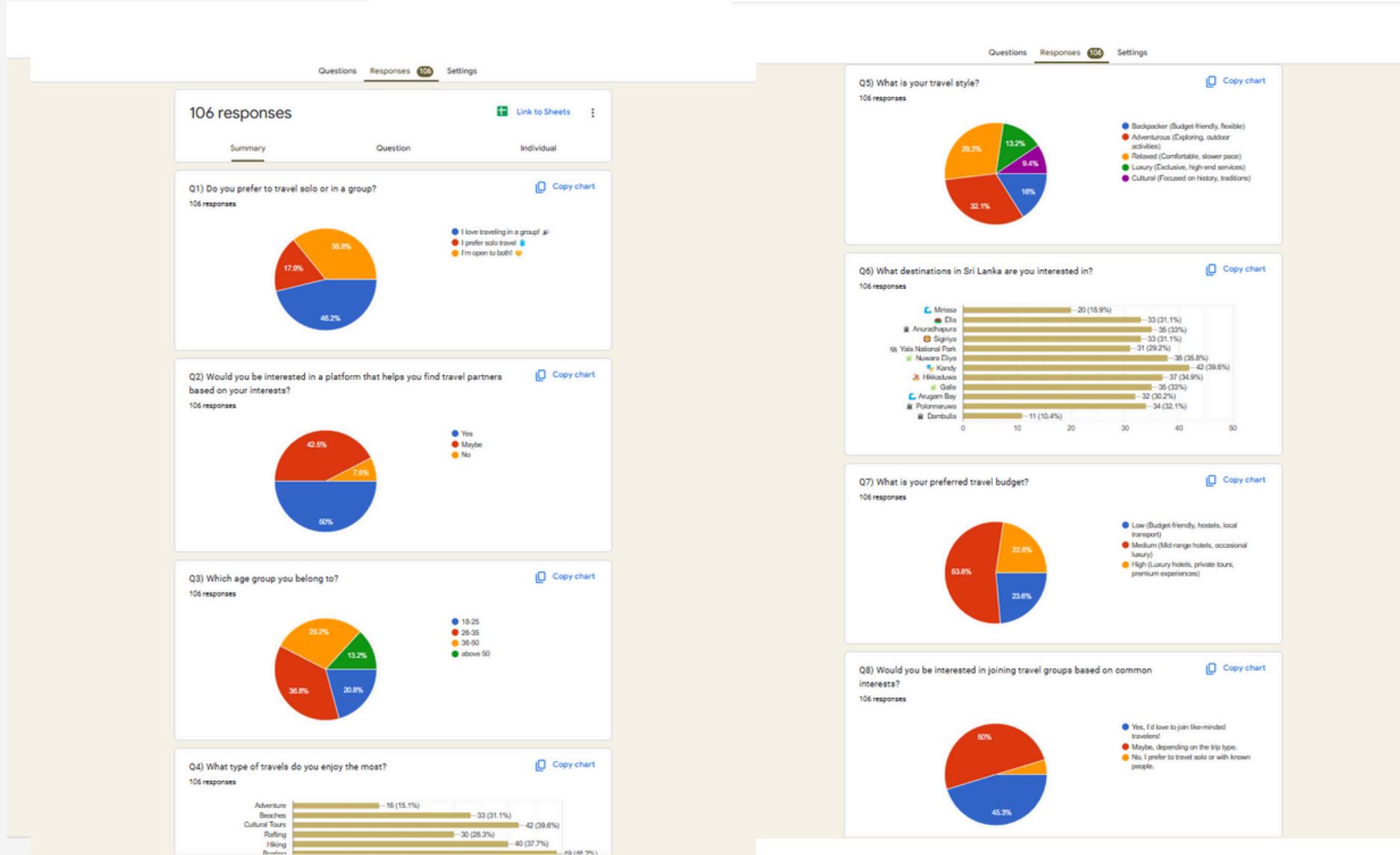
- Commits on Sep 14, 2025**
  - made some UI update** (IT21821240 committed 2 hours ago) - Commit hash: 669a504
  - fetch unique plan pool names** (IT21821240 committed 14 hours ago) - Commit hash: 6a92724
  - UI update** (IT21821240 committed 14 hours ago) - Commit hash: 5e97a75
- Commits on Sep 13, 2025**
  - updated some flow** (IT21821240 committed 18 hours ago) - Commit hash: 81b13dd
  - Merge branch 'feature/ui-development' of https://github.com/SLIIT-Projects-Y4/Research-Project into feature/ui-development** (IT21821240 committed 19 hours ago) - Commit hash: 1a88d24
  - updated some UI** (IT21821240 committed 19 hours ago) - Commit hash: 4b408c6
  - UI update**



# COMPLETION OF THE PROJECT

50%

# DATA COLLECTION - SURVEY



# DATA SET CREATION

	A	B	C	D	E	F	G	H	I	J	K
1	User_ID	Age	User_Interest	Preferred_Destination	Budget	Travel_Style					
2	U0001	20	Cultural Tours	Gregory Lake Community Tsunami Museum	Low	Balanced					
3	U0002	22	Educational Tours, Rafting, Cultural Activities	Arugam Bay	Medium	Luxury					
4	U0003	19	Animal Watching	Pinnawala Elephant Orphanage, Sea Turtle Farm Galle Mahamodara, Ceylon Tea Museum	Low	Backpacker					
5	U0004	25	Yoga, Rafting, Boating	Colombo National Museum, Arugam Bay	Low	Adventurous					
6	U0005	24	Kit Surfing, Swimming	Jaya Sri Maha Bodhi, Polonnaruwa, Rriegala Forest Monastery	Low	Relaxed					
7	U0006	20	Architectural Marvel, Wildlife Spotting	Arugam Bay	Medium	Relaxed					
8	U0007	23	Surfing, Animal Watching, Sunbathing	Lover's Leap Falls	High	Relaxed					
9	U0008	19	Boating, Cultural Tours, Kite Surfing	Lover's Leap Falls, Handunugoda Tea Estate, Jaya Sri Maha Bodhi	Medium	Balanced					
10	U0009	18	Nature Walks, Jet Skiing	Kelaniya Raja Maha Vihara	High	Balanced					
11	U0010	18	Historical Exploration, Wildlife Safaris, Learning History	Udawalawe National Park	High	Balanced					
12	U0011	21	Animal Watching, Fruit Picking	Bennota Beach	High	Luxury					
13	U0012	20	Walking Trails	Temple of the Sacred Tooth Relic, Mihintale, Victoria Park of Nuwara Eliya	Medium	Luxury					
14	U0013	23	Educational Tours, Swimming, Fruit Picking	Dambatenne Tea Factory	Low	Adventurous					
15	U0014	18	Hiking, Yoga, Sunbathing	Horton Plains National Park, National Zoological Gardens of Sri Lanka, Wilpattu National Park	Medium	Adventurous					
16	U0015	18	Kite Surfing	Ravina Ella Falls, Arugam Bay	High	Backpacker					
17	U0016	24	Trekking	National Zoological Gardens of Sri Lanka	Low	Relaxed					
18	U0017	24	Photography, Farm Tours, Rafting	Colombo National Museum, Ceylon Tea Museum, Kalametiya Eco Bird Watching	Medium	Luxury					
19	U0018	23	Trekking, Yoga	Dambulla Cave Temple	Low	Backpacker					
20	U0019	21	Yoga, Hiking	Negombo Beach, Gregory Lake	Low	Adventurous					
21	U0020	18	Photography, Snorkeling	Kalametiya Eco Bird Watching, Hakgala Botanic Gardens	High	Adventurous					
22	U0021	19	Kite Surfing, Worship, Animal Watching	Diyakula Falls	High	Relaxed					
23	U0022	22	Worship, Kite Surfing	Arugam Bay	Low	Balanced					
24	U0023	19	Kite Surfing	Bennota Beach, Tissa Wewa	Low	Relaxed					
25	U0024	18	Boating, Exploring Exhibits	Mount Lavinia Beach, Ramboda Waterfall, Mirissa Beach	Low	Relaxed					
26	U0025	23	Trekking	Passikudah Beach, Temple of the Sacred Tooth Relic, Jethawanaramaya Stupa	High	Adventurous					

	A	B	C	D	E	F	G
1	Group_ID	Group_Name	Status	Budget	Travel_Style	Destinations_Planned	Group_Interest
2	G0001	Radiant Nomads	Active	High	Relaxed	Wilpattu National Park, Sinharaja Forest Reserve, Diyakula Falls	Kite Surfing, Agricultural Workshops
3	G0002	Adventure Nomads	Inactive	Medium	Relaxed	Ranawana Falls, Samadhi Statue, Brief Garden, Bevis Bawa, Rriegala Forest Monastery	Hiking, Fruit Picking
4	G0003	Thrill-seekers	Active	High	Adventurous	Brief Garden, Bevis Bawa, Ceylon Tea Museum, Sigiriya The Ancient Rock Fortress, Pigeon Island National Park	Snorkeling, Agricultural Workshops
5	G0004	Wanderlust Discoverers	Inactive	Low	Relaxed	Pedro Tea Factory, Pigeon Island National Park	Animal Watching, Swimming
6	G0005	Relaxed Sojourners	Inactive	Medium	Luxury	Diyakula Falls, Ariyapala Mask Museum, Brief Garden - Bevis Bawa	Cultural Tours, Hiking
7	G0006	Uively Wanderers	Inactive	Low	Balanced	Colombo National Museum, Pigeon Island National Park, World Buddhist Museum	Sightseeing, Hiking
8	G0007	Elite Pioneers	Active	Adventurous	Kandy Lake, Polonnaruwa	Tea Plucking, Rafting	
9	G0008	Energetic Pathfinders	Active	High	Balanced	National Zoological Gardens of Sri Lanka, Marble Beach, Bundala National Park	Animal Watching, Fishing
10	G0009	Epic Drifters	Inactive	Low	Backpacker	Bluefield Tea Gardens, Royal Botanical Gardens	Educational Tours, Historical Exploration
11	G0010	Cheerful Wanderers	Inactive	High	Adventurous	Wilpattu National Park, Victoria Park of Nuwara Eliya, Mihintale	Fishing, Kite Surfing
12	G0011	Epic Pioneers	Active	High	Adventurous	Jaya Sri Maha Bodhi, New Ranwell Spice Garden, Udawalawe National Park	Sightseeing, Wildlife Safaris
13	G0012	Brave Travellers	Active	Medium	Backpacker	Ariyapala Mask Museum, Udawattekelle Sanctuary, Udawalawe National Park, Pigeon Island National Park	Swimming, Boating
14	G0013	Relaxing Pioneers	Active	Medium	Backpacker	Marble Beach, Tissa Wewa, Baker's Falls	Nature Photography, Boating
15	G0014	Relaxing Sojourners	Active	Medium	Adventurous	Samadhi Statue	Hiking, Meditation
16	G0015	Majestic Adventurers	Inactive	High	Balanced	Mount Lavinia Beach	Trekking, Animal Watching
17	G0016	Cultural Globetrotters	Active	Low	Backpacker	World Buddhist Museum, Samadhi Statue, Ramboda Waterfall, Community Tsunami Museum, Ruwanwelisaya	Photography, Cultural Exploration
18	G0017	Majestic Seekers	Inactive	Medium	Backpacker	Wilpattu National Park, Sea Turtle Farm Galle Mahamodara	Animal Watching, Hiking
19	G0018	Explorer Adventurers	Inactive	Low	Backpacker	Abhayagiri Dagoba, Baker's Falls, Udawattekelle Sanctuary	Animal Watching, Architectural Marvel
20	G0019	Vibrant Escapists	Active	High	Adventurous	Kandy Lake, St. Clair's Falls, Nilaveli Beach	Nature Walks, Surfing
21	G0020	Energetic Escapists	Active	High	Relaxed	Jaffna Fort, Polonnaruwa, Dagoba of Thuparama, Samadhi Statue, Glenloch Tea Factory	Tea Plucking, Animal Watching
22	G0021	Thrilling Dreamers	Active	Low	Balanced	Colombo National Museum, Konawewama Temple, Ambewela Farms	Learning History, Wildlife Spotting
23	G0022	Majestic Dreamers	Active	Low	Relaxed	Twinni Beach (Kuttim Poluna), Gangaramaya (Vihara) Buddhist Temple	Worship, Cultural Tours
24	G0023	Epic Travellers	Active	Medium	Backpacker	Sigiriya Museum, Galile Fort, Glenloch Tea Factory	Educational Tours, Kit Surfing
25	G0024	Wanderlust Pioneers	Inactive	High	Luxury	Lover's Leap Falls, World Buddhist Museum, Rriegala Forest Monastery, Ravana Ella Falls, Arugam Bay	Architectural Marvel, Hiking
26	G0025	Vibrant Explorers	Active	Low	Adventurous	Diyakula Falls, Ariyapala Mask Museum, Brief Garden - Bevis Bawa	

# DATA PRE PROCESSING

```
# Load Dataset
user_df = pd.read_csv("/content/drive/My Drive/Colab Notebooks/User_Preference.csv")
group_df = pd.read_csv("/content/drive/My Drive/Colab Notebooks/Group_Dataset.csv")
interaction_df = pd.read_csv("/content/drive/My Drive/Colab Notebooks/User-Group_Interactions_Dataset.csv")
```

Python

```
# Step 1: Preprocess multi-label fields
def split_multi(x):
    if pd.isna(x):
        return []
    return [i.strip() for i in x.split(',') if isinstance(x, str) else []]

user_df['User_Interest'] = user_df['User_Interest'].apply(split_multi)
user_df['Preferred_Destination'] = user_df['Preferred_Destination'].apply(split_multi)
group_df['Group_Interest'] = group_df['Group_Interest'].apply(split_multi)
group_df['Destinations_Planned'] = group_df['Destinations_Planned'].apply(split_multi)
```

Python

```
#Step 2: Merge DataFrames into a single DataFrame
df = interaction_df.merge(user_df, on='User_ID').merge(group_df, on='Group_ID')
```

Python

```
#Step 3: Ensure no missing values
df['Budget_x'] = df['Budget_x'].fillna('Unknown')
df['Travel_Style_x'] = df['Travel_Style_x'].fillna('Unknown')
df['Age'] = df['Age'].fillna(df['Age'].median())
df['Current_Members'] = df['Current_Members'].fillna(df['Current_Members'].median())
```

Python

# MODEL SELECTION

## Selected Model LightGBM with LambdaRank

Explored Random Forest Classifier model to predict whether a user would join a group

It had two major limitations:

- It did not rank groups by relevance.
- It could only say “join or not”, not which group is most suitable

### Reasons:

- Optimized for Ranking
- Supports Complex Features
- High Performance
- Better Accuracy
- Hybrid Capability



# MODEL TRAINING

```
# Model.ipynb ×
Backend > TrainedModel > Model.ipynb > #Step 2: Merge DataFrames into a single DataFrame
+ Code + Markdown | ▶ Run All ...
... ['/content/drive/MyDrive/feature_scaler.pkl']

#Step 6: Prepares data for the ranking model
X = np.hstack([
    age_scaled,
    budget_encoded,
    style_encoded,
    df[['Interest_Similarity', 'Destination_Similarity']].values,
    members_scaled
])
y = (df['Joined'] == 'Yes').astype(int).values
user_ids = df['User_ID'].values

#Step 6: Prepares data for evaluation and training
train_users, test_users = train_test_split(np.unique(user_ids), test_size=0.2, random_state=42)
train_mask = df['User_ID'].isin(train_users)

X_train, X_test = X[train_mask], X[~train_mask]
y_train, y_test = y[train_mask], y[~train_mask]
train_group = df[train_mask].groupby('User_ID').size().tolist()
test_group = df[~train_mask].groupby('User_ID').size().tolist()
test_user_ids = user_ids[~train_mask]

#Step 7: Builds a model that ranks groups
train_data = lgb.Dataset(X_train, label=y_train, group=train_group)
valid_data = lgb.Dataset(X_test, label=y_test, group=test_group)

params = {
    'objective': 'lambdaRank',
    'metric': 'ndcg',
    'learning_rate': 0.1,
    'max_depth': 7,
    'verbosity': -1,
    'random_state': 42
}

callbacks = [
    lgb.early_stopping(stopping_rounds=10),
    lgb.log_evaluation(period=10)
]

ranker = lgb.train(
    params=params,
    train_set=train_data,
    num_boost_round=150,
    valid_sets=[valid_data],
    valid_names=['validation'],
    callbacks=callbacks
)

...
Training until validation scores don't improve for 10 rounds
[10] validation's ndcg@1: 0.66      validation's ndcg@2: 0.735712  validation's ndcg@3: 0.853632  validation's ndcg@4: 0.853632  validation's ndcg@5: 0.853632
Early stopping, best iteration is:
[1] validation's ndcg@1: 0.73      validation's ndcg@2: 0.775178  validation's ndcg@3: 0.883427  validation's ndcg@4: 0.883427  validation's ndcg@5: 0.883427
```

# MODEL EVALUATION

The screenshot shows a Jupyter Notebook interface with two code cells.

**Cell 1:**

```
# Compute Top-3 Hit Rate
top_3_hit_count = 0
valid_users = 0

for true, pred in zip(true_groups, pred_groups):
    if len(true) < 3:
        continue # skip users with < 3 options

    # Get indices of top-3 predicted scores
    top3_idx = np.argsort(pred)[::-1][:3]

    # Check if top-3 contains the max relevance value
    if np.max(true[top3_idx]) == np.max(true):
        top_3_hit_count += 1

    valid_users += 1

hit_at_3 = top_3_hit_count / valid_users
print(f"Top-3 Hit Rate: {hit_at_3:.2%}")
```

Output:

- ✓ NDCG@3 Score: 0.8130903102557878
- ⚡ Top-3 Hit Rate: 99.01%

**Cell 2:**

```
model_path = "/content/drive/MyDrive/trained_lgbm_ranker_semantic.txt"
ranker.save_model(model_path)
print("Model saved to:", model_path)
```

Output:

- ✓ Model saved to: /content/drive/MyDrive/trained\_lgbm\_ranker\_semantic.txt

Bottom navigation bar:

- { } Variables
- Terminal

# WORKING COMPONENT

```
> import joblib
> import numpy as np
> import pandas as pd
> from sentence_transformers import SentenceTransformer
> import lightgbm as lgb

# Load saved components
model1 = lgb.Booster(model_file="/content/drive/MyDrive/trained_lgbm_ranker_semantic.txt")
budget_encoder = joblib.load("/content/drive/MyDrive/budget_encoder.pkl")
style_encoder = joblib.load("/content/drive/MyDrive/style_encoder.pkl")
scaler = joblib.load("/content/drive/MyDrive/feature_scaler.pkl")
sbert = SentenceTransformer('all-MiniLM-L6-v2')

# Load group dataset
group_df = pd.read_csv("/content/drive/My Drive/Colab Notebooks/Group_Dataset.csv")
group_df['Group_Interest'] = group_df['Group_Interest'].fillna('').apply(lambda x: [i.strip() for i in x.split(',')])
group_df['Destinations_Planned'] = group_df['Destinations_Planned'].fillna('').apply(lambda x: [i.strip() for i in x.split(',')])

def recommend_groups_for_user(age, budget, interests, preferred_destinations=None, travel_style="Unknown"):
    # Clean up destinations list
    preferred_destinations = preferred_destinations if preferred_destinations else []

    # Compute user embeddings
    user_interest_text = ", ".join(interests)
    user_interest_emb = sbert.encode([user_interest_text])[0]

    if preferred_destinations:
        user_dest_text = ", ".join(preferred_destinations)
        user_dest_emb = sbert.encode([user_dest_text])[0]
    else:
        user_dest_emb = np.zeros(384) # Fallback for destination

    # Group embeddings
    group_interest_embs = sbert.encode(group_df['Group_Interest'].apply(lambda x: ', '.join(x)).tolist())
    group_dest_embs = sbert.encode(group_df['Destinations_Planned'].apply(lambda x: ', '.join(x)).tolist())

    # Compute distances between user embeddings and group embeddings
    user_interest_emb_norm = user_interest_emb / np.linalg.norm(user_interest_emb)
    user_dest_emb_norm = user_dest_emb / np.linalg.norm(user_dest_emb)

    group_interest_embs_norm = group_interest_embs / np.linalg.norm(group_interest_embs, axis=1, keepdims=True)
    group_dest_embs_norm = group_dest_embs / np.linalg.norm(group_dest_embs, axis=1, keepdims=True)

    # Compute cosine similarity between user embeddings and group embeddings
    sim_scores = np.dot(user_interest_emb_norm, group_interest_embs_norm.T) + np.dot(user_dest_emb_norm, group_dest_embs_norm.T)

    # Sort groups by similarity
    sorted_indices = np.argsort(sim_scores, axis=0)[::-1]
    sorted_group_interest_embs = group_interest_embs[sorted_indices]
    sorted_group_dest_embs = group_dest_embs[sorted_indices]
    sorted_sim_scores = sim_scores[sorted_indices]

    # Filter groups based on budget and travel style
    filtered_indices = []
    for i in range(len(sorted_sim_scores)):
        if sorted_sim_scores[i] > budget and travel_style == sorted_group_interest_embs[i].split(',')[0]:
            filtered_indices.append(i)

    # Get top 5 recommended groups
    top_5_indices = filtered_indices[:5]
    top_5_group_interest_embs = sorted_group_interest_embs[top_5_indices]
    top_5_group_dest_embs = sorted_group_dest_embs[top_5_indices]
    top_5_sim_scores = sorted_sim_scores[top_5_indices]

    # Compute average group embeddings for each recommendation
    avg_group_interest_emb = np.mean(top_5_group_interest_embs, axis=0)
    avg_group_dest_emb = np.mean(top_5_group_dest_embs, axis=0)

    # Compute final recommendation scores
    final_scores = np.dot(avg_group_interest_emb, user_interest_emb_norm) + np.dot(avg_group_dest_emb, user_dest_emb_norm)

    # Sort recommendations by score
    sorted_final_scores = np.argsort(final_scores, axis=0)[::-1]
    sorted_top_5_group_interest_embs = top_5_group_interest_embs[sorted_final_scores]
    sorted_top_5_group_dest_embs = top_5_group_dest_embs[sorted_final_scores]
    sorted_top_5_sim_scores = top_5_sim_scores[sorted_final_scores]

    # Get top 5 recommended groups
    top_5_recommended_groups = []
    for i in range(len(sorted_top_5_sim_scores)):
        top_5_recommended_groups.append({
            'Group': sorted_group_interest_embs[sorted_top_5_group_interest_embs[i]].split(',')[0],
            'SimScore': sorted_top_5_sim_scores[i],
            'AvgGroupInterestEmb': sorted_top_5_group_interest_embs[i],
            'AvgGroupDestEmb': sorted_top_5_group_dest_embs[i]
        })

    return top_5_recommended_groups
```

```
▶ recommend_groups_for_user(  
    age=23,  
    budget="High",  
    interests=["Hiking", "Nature"],  
    travel_style="Backpacking"  
)  
[47] Python  
... /usr/local/lib/python3.11/dist-packages/sklearn/utils/validation.py:2739: UserWarning: X does not have valid feature names, but OneHotEncoder was fitted  
warnings.warn(  
/usr/local/lib/python3.11/dist-packages/sklearn/utils/validation.py:2739: UserWarning: X does not have valid feature names, but OneHotEncoder was fitted  
warnings.warn(  
/usr/local/lib/python3.11/dist-packages/sklearn/utils/validation.py:2739: UserWarning: X does not have valid feature names, but MinMaxScaler was fitted  
warnings.warn(  
...  


|     | Group_ID | Group_Interest          | Destinations_Planned                               | Current_Members | Score    |
|-----|----------|-------------------------|----------------------------------------------------|-----------------|----------|
| 19  | G0020    | [Nature Walks, Surfing] | [Jaffna Fort, Polonnaruwa, Dagoba of Thuparama...] | 8               | 0.151932 |
| 118 | G0119    | [Sightseeing, Wildlife] | [Mount Lavinia Beach, National Zoological Gard...] | 9               | 0.151932 |
| 74  | G0075    | [Meditation, Hiking]    | [Negombo Beach, Abhayagiri Dagaba, Mount Lavin...] | 6               | 0.151932 |


```



# COMPLETION OF THE PROJECT

90%

# KEY FUNCTIONALITIES

- **Personalized Group Recommendations:** Uses a trained LightGBM ranker with encoded user data to suggest the best-fit travel groups.
- **Machine Learning-Powered Matching:** Employs feature scaling and encoders along with a semantic ranker to make accurate group predictions.
- **Dynamic Group Creation:** If no suitable group exists, a user can create a new group and wait for other members to join.
- **AI Assistant (TripBot) with Multi-Intent Classification:** TripBot leverages an SBERT-based intent classifier and extractors to understand queries and provides contextual travel suggestions inside group chats.
- **Interactive Group Chat Features:** Enriched chat experience with polls, reactions, file sharing, translations, and TripBot integration—making group planning collaborative and engaging.

# INTENT CLASSIFIER MODEL

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	message	intent														
2	Good evening folks!	greet														
3	Had great street food in Colombo!	share_experience														
4	Shut up man	abuse_or_noise														
5	I am free first weekend of next month	plan_trip														
6	Is Hinkleya crowded in December?	ask_help														
7	Is anyone interested in a trip to Dambulla?	plan_trip														
8	Pettah shopping was fun chaos	share_experience														
9	I'd love to go to Kandy next week	plan_trip														
10	Do we need to pre-book tickets for Peraliya?	ask_help														
11	How about a historical tour in Anuradhapura?	plan_trip														
12	Let's organize a food tour in Colombo	plan_trip														
13	What's the best route to Ella?	ask_help														
14	You're dumb	abuse_or_noise														
15	Took the scenic train to Badulla, unforgettable	share_experience														
16	I'm excited!	general														
17	Ambuluwawa Tower was totally worth it	share_experience														
18	How about a trip to Nuwara Eliya?	plan_trip														
19	My favorite trip was to Haputale	share_experience														
20	Hey team	greet														
21	Let's visit a waterfall spot this month	plan_trip														
22	Want an adventure group hike	plan_trip														
23	Any cool holidays near Gregory Lake?	ask_help														
24	Adventure weekend in Knuckles?	plan_trip														
25	Hey everyone!	greet														
26	I hiked Little Adam's Peak last year	share_experience														
27	How about visiting Horton Plains?	plan_trip														

```
import pandas as pd
import joblib

# Load dataset
df = pd.read_excel("/content/drive/My Drive/dataset/message-intent.xlsx")
x = df["message"].astype(str).tolist()
y = df["intent"].astype(str).tolist()

# SBERT embeddings
model = SentenceTransformer("all-MiniLM-L6-v2")
X_vectors = model.encode(X)

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X_vectors, y, test_size=0.2, random_state=42)

# Train classifier
clf = LogisticRegression(max_iter=1000)
clf.fit(X_train, y_train)

# Evaluate
preds = clf.predict(X_test)
print("Accuracy:", accuracy_score(y_test, preds))

# Save model
joblib.dump(clf, "/content/drive/My Drive/dataset/sbert_intent_classifier.pkl")
```

/usr/local/lib/python3.11/dist-packages/torch/nn/modules/module.py:1750: FutureWarning: 'encoder\_attention\_mask' is deprecated and will be removed in version 4.55.0 for BertSdpA  
return forward\_call(\*args, \*\*kwargs)  
Accuracy: 0.906976744186465  
['/content/drive/My Drive/dataset/sbert\_intent\_classifier.pkl']

# RECOMMENDER.PY

```
◆ recommender.py 2.0 X
collaborative > ml > recommender.py > load_groups_from_db
4  from sentence_transformers import SentenceTransformer
5  import lightgbm as lgb
6  from database.database import db
7  from typing import List, Optional
8
9  # Load models
10 model = lgb.Booster(model_file="TrainedModel/trained_lgbm_ranker_semantic.txt")
11 budget_encoder = joblib.load("TrainedModel/budget_encoder.pkl")
12 style_encoder = joblib.load("TrainedModel/style_encoder.pkl")
13 scaler = joblib.load("TrainedModel/feature_scaler.pkl")
14 sbert = SentenceTransformer('all-MiniLM-L6-v2')
15
16 # Load groups from MongoDB
17 async def load_groups_from_db():
18     groups = await db.groups.find().to_list(length=None)
19
20     if not groups:
21         print("X No groups found in database.")
22         return pd.DataFrame() # safe empty return
23
24 df = pd.DataFrame(groups)
25 print("✓ Loaded DataFrame columns:", df.columns.tolist())
26
27 if 'Group_Interest' in df.columns:
28     df['Group_Interest'] = df['Group_Interest'].fillna('').apply(
29         lambda x: [i.strip() for i in x] if isinstance(x, list) else []
30     )
31 else:
32     print("▲ 'Group_Interest' missing from DataFrame. Adding empty lists.")
33     df['Group_Interest'] = [[] for _ in range(len(df))]
34
35 if 'Destinations_Planned' in df.columns:
36     df['Destinations_Planned'] = df['Destinations_Planned'].fillna('').apply(
37         lambda x: [i.strip() for i in x] if isinstance(x, list) else []
38     )
39 else:
40     print("▲ 'Destinations_Planned' missing. Adding empty lists.")
41     df['Destinations_Planned'] = [[] for _ in range(len(df))]
```

```
◆ recommender.py 2.0 X
collaborative > ml > recommender.py > load_groups_from_db
46  async def recommend_groups_for_user(age, budget, interests, preferred_destinations=None, travel_style="Unknown", exclude_ids=None):
47      user_interest_text = ", ".join(interests)
48      user_interest_emb = sbert.encode([user_interest_text])[0]
49
50      if preferred_destinations:
51          user_dest_text = ", ".join(preferred_destinations)
52          user_dest_emb = sbert.encode([user_dest_text])[0]
53      else:
54          user_dest_emb = np.zeros(384)
55
56      group_interest_embs = sbert.encode(group_df['Group_Interest'].apply(lambda x: ', '.join(x)).tolist())
57      group_dest_embs = sbert.encode(group_df['Destinations_Planned'].apply(lambda x: ', '.join(x)).tolist())
58
59      interest_sims = np.dot(group_interest_embs, user_interest_emb) / (
60          np.linalg.norm(group_interest_embs, axis=1) * np.linalg.norm(user_interest_emb))
61
62      dest_sims = np.dot(group_dest_embs, user_dest_emb) / (
63          np.linalg.norm(group_dest_embs, axis=1) * np.linalg.norm(user_dest_emb))
64
65      if preferred_destinations else np.zeros(len(group_df))
66
67      budget_encoded = budget_encoder.transform([[budget]])
68      style_encoded = style_encoder.transform([[travel_style]])
69      age_scaled = np.repeat(scaler.transform([[age]]), len(group_df), axis=0)
70      members_scaled = scaler.transform(group_df[['Current_Members']])[:, 0]
71
72      X_new = np.hstack([
73          age_scaled.reshape(-1, 1),
74          np.repeat(budget_encoded, len(group_df), axis=0),
75          np.repeat(style_encoded, len(group_df), axis=0),
76          interest_sims.reshape(-1, 1),
77          dest_sims.reshape(-1, 1),
78          members_scaled.reshape(-1, 1)
79      ])
80
81      scores = model.predict(X_new)
82      group_df['Score'] = scores
83      top_6 = group_df.sort_values(by='Score', ascending=False).head(6)
84
85      return top_6[['Group_ID', 'Group_Name', 'Budget', 'Group_Interest', 'Destinations_Planned', 'Travel_Style', 'Current_Membe
```

# GROUP CHAT IMPLEMENTATION

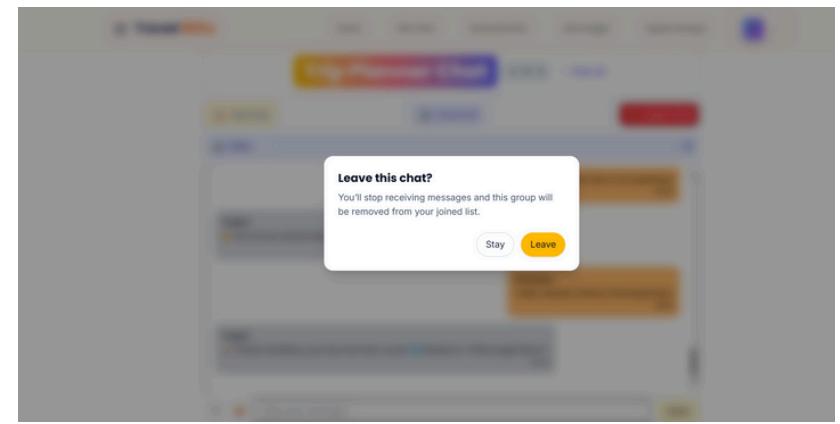
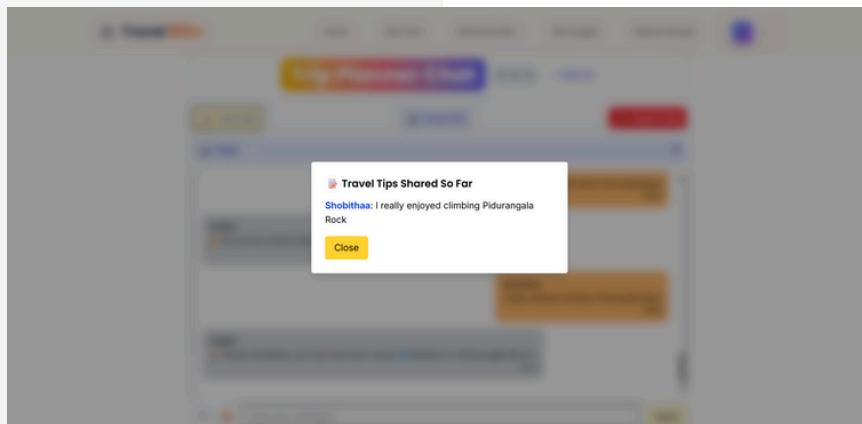
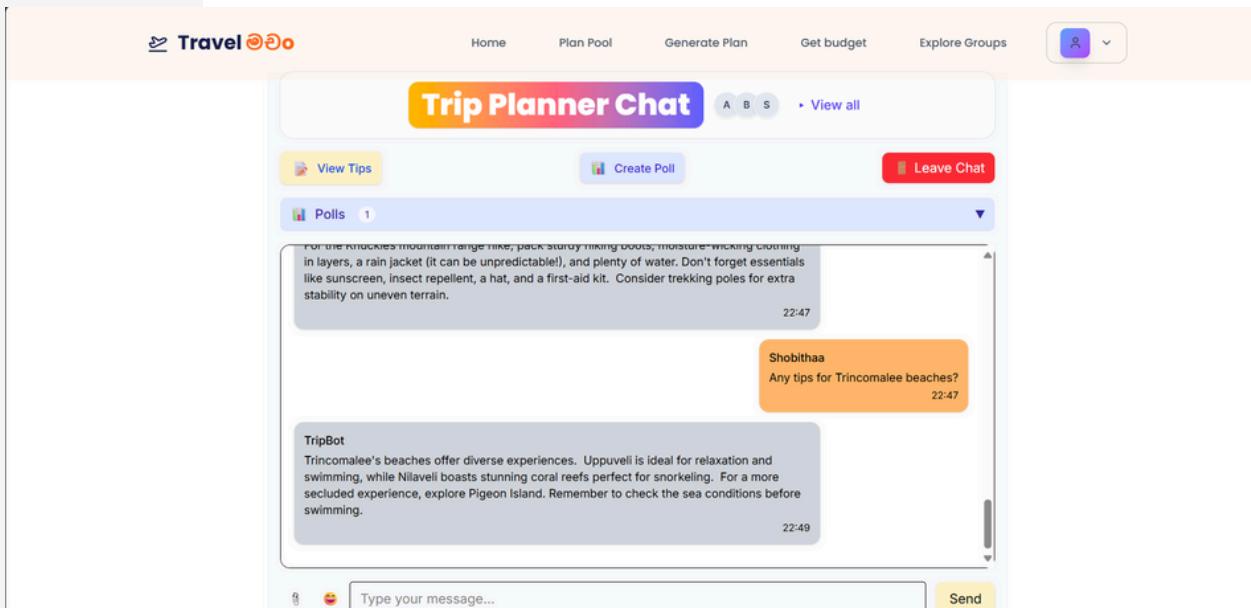
```
chat_ws.py 3, U
collaborative > routes > chat_ws.py > ...
36
37 UPLOAD_DIR = "uploads"
38 os.makedirs(UPLOAD_DIR, exist_ok=True)
39
40
41 def predict_intent(message: str) -> str:
42     vector = sbert_model.encode([message])
43     return clf.predict(vector)[0]
44
45
46 router = APIRouter()
47 active_connections: Dict[str, List[WebSocket]] = {}
48
49 # Group context reference
50 group_context: Dict[str, Dict] = {}
51
52 # Temporarily buffer experience messages per group/user
53 experience_buffer: Dict[str, Dict[str, Dict]] = {}
54
55
56 # Connect user to group
57 async def connect_user(group_id: str, websocket: WebSocket):
58     await websocket.accept()
59     if group_id not in active_connections:
60         active_connections[group_id] = []
61     active_connections[group_id].append(websocket)
62
63
64 # Disconnect user
65 def disconnect_user(group_id: str, websocket: WebSocket):
66     if group_id in active_connections:
67         active_connections[group_id].remove(websocket)
68
69
70 # Broadcast message to all users in a group
71 async def broadcast_message(group_id: str, message: dict):
72     for connection in active_connections.get(group_id, []):
73         await connection.send_json(message)
74
75
```

```
chat_ws.py 3, U
collaborative > routes > chat_ws.py > ...
77 def clean_mongo(obj):
78     for k, v in obj.items():
79         if isinstance(v, list):
80             return [clean_mongo(i) for i in v]
81     return obj
82
83
84 @asyncio.coroutine
85 def flush_experience_after_silence(group_id, user_id, username, bot, timeout=35):
86     await asyncio.sleep(timeout)
87     group_buf = experience_buffer.get(group_id, {})
88     user_buf = group_buf.get(user_id)
89     if not user_buf or user_buf.get("intent") != "share_experience":
90         return # Don't flush if it's not meant to be experience
91
92     if user_buf and user_buf["messages"]:
93         combined = "\n".join(user_buf["messages"])
94
95         # Extract all destinations and activities
96         destinations = extract_destination(combined)
97         activities = extract_activity(combined)
98
99         # Save to context
100        await bot.record_experience(username, combined, destinations, activities)
101
102        # Build rich response
103        response = f"\ud83d\udcbb Thanks {username}, your tips have been saved!"
104        if destinations:
105            response += f"\ud83d\udcbb Related to {', '.join(f'**{d}**' for d in destinations)}."
106        if activities:
107            response += f"\ud83d\udcbb Tagged as {', '.join(f'**{a}**' for a in activities)}."
108
109        await send_bot_message(group_id, response)
110
111
112    # Reset buffer
113    experience_buffer[group_id][user_id] = {
114        "messages": [],
115        "last_message_time": datetime.utcnow(),
116    }
117
118
119
```

# WORKING PRODUCT

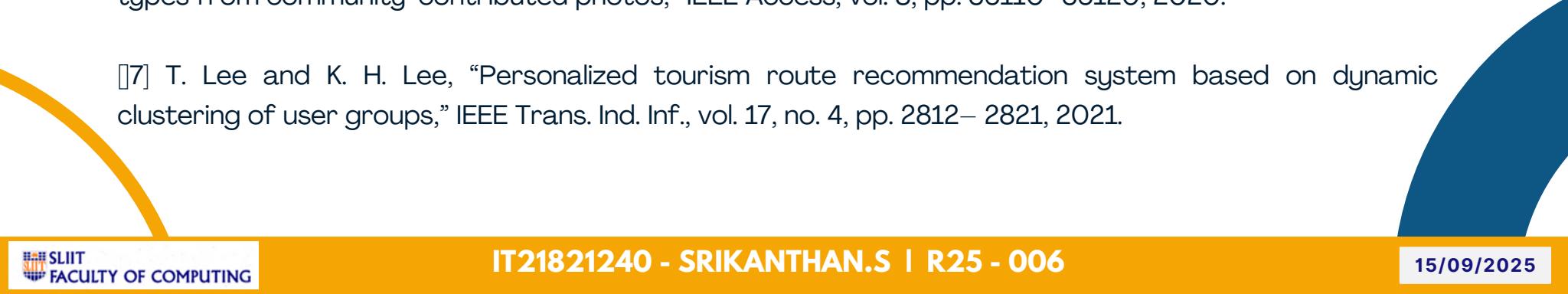
The screenshot displays the homepage of the Travel ඔබo website. At the top, there is a navigation bar with links for Home, Plan Pool, Generate Plan, Get budget, Explore Groups, and a search bar. Below the navigation bar, a large orange header banner reads "Find your perfect travel group". A sub-instruction below the banner says: "Based on your preferences. Join an active group or create a new one and we'll match additional members automatically." Under this, a section titled "Your Joined Groups" shows a card for "Beach Hopping Relaxed Low..." with interests in Beach Hopping, Cultural Activities, Cultural Exploration, Guided Tours, Hiking, Kite Surfing, and Style: Relaxed. A yellow "Go to Chat" button is at the bottom. To the right, a modal window titled "Join inactive group?" asks if the user wants to continue joining an inactive group, with "Cancel" and "Continue" buttons. Below these sections, a larger area titled "Recommended Groups" lists several groups with their names, budgets, interests, related destinations, styles, and "Join Group" buttons. The groups listed are Spirited Questers (High Budget), Radiant... (Medium Budget), Vibrant Navigators (Low Budget), Serene Questers (High Budget), and Radiant Drifters (High Budget). At the bottom, there are buttons for "Create New Group" and "Recommend More".

# WORKING PRODUCT





# REFERENCES

- [1] J. Zhang, B. Chen, and M. Zhang, "A group travel recommender system based on group approximate constraint satisfaction," in Proc. IEEE Int. Conf. on Artificial Intelligence and Computer Applications (ICAICA), 2021.
  - [2] Y. Liu and X. Wang, "A systematic review of group recommender systems techniques," IEEE Access, vol. 9, pp. 45327–45340, 2021.
  - [3] S. Li, J. Zhao, and H. Wang, "An intelligent personalized recommendation for travel group planning based on reviews," in Proc. IEEE Int. Conf. on Information Technology Applications (ITA), 2022.
  - [4] A. N. Mirza and S. Rehman, "User profiling in travel recommender system using hybridization and collaborative method," in Proc. IEEE Int. Conf. on Communication, Computing and Signal Processing (ICCCSP), 2021.
  - [5] M. Z. Khan, M. R. Hossain, and A. I. Haque, "Collaborative filtering algorithm in the design of intelligent tourism service robots," in Proc. IEEE Int. Conf. on Computer and Information Technology (ICCIT), 2020.
  - [6] Y. Sun, J. Xu, and W. Wang, "Travel recommendation by mining people attributes and travel group types from community-contributed photos," IEEE Access, vol. 8, pp. 56110–56120, 2020.
  - [7] T. Lee and K. H. Lee, "Personalized tourism route recommendation system based on dynamic clustering of user groups," IEEE Trans. Ind. Inf., vol. 17, no. 4, pp. 2812– 2821, 2021.
- 



**IT21831768 - SENEVIRATHNE.S.D.C.D**

**SPECIALIZING - SOFTWARE ENGINEERING**





# AGENDA

01. BACKGROUND
02. RESEARCH QUESTIONS
03. OBJECTIVES
04. SYSTEM DIAGRAM





# BACKGROUND

## RECOMMENDATION SYSTEM

---

- Travelers often struggle to choose where to go, we use user demographics and preferences to recommend locations.
- Traditional systems offer little personalization.
- Combining multiple recommendation approaches improves relevance and accuracy.

## PLANNER SYSTEM

---

- Efficient route planning is essential for tourism in Sri Lanka.
- Province-based optimization reduces travel time and wasted effort.
- Finally Optimize based on the distance of locations.

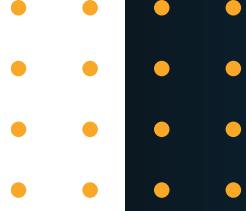
## ITINERARY SWITCHER

---

- Replace the generated route with smarter alternative itineraries.
- Swap stops with locations that offer similar activities.

# RESEARCH Problems?

---



01

How can a hybrid recommender (CBF + CF + ML) that uses user demographics and preferences outperform single-method baselines for top-N location suggestions in Sri Lanka?

02

To what extent does province-based route optimization reduce total travel time and backtracking compared with distance-only or greedy planners?

03

Can the itinerary switcher replace stops with similar-activity locations while keeping routes feasible and efficient, and how do users rate these swaps ?



# OBJECTIVES

Generate Personalized Recommendations

Efficient Plan Optimization

Suggest Smart Alternatives

Enhance User Experience

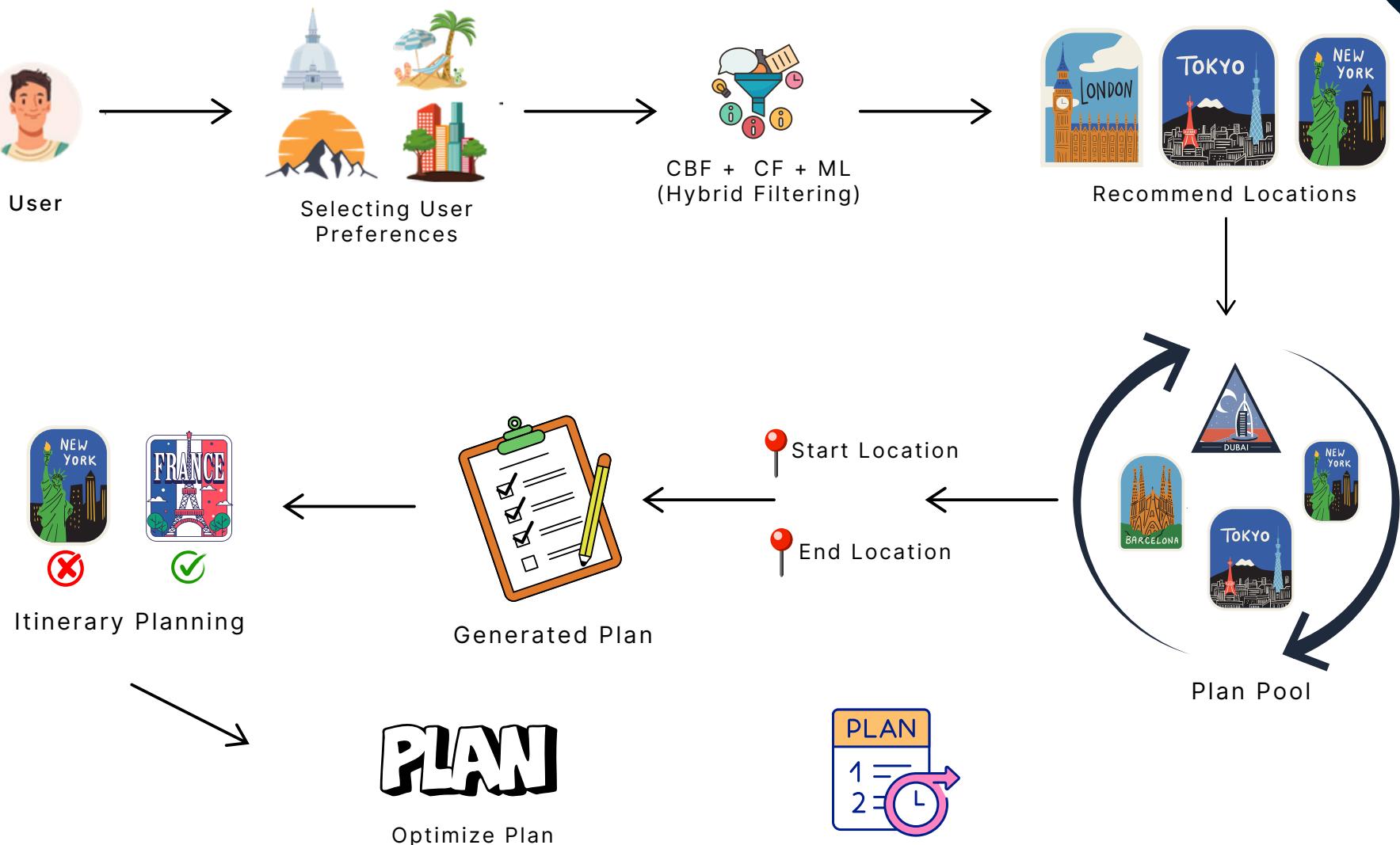


# METHODOLOGY

- 01. SYSTEM DIAGRAM
- 02. TOOL & TECHNOLOGIES
- 03. REQUIREMENTS



# SYSTEM DIAGRAM



# TOOL AND TECHNOLOGIES



## Programming Languages

Python ,  
JavaScript



## Project Management

Jlra



## Database

MongoDB



## Frameworks & Libraries

Pandas, Spacy, React,  
Scikit-learn, mantine



## Other Tools

Git , Draw.io Postman,  
Docker

# REQUIREMENTS

## FUNCTIONAL

- User Demographic data collection and Preference collection.
- Real-time recommendation generation based on user inputs.
- Provide Itineraries based on rating, nearby and smart alternatives.
- Ability to optimize the plan after the itinerary changes

## NON-FUNCTIONAL

- Performance – Recommending locations under 10 seconds for the user.
- Portability – Containerized docker environment.
- Security – Protect User demographic and sensitive data

# GITHUB COMMITS

-o- Commits on Sep 7, 2025

- plan generation , itinerary options and switching and optimization initila implementaion** 6e5b841 ⌂ ↗  
last-chaanz committed last week
- Merge pull request #8 from SLIIT-Projects-Y4/feature/dashboard** a57a91b ⌂ ↗  
last-chaanz authored last week Verified
- frontendfiles update** 01591dd ⌂ ↗  
last-chaanz committed last week
- planner update** 30d4cf6 ⌂ ↗  
last-chaanz committed last week
- api updates frontend** 98ba8e7 ⌂ ↗  
last-chaanz committed last week
- model update** b28b59d ⌂ ↗  
last-chaanz committed last week
- small update** 92127cb ⌂ ↗  
last-chaanz committed last week
- backend file rpoery adding and modifications** 2e9a664 ⌂ ↗  
last-chaanz committed last week
- plan itinerary and geo updates and implementations** 3fc0d3e ⌂ ↗  
last-chaanz committed last week

-o- Commits on Sep 6, 2025



# COMPLETION OF THE PROJECT

## 50%

# DATA COLLECTION

User_ID	User_Country	User_Age_Group	Gender	Travel_Companion	Location_Type	Preferred_Activities
User 1	Australia	36-50	Male	Family	['Nature & Wildlife Areas', 'Gardens', 'Beaches']	['Wildlife Safaris', 'Fishing', 'Factory Tours', 'Hiking']
User 2	Australia	26-35	Male	Friends	['Nature & Wildlife Areas', 'National Parks', 'Beaches']	['Cultural Tours', 'Boat Tours', 'Sightseeing', 'Snorkeling']
User 3	Australia	18-25	Male	Friends	['National Parks', 'Beaches']	['Boat Tours', 'Jet Skiing', 'Hiking', 'Rafting']
User 4	Portugal	18-25	Male	Solo	['Museums', 'Farms', 'Beaches']	['Yoga', 'Photography', 'Jet Skiing', 'Snorkeling']
User 5	Italy	18-25	Male	Family	['Religious Sites', 'Historic Sites', 'Beaches']	['Rafting', 'Wildlife Spotting', 'Hiking', 'Jet Skiing']
User 6	United Arab Emirates	18-25	Female	Solo	['Historic Sites', 'Beaches']	['Beach Hopping', 'Nature Walks', 'Photography', 'Hiking']
User 7	Indonesia	36-50	Male	Friends	['Historic Sites', 'Beaches']	['Historical Exploration', 'Hiking', 'Walking Trails', 'Bird Watching']
User 8	Canada	18-25	Female	Solo	['Historic Sites', 'Beaches', 'Bodies of Water']	['Nature Walks', 'Photography', 'Beach Hopping', 'Snorkeling']
User 9	New Zealand	36-50	Male	Family	['Beaches', 'Historic Sites', 'Farms']	['Agricultural Workshops', 'Historical Exploration', 'Wildlife Safaris', 'Hiking']
User 10	United Kingdom	51+	Female	Couple	['Museums', 'Historic Sites', 'Beaches']	['Cultural Activities', 'Meditation', 'Relaxation']
User 11	United Kingdom	18-25	Female	Friends	['Waterfalls', 'Beaches', 'Bodies of Water']	['Boat Tours', 'Beach Hopping', 'Snorkeling', 'Surfing']
User 12	Japan	36-50	Male	Family	['Gardens', 'Beaches']	['Fishing', 'Historical Exploration', 'Wildlife Safaris', 'Hiking']
User 13	Australia	26-35	Female	Family	['Historic Sites', 'Beaches']	['Photography', 'Yoga', 'Nature Walks']
User 14	United Kingdom	26-35	Male	Family	['Religious Sites', 'Beaches']	['Hiking', 'Sightseeing', 'Cultural Tours']
User 15	Germany	18-25	Female	Couple	['Beaches', 'Zoological Gardens']	['Relaxation', 'Yoga', 'Beach Hopping', 'Sunbathing']
User 16	Ukraine	18-25	Female	Solo	['Nature & Wildlife Areas', 'Beaches']	['Photography', 'Hiking', 'Nature Walks', 'Yoga']
User 17	United Kingdom	26-35	Female	Couple	['National Parks', 'Beaches']	['Sunbathing', 'Meditation', 'Yoga']
User 18	India	26-35	Male	Family	['Beaches', 'Nature & Wildlife Areas', 'Religious Sites']	['Wildlife Spotting', 'Factory Tours', 'Cultural Tours']
User 19	United Kingdom	51+	Male	Friends	['Waterfalls', 'Historic Sites', 'Beaches', 'Museums', 'Gardens']	['Meditation', 'Wildlife Safaris', 'Worship']
User 20	Viet Nam	51+	Female	Friends	['Religious Sites', 'National Parks', 'Beaches']	['Cultural Exploration', 'Gardens', 'Worship', 'Nature Walks']

Location_ID	Location_Name	Located_City	Located_Province	Location_Type	Latitude	Longitude	Avg_Rating	Review_Count	Activities
LOC_1	Arugam Bay	Arugam Bay	Eastern Province	Beaches	6.8379771	81.8251687	4.06	187	['Swimming', 'Surfing', 'Sunbathing', 'Yoga', 'Boat Tours']
LOC_2	Bentota Beach	Bentota	Southern Province	Beaches	6.4234707	79.9953199	4.59	233	['Swimming', 'Surfing', 'Sunbathing', 'Snorkeling']
LOC_3	Hikkaduwa Beach	Hikkaduwa	Southern Province	Beaches	6.1377266	80.0990596	4.02	213	['Swimming', 'Surfing', 'Sunbathing', 'Snorkeling']
LOC_4	Jungle Beach	Unawatuna	Southern Province	Beaches	6.0186943	80.2394104	3.54	183	['Swimming', 'Sunbathing', 'Snorkeling', 'Hiking']
LOC_5	Kalutara Beach	Kalutara	Western Province	Beaches	6.6098466	79.9474179	3.67	218	['Swimming', 'Surfing', 'Sunbathing', 'Snorkeling']
LOC_6	Marble Beach	Trincomalee	Eastern Province	Beaches	7.7853051	81.4278984	3.97	145	['Swimming', 'Surfing', 'Sunbathing', 'Snorkeling']
LOC_7	Mirissa Beach	Mirissa	Southern Province	Beaches	5.9447028	80.4591608	4.27	217	['Swimming', 'Surfing', 'Sunbathing', 'Snorkeling']
LOC_8	Mount Lavinia Beach	Colombo	Western Province	Beaches	6.842362	79.8624086	4.13	180	['Swimming', 'Sunbathing', 'Kite Surfing']
LOC_9	Negombo Beach	Negombo	Western Province	Beaches	7.2055208	79.8512562	3.12	209	['Swimming', 'Surfing', 'Sunbathing', 'Snorkeling']
LOC_10	Nilaveli Beach	Nilaveli	Eastern Province	Beaches	8.7003072	81.19205	4.33	165	['Swimming', 'Surfing', 'Sunbathing', 'Snorkeling']
LOC_11	Passikudah Beach	Kalkudah	Eastern Province	Beaches	7.9299939	81.5611852	4.56	160	['Swimming', 'Surfing', 'Sunbathing', 'Snorkeling']
LOC_12	Gregory Lake	Nuwara Eliya	Central Province	Bodies of Water	6.95863634	80.77845098	3.96	256	['Boating', 'Walking Trails']
LOC_13	Kandy Lake	Kandy	Central Province	Bodies of Water	7.2912017	80.6420565	3.94	301	['Boating', 'Walking Trails']
LOC_14	Tissa Wewa	Tissamaharama	Southern Province	Bodies of Water	6.2895818	81.2821647	4.43	91	['Historical Exploration', 'Architectural Marvel']
LOC_15	Twin Baths (Kuttam Pokuna)	Anuradhapura	North Central Province	Bodies of Water	8.1995638	80.6326916	3.89	191	['Historical Exploration', 'Architectural Marvel']
LOC_16	Ambewela Farms	Nuwara Eliya	Central Province	Farms	6.8917934	80.8038646	3.9	136	['Factory Tours', 'Agricultural Workshops']
LOC_17	Bluefield Tea Gardens	Nuwara Eliya	Central Province	Farms	7.2564996	80.7214417	4.22	314	['Factory Tours', 'Agricultural Workshops']
LOC_18	Dambatenne Tea Factory	Haputale	Uva Province	Farms	6.7833795	81.0034028	3.36	172	['Factory Tours', 'Agricultural Workshops']
LOC_19	Damro Labookellie Tea Centre	Nuwara Eliya	Central Province	Farms	7.2564996	80.7214417	4.04	90	['Factory Tours', 'Agricultural Workshops']
LOC_20	Glenloch Tea Factory	Katukitula	Central Province	Farms	7.0771474	80.675667	3.82	302	['Factory Tours', 'Agricultural Workshops']



# RECOMMENDATION SYSTEM

a). Three-tier approach

- Content-Based: Matches user activities with location offerings (40% weight)
- Collaborative: Leverages similar travelers' ratings (35% weight)
- Machine Learning: Predicts ratings using XGBoost model (25% weight)

b). Adaptive weighting adjusts based on available data

## XGBOOST

```
def _train_enhanced_model(self):
    """Train an enhanced XGBoost model with more features and hyperparameter tuning"""
    # Encode locations for ML
    location_encoder = LabelEncoder()
    location_encoder.fit(self.reviews_df['Location_Name'])
    self.reviews_df['Location_Name_encoded'] = location_encoder.transform(self.reviews_df['Location_Name'])
    self.locations_df['location_Name_encoded'] = location_encoder.transform(self.locations_df['Location_Name'])

    # Prepare feature set - include all encoded features and engineered features
    features = ['User_Age_Group_encoded', 'Gender_encoded', 'Travel_Companion_encoded', 'Location_Name_encoded']

    # Add location type if available
    if 'Location_Type_encoded' in self.reviews_df.columns:
        features.append('Location_Type_encoded')

    # Add user country if available
    if 'User_Country_encoded' in self.reviews_df.columns:
        features.append('User_Country_encoded')

    # Add user rating behavior features if available
    for col in ['Avg_Rating', 'Rating_Count', 'Rating_Variability']:
        if col in self.reviews_df.columns:
            features.append(col)

    # Add activity features
    features.append('Activity_Match_Ratio')
    features.append('Activity_Count')

    # Add activity-specific features
    for col in self.reviews_df.columns:
        if col.startswith('Has_'):
            features.append(col)
```

```
XGBoost model performance: RMSE=0.4891, R2=0.7655
Interpretation:
- RMSE: Average prediction error is 0.49 stars
- R2: Model explains 0.77 of rating variance (excellent)
```

## RANDOM FOREST

```
def _train_rf_model(self):
    """Train a Random Forest model with hyperparameter tuning"""
    start_time = time.time() # Start measuring training time

    # Encode locations for ML
    location_encoder = LabelEncoder()
    location_encoder.fit(self.reviews_df['Location_Name'])
    self.reviews_df['Location_Name_encoded'] = location_encoder.transform(self.reviews_df['Location_Name'])
    self.locations_df['location_Name_encoded'] = location_encoder.transform(self.locations_df['Location_Name'])

    # Prepare feature set - include all encoded features and engineered features
    features = ['User_Age_Group_encoded', 'Gender_encoded', 'Travel_Companion_encoded', 'Location_Name_encoded']

    # Add location type if available
    if 'Location_Type_encoded' in self.reviews_df.columns:
        features.append('Location_Type_encoded')

    # Add user country if available
    if 'User_Country_encoded' in self.reviews_df.columns:
        features.append('User_Country_encoded')

    # Add user rating behavior features if available
    for col in ['Avg_Rating', 'Rating_Count', 'Rating_Variability']:
        if col in self.reviews_df.columns:
            features.append(col)

    # Add activity features
    features.append('Activity_Match_Ratio')
    features.append('Activity_Count')

    # Add activity-specific features
    for col in self.reviews_df.columns:
        if col.startswith('Has_'):
            features.append(col)
```

```
Random Forest model performance: RMSE=0.5235, R2=0.6835
Interpretation:
- RMSE: Average prediction error is 0.52 stars
- R2: Model explains 0.68 of rating variance (good)
```

## OUTPUT

```
=====
TOP RECOMMENDED LOCATIONS
=====

🌟 TOP PICK: Wilpattu National Park (📍 Nature & Wildlife Areas)
★ Overall Score: 0.86/1.0
📍 Activity Match: 3/4 activities
⭐ Ratings: CBF: 0.94, CF: 4.33, ML: 3.57
📍 Activities: 🦌 Wildlife Spotting, 🐦 Bird Watching, 🏞️ Nature Walks
📄 Based on 18 reviews
```



# LOCATION ITINERARY SYSTEM

---

- a). Smart Alternatives based on ratings and nearby.
- b). Nearby Alternatives based on the distance of the selected location.
- c). Top Rated Alternatives based on Travelers ratings.

```

# 1. Get details for start and end cities
start_info = get_city_details_combined(start_city, locations_dataset)
end_info = get_city_details_combined(end_city, locations_dataset)

start_province = start_info["province"]
end_province = end_info["province"]

print(f"\nStart: {start_info['name']} in {start_province}")
print(f"End: {end_info['name']} in {end_province}")

# Track locations to exclude from city attractions (to avoid duplication)
excluded_locations = [start_info["name"], end_info["name"]]

```

```

# 2. Find the province corridor (path of provinces between start and end)
province_corridor = find_province_path(start_province, end_province)
print(f"\nProvince corridor: {' → '.join(province_corridor)}")

```

```

# 3. Get plan pool locations by province corridor
corridor_plan = get_plan_pool_by_corridor(province_corridor, locations_dataset,
plan_pool_locations)

```

```

# Now build the itinerary, province by province
itinerary = []

# Start with start city
itinerary.append({
    "name": start_info["name"],
    "type": "City" if start_info.get("type") is None else start_info["type"],
    "province": start_province,
    "is_city": True,
    "lat": start_info["lat"],
    "lng": start_info["lng"]
})

# Current position is start city
current_lat, current_lng = start_info["lat"], start_info["lng"]

```

## OUTPUT

```

TRIP SUMMARY
=====
📍 FROM: Galle Fort (Southern Province)
📍 TO: Anuradhapura (North Central Province)
🔴 TOTAL DISTANCE: 309.9 km
⭐ ATTRACTIONS: 5
📍 ROUTE: 🇸ා Southern Province → 🇸ා Western Province → 🌳 North Western Province → 🏰 North Central Province

```



# PLANNER SYSTEM

- Get the start and end city and create the plan for the selected plan pool items for the given radius.
- Locations are matched based on the distance from the start city to end city.
- Province based route matching included.

```

# Convert numeric columns and fill missing values with 0
locations["Avg_Rating"] = pd.to_numeric(locations["Avg_Rating"], errors='coerce').fillna(0)
locations["Review_Count"] = pd.to_numeric(locations["Review_Count"], errors='coerce').fillna(0)
# Fill missing activities with an empty string
locations['Activities'] = locations['Activities'].fillna('')

# Normalize numerical features using MinMaxScaler
scaler = MinMaxScaler()
locations[["Avg_Rating", "Review_Count"]] = scaler.fit_transform(locations[["Avg_Rating",
    "Review_Count"]])

```

```

plan = get_itinerary_plan(
    location_name="Galle Fort",
    included_provinces=["Central Province", "Southern Province", "Western Province"],
    top_n=5
)

```

## OUTPUT

```

📍 LOCATION: Galle Fort
=====
📍 LOCATION DETAILS
-----
📍 Type:📍 Historic Sites
📍 Province:📍 Southern Province
📍 Rating:📍 ★★★★ (4.5/5)
📍 Activities:📍 ['Sightseeing',📍 'Photography',📍 'Learning History',📍 'Cultural Tours']

```

```

def recommend_by_activities(user_activities, top_n=5):
    # Clean user activities
    user_set = set([act.strip().lower() for act in user_activities])

    scores = {}
    for idx, row in locations.iterrows():
        loc_set = clean_activities(row["Activities"])
        if not user_set and not loc_set:
            sim = 1.0
        else:
            union = user_set.union(loc_set)
            sim = len(user_set.intersection(loc_set)) / len(union) if union else 0.0
        scores[row["Location_Name"]] = sim

    scores_df = pd.DataFrame(scores.items(), columns=["Location",
        "Activities_Jaccard_Similarity"])
    scores_df = scores_df.sort_values("Activities_Jaccard_Similarity", ascending=False)
    return scores_df.head(top_n)

```

### 📍 TOP RECOMMENDED PLACES (HYBRID SYSTEM)

- 📍 Sigiriya The Ancient Rock Fortress
  - 📍 Match:📍 0.66
  - 📍 Rating:📍 ★★★★ (4.7/5)
  - 📍 Province:📍 Central Province

### 📍 SIMILAR HISTORIC SITES LOCATIONS

- 📍 Sigiriya The Ancient Rock Fortress
  - 📍 Rating:📍 ★★★★ (4.7/5)
  - 📍 Province:📍 Central Province



# COMPLETION OF THE PROJECT

90%

# KEY FUNCTIONALITIES

- **Preference Collection & Personalization:** Collect age group, travel companion, and other user demographic details to deliver personalized suggestions.
- **AI-Powered Recommendations:** Uses XGBoost Model to suggest destinations and attractions that best match user preferences.
- **Trip Planning Workflow:** Generate trip planning workflow for the given start and end city with the saved travel locations.
- **Itinerary Management:** Provides similar alternatives and smart suggestion for generated plan locations.
- **Plan Optimization:** Optimize the trip planning route after the itinerary changes according to the distances of each location.

# WORKING PRODUCT

The screenshot shows the homepage of the Travel තුව website. At the top, there is a navigation bar with links for Home, Plan Pool, Category, Generate Plan, Get budget, and Explore Groups. Below the navigation bar, a section titled "Recommended for You" displays three travel destinations:

- Tissa Wewa** (Anuradhapura, North Central Province)  
A 2,000-year-old reservoir in Tissamaharama, Tissa Wewa exemplifies ancient Sri Lankan architecture.
- Twin Baths (Kuttam Pokuna)** (Anuradhapura, North Central Province)  
Located in Anuradhapura, Kuttam Pokuna ("Twin Ponds") are ancient bathing tanks crafted with...
- Gangaramaya (Vihara) Buddhist Temple** (Colombo, Western Province)  
Gangaramaya Temple in Colombo is a vibrant cultural and spiritual hub founded in the late 19th...

Each destination card includes a "Details" button and an "Add to Plan" button.

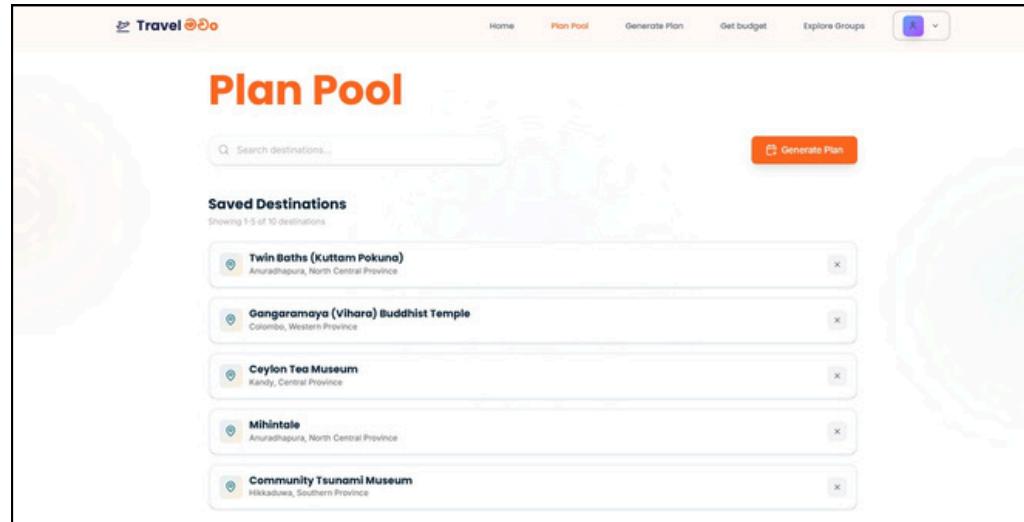
The screenshot shows a detailed view of the Gangaramaya (Vihara) Buddhist Temple page. The page title is "Gangaramaya (Vihara) Buddhist Temple" located in Colombo - Western Province. There is a "Add to Plan" button. Below the title, there is a category link "RELIGIOUS SITES". A large image of the temple complex reflected in water is displayed, along with a five-star rating and social sharing icons. The "Description" section provides information about the temple's history and architecture.

**Gangaramaya (Vihara) Buddhist Temple**  
Colombo - Western Province

**Description**

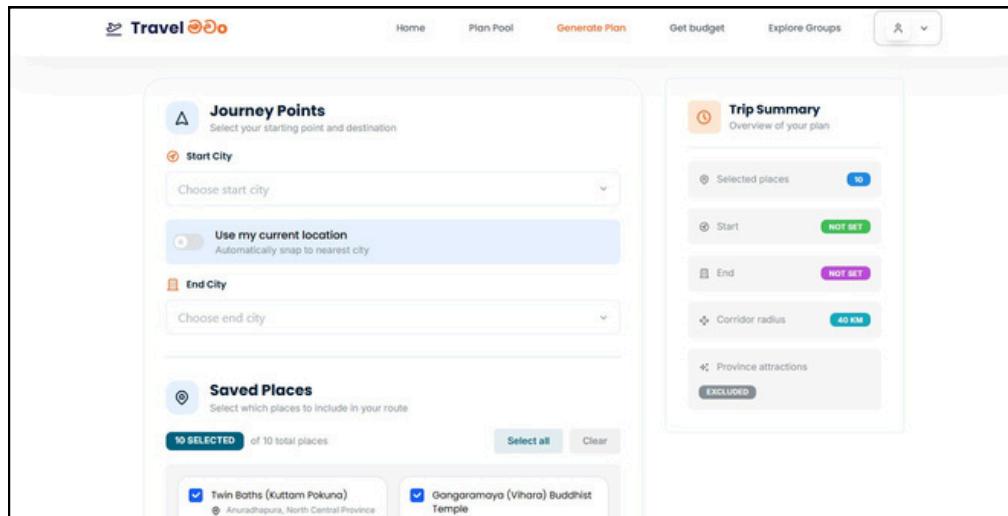
Gangaramaya Temple in Colombo is a vibrant cultural and spiritual hub founded in the late 19th century by Hikkaduwe Sri Sumangala Nayaka Thera. Situated near Beira Lake, it combines traditional Sri Lankan architecture with modern elements. The temple complex includes a museum, library, and educational center.

# WORKING PRODUCT



The screenshot shows the 'Plan Pool' section of the Travel Srilanka website. At the top, there is a search bar with the placeholder 'Search destinations...' and a red 'Generate Plan' button. Below the search bar, the heading 'Saved Destinations' is displayed, followed by the subtext 'Showing 1-5 of 10 destinations'. A list of five saved destinations is shown in a grid format:

- Twin Baths (Kuttam Pokuna) - Anuradhapura, North Central Province
- Gangaramaya (Vihara) Buddhist Temple - Colombo, Western Province
- Ceylon Tea Museum - Kandy, Central Province
- Mihintale - Anuradhapura, North Central Province
- Community Tsunami Museum - Hikkaduwa, Southern Province



The screenshot shows the 'Generate Plan' section of the Travel Srilanka website. It includes fields for 'Start City' and 'End City', both with dropdown menus. There is also a 'Use my current location' option. On the right, a 'Trip Summary' panel displays the following information:

- Selected places: 10
- Start: NOT SET
- End: NOT SET
- Corridor radius: 40 KM
- Province attractions: EXCLUDED

# WORKING PRODUCT

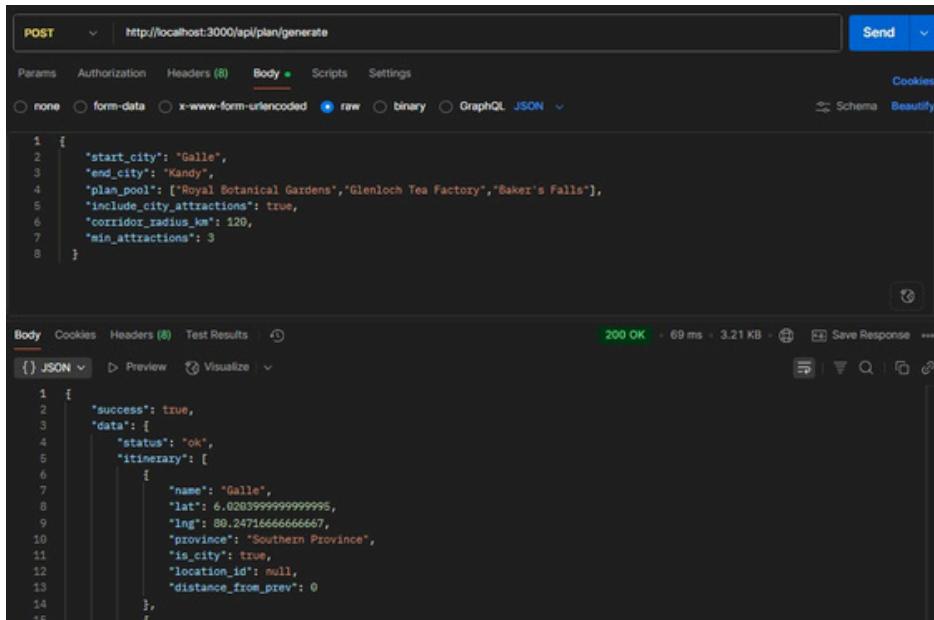
The screenshot shows the 'Beaches' category on the Travel තේඛ website. It features three cards for Arugam Bay, Bentota Beach, and Hikkaduwa Beach. Each card includes a photo, a rating of 5 stars, and a brief description. Buttons for 'Details' and 'Add to Plan' are visible at the bottom of each card.

Location	Description	Rating
Arugam Bay, Eastern Province	A crescent-shaped stretch of golden sand on Sri Lanka's eastern coast. Arugam Bay Beach is a...	5 stars
Bentota Beach, Southern Province	Bentota Beach on Sri Lanka's west coast combines historical significance with natural...	5 stars
Hikkaduwa Beach, Southern Province	Hikkaduwa Beach is a bustling coastal town in southern Sri Lanka famed for its vibrant nightlife...	5 stars

The screenshot shows the 'Saved Plans' section on the Travel තේඛ website. It displays a 'Saved Plan List' with four entries: Colombo -> Anuradhapura (40km corridor), Colombo -> Anuradhapura (100km corridor), Colombo -> Nuwara Eliya (100km corridor), and Colombo -> Kandy (100km corridor). To the right, a detailed route for the Colombo -> Anuradhapura (40km corridor) is shown, listing stops including Colombo, Gangaramaya (Vihara) Buddhist Temple, Tissa Wewa, Isurumuniya Temple, and Twin Baths (Kuttam Pokuna).

Route	Distance	Stops
Colombo -> Anuradhapura (40km corridor)	99.3 km	7 stops
Colombo -> Anuradhapura (100km corridor)	500.3 km	12 stops
Colombo -> Nuwara Eliya (100km corridor)	271.4 km	4 stops
Colombo -> Kandy (100km corridor)	554.4 km	8 stops

# POSTMAN API CHECKS



POST http://localhost:3000/api/plan/generate

Headers (8) Body Scripts Settings Cookies

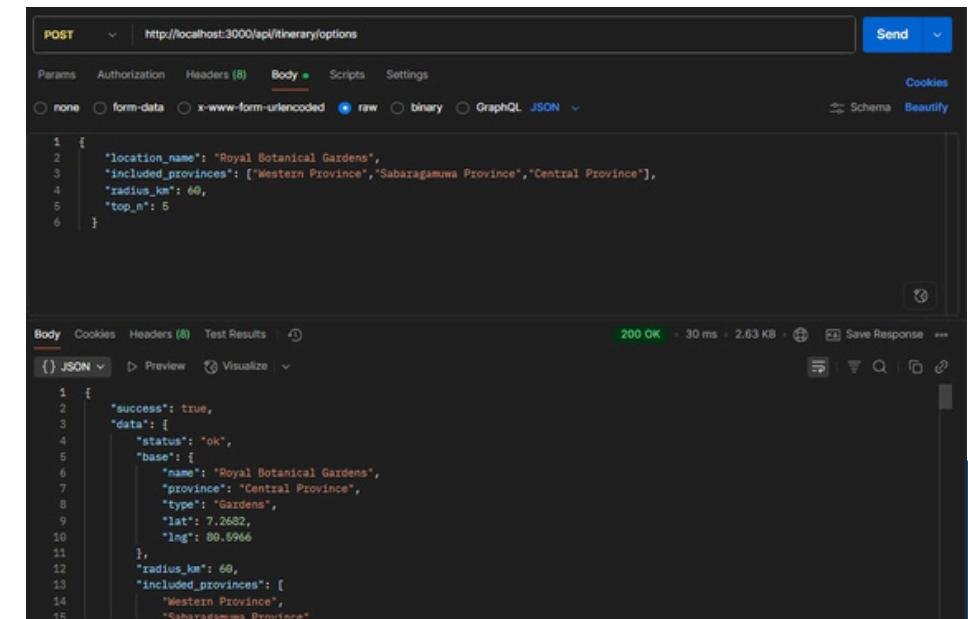
Body (JSON)

```
1 {
  "start_city": "Galle",
  "end_city": "Kandy",
  "plan_pool": ["Royal Botanical Gardens", "Glenloch Tea Factory", "Baker's Falls"],
  "include_city_attractions": true,
  "corridor_radius_km": 120,
  "min_attractions": 3
}
```

200 OK 69 ms 3.21 KB Save Response

Body (JSON) Preview Visualize

```
1 {
  "success": true,
  "data": [
    {
      "status": "ok",
      "itinerary": [
        {
          "name": "Galle",
          "lat": 6.020399999999999,
          "lng": 80.24716566666667,
          "province": "Southern Province",
          "is_city": true,
          "location_id": null,
          "distance_from_prev": 0
        }
      ]
    }
  ]
}
```



POST http://localhost:3000/api/itinerary/options

Headers (8) Body Scripts Settings Cookies

Body (JSON)

```
1 {
  "location_name": "Royal Botanical Gardens",
  "included_provinces": ["Western Province", "Sabaragamuwa Province", "Central Province"],
  "radius_km": 60,
  "top_n": 5
}
```

200 OK 30 ms 2.63 KB Save Response

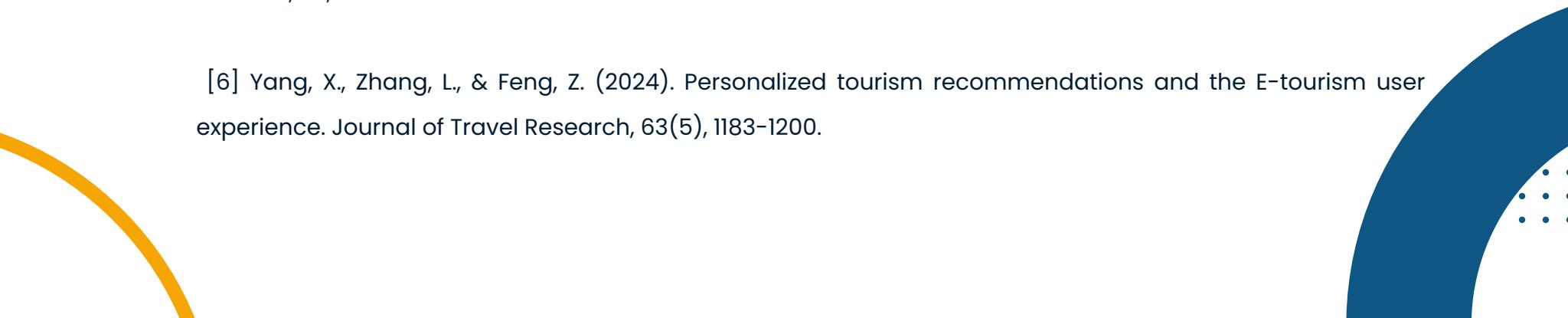
Body (JSON) Preview Visualize

```
1 {
  "success": true,
  "data": [
    {
      "status": "ok",
      "base": [
        {
          "name": "Royal Botanical Gardens",
          "province": "Central Province",
          "type": "Gardens",
          "lat": 7.2682,
          "lng": 80.5966
        }
      ],
      "radius_km": 60,
      "included_provinces": [
        "Western Province",
        "Sabaragamuwa Province"
      ]
    }
  ]
}
```



# REFERENCES



- [1] Overgoor, G., Chica, M., Rand, W., & Weishampel, A. (2019). Letting the computers take over: Using AI to solve marketing problems. *California Management Review*, 61(4), 156–185.
  - [2] Topol, E. J. (2019). High-performance medicine: the convergence of human and artificial intelligence. *Nature medicine*, 25(1), 44–56.
  - [3] Duarte, L., Torres, J., Ribeiro, V., & Moreira, I. (2020). Artificial Intelligence Systems applied to tourism: A Survey. arXiv preprint arXiv:2010.14654.
  - [4] <https://wttc.org/LinkClick.aspx?fileticket=2VluL2e-nAA%3D&portalid=0&utm>
  - [5] Gretzel, U., Sigala, M., Xiang, Z., & Koo, C. (2015). Smart tourism: foundations and developments. *Electronic markets*, 25, 179–188.
  - [6] Yang, X., Zhang, L., & Feng, Z. (2024). Personalized tourism recommendations and the E-tourism user experience. *Journal of Travel Research*, 63(5), 1183–1200.
- 



**IT21835728 - THUWAKARAN. R**

**SPECIALIZING - SOFTWARE ENGINEERING**





# INTRODUCTION

01. BACKGROUND
02. RESEARCH QUESTIONS
03. MAIN AND SUB OBJECTIVES
04. METHODOLOGY
05. COMPLETION OF PROJECT



# BACKGROUND



## Static Budgeting Tools

- Traditional budgeting tools rely on outdated, static data and manual processes, which don't adjust to fluctuating travel costs or changing user preferences.

## Lack of Personalization

- Current platforms offer generalized recommendations that don't meet the unique needs of travelers, such as varying group sizes, budgets, and dynamic itineraries.

## Limited Real-Time Adaptation

- Without real-time data integration or dynamic adaptability, platforms fail to adjust to unforeseen changes like weather, price fluctuations, or schedule conflicts, leaving travelers with inefficient plans.

# RESEARCH Questions?

---



01

What are the limitations of current tools for travel budget allocation?

02

How can predictive algorithms and real-time data improve the accuracy and reliability of travel budgeting?

03

How can user-centric design enhance accessibility and usability for a broader audience?

04

How does the proposed system address these gaps and stand out from existing solutions?

# OBJECTIVES

- **Develop a Dynamic Budget System**

Create an intelligent Travel Budget Allocation System that adapts to real-time data and user preferences, offering personalized and accurate budget recommendations.

- **Implement Predictive Algorithms**

Integrate predictive models to estimate travel costs (accommodations, transport, meals) using historical data and real-time inputs for accurate forecasts.

- **Design a User-Centric Interface**

Build an intuitive interface allowing travelers to customize budget preferences and receive tailored recommendations based on their travel style and goals.

- **Integrate APIs & Validate System**

Integrate external APIs for real-time cost data and conduct rigorous testing to ensure system accuracy, reliability, and adaptability to changing market conditions and user needs.

# SMART BUDGET ALLOCATION & ADAPTIVE PLANNING

"Our budget system tailors recommendations based on traveler type and package, offering options like affordable hostels for solo travelers and resorts for families, ensuring a personalized, budget-friendly experience."

**Customizable Packages:** Basic, Moderate, and Premium options tailored to different traveler types, offering varying levels of amenities and services.

**Diverse Travel Companions:** Plans customized for Solo, Couple, Family, and Friends, based on group size and preferences.

**Dynamic Cost Predictions:** Predicts breakdowns for Transport, Stay, Activities, Food, and Accommodation, adjusting based on real-time pricing and user preferences.

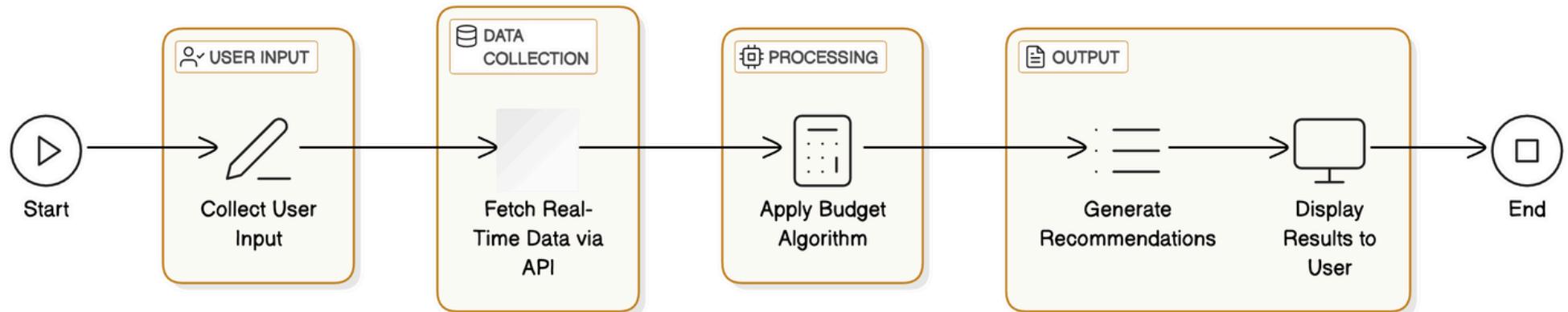


# METHODOLOGY

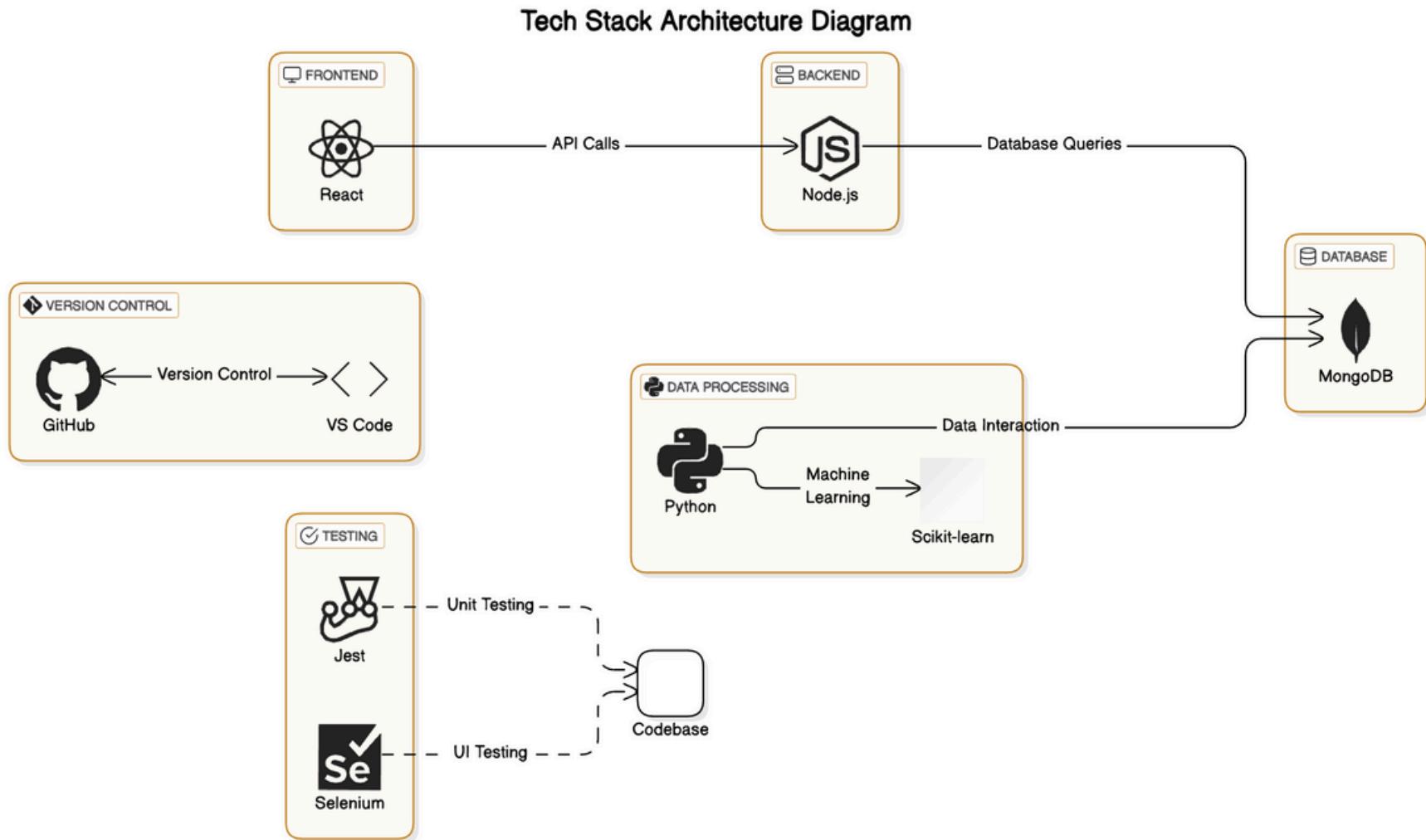
01. SYSTEM DIAGRAM
02. TOOLS AND TECHNOLOGIES
03. REQUIREMENTS
04. GITHUB COMMITS



# BUDGET PREDICTION SYSTEM DIAGRAM



# TOOLS & TECHNOLOGIES



# REQUIREMENTS

## FUNCTIONAL

- Real-Time Data Integration
- Accurate Predictions
- Customizable Suggestions
- Notification System
- Collaboration

## NON-FUNCTIONAL

- Reliability of Predictions
- Fast Response Times
- User-Friendly Interface
- Maintainability
- Localization

## BEST PRACTICES

- Code quality
- Web security
- Modal quality
- Data Privacy



# GITHUB COMMITS

Commits on Mar 18, 2025			
<b>generated file set 02</b>		9109f66	 
 thuwakaranR committed 3 weeks ago			
<b>generated file set 01</b>		50d7f34	 
 thuwakaranR committed 3 weeks ago			
<b>file updatios and modal creatiion starting with metrices</b>		9e4db10	 
 thuwakaranR committed 3 weeks ago			
<b>working with user demographic data set</b>		32c2f49	 
 thuwakaranR committed 3 weeks ago			
<b>similarity metrices implementations</b>		48c3238	 
 thuwakaranR committed 3 weeks ago			
Commits on Mar 16, 2025			
<b>generated csv file v2</b>		229f6f2	 
 thuwakaranR committed 3 weeks ago			
<b>generated csv file v1</b>		f8216f3	 
 thuwakaranR committed 3 weeks ago			
<b>hybrid and other implementation 02</b>		f28933f	 
 thuwakaranR committed 3 weeks ago			
<b>hybrid and other implementation 01</b>		61f7547	 
 thuwakaranR committed 3 weeks ago			



# COMPLETION OF THE PROJECT



# COMPLETION OF THE PROJECT

50%

# DATA COLLECTION

2617	LOC_76	PACK_76Z3-2	Premium	Sunrise Visit, Attend Thaipoosam Festival with Private Viewing, Explore Temple with Private Guide	1	Luxury Resort	All-Inclusive, Private Transport	20,500	Exclusive Sunrise Visit, Festival Experience	3.9	70	Couple
2618	LOC_76	PACK_76Z2-3	Moderate	Visit Temple, Attend Special Worship, Visit the Garden and Pond	1	Boutique Hotel	Full Board, Shared Transport	15,300	Worship, Spiritual Serenity	3.6	55	Couple
2619	LOC_76	PACK_76Z3-3	Premium	Private Cultural Experience: Attend Special Worship, Explore Temple Architecture, Visit the Pond	1	Luxury Resort	All-Inclusive, Private Transport	21,500	Tailored Experience, Private Guide	2.7	80	Couple
2620	LOC_76	PACK_76Z2-2	Moderate	Attend the Annual Festival, Explore Temple Parades with Music & Dance	1	Boutique Hotel	Full Board, Shared Transport	15,000	Festival Atmosphere, Parades	3.9	50	Couple
2621	LOC_76	PACK_76Z3-4	Premium	Full Day Experience: Nallur Kandaswamy Temple, Lord Muruga's Worship, Special Blessings	1	Luxury Boutique Stay	All-Inclusive, VIP Transport	22,500	Exclusive Experience, VIP Worship	4.9	90	Couple
2622	LOC_1	PACK_1Z1-1	Basic	Surfing (Beginner), Beach Walks	1	Budget Guesthouse	Breakfast Only, Local Transport	34,500	Public Beach Access, Local Culture	2.8	95	Family
2623	LOC_1	PACK_1Z1-2	Basic	Lagoon Safari, Elephant Rock Visit	1	Homestay	Breakfast & Dinner, Shared Transport	38,500	Wildlife Spotting, Scenic Views	3.2	80	Family
2624	LOC_1	PACK_1Z2-1	Moderate	Surfing (Intermediate), Jeep Safari in Kumana	2	Beachfront Hotel	Half Board, Private Transport	51,000	Beachfront View, Guided Tours	4.1	120	Family
2625	LOC_1	PACK_1Z2-2	Moderate	Snorkeling, Yoga, Local Cuisine Tour	2	Boutique Villa	Full Board, Shared Transport	54,000	Traditional Food, Cultural Immersion	3.7	110	Family
2626	LOC_1	PACK_1Z3-1	Premium	Private Surfing Lessons, Private Boat Tours	2	Luxury Resort	Full Board, Private Chauffeur	81,000	Private Beach Access, Luxury Spa	4.6	150	Family

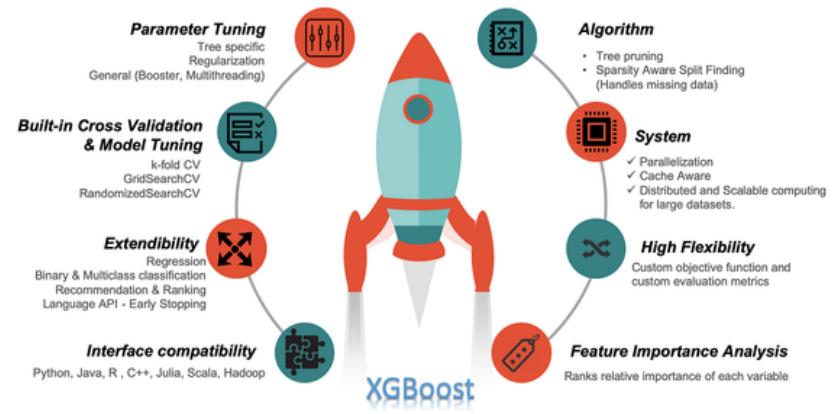


	Location_ID	Package_ID	Package_Type	Activities	Duration (Days)	Accommodation	Food & Transport	Budget (LKR)	Location_Features	Avg_Rating	Review_Count	Travel_Companion
1	LOC_1	PACK_1Z1-1	Basic	Surfing (Beginner), Beach Walks	1	Budget Guesthouse	Breakfast Only, Local Transport	23,500	Public Beach Access, Local Culture	2.8	95	Solo
2	LOC_1	PACK_1Z1-2	Basic	Lagoon Safari, Elephant Rock Visit	1	Homestay	Breakfast & Dinner, Shared Transport	27,500	Wildlife Spotting, Scenic Views	3.2	80	Solo
3	LOC_1	PACK_1Z2-1	Moderate	Surfing (Intermediate), Jeep Safari in Kumana	2	Beachfront Hotel	Half Board, Private Transport	40,000	Beachfront View, Guided Tours	4.1	120	Solo
4	LOC_1	PACK_1Z2-2	Moderate	Snorkeling, Yoga, Local Cuisine Tour	2	Boutique Villa	Full Board, Shared Transport	43,000	Traditional Food, Cultural Immersion	3.7	110	Solo
5	LOC_1	PACK_1Z3-1	Premium	Private Surfing Lessons, Private Boat Tours	2	Luxury Resort	Full Board, Private Chauffeur	70,000	Private Beach Access, Luxury Spa	4.6	150	Solo
6	LOC_1	PACK_1Z3-2	Premium	Helicopter Tour, Exclusive Beachfront Dining	3	5-Star Resort	All-Inclusive, Luxury Transport	90,000	VIP Beachfront, Exclusive Experience	4.3	130	Solo
7	LOC_1	PACK_1Z3-3	Premium	Yala Safari Camping, Traditional 'Oruwa' Rafting	2	Luxury Safari Camp	Full Board, Private Guide	100,000	Unique Wildlife Encounters	4.8	140	Solo
8	LOC_1	PACK_1Z1-3	Basic	Beach Volleyball, Turtle Watching	1	Hostel Dorm	Breakfast Only, Shared Transport	22,500	Public Beach Access, Marine Life	2.5	70	Solo
9	LOC_1	PACK_1Z1-4	Basic	Mangrove Kayaking, Fishing Experience	1	Eco Lodge	Breakfast & Dinner, Local Transport	25,000	Scenic Views, Local Culture	3.1	65	Solo
10	LOC_1	PACK_1Z2-3	Moderate	Stand-up Paddleboarding, Cultural Dance Show	2	Eco Boutique Hotel	Full Board, Private Transport	41,500	Traditional Food, Cultural Immersion	4.4	100	Solo
11	LOC_1	PACK_1Z2-4	Moderate	Hiking, Tea Plantation Visit	2	Hillside Retreat	Half Board, Shared Transport	39,500	Scenic Views, Nature Trails	3.8	105	Solo
12	LOC_1	PACK_1Z2-5	Moderate	Sunset Cruise, Local Handicraft Workshop	2	Riverside Bungalow	Full Board, Local Transport	42,000	Traditional Crafts, River Views	4	90	Solo

# MODEL SELECTION

## -XGBoost Regressor with RandomizedSearchCV-

- Explored XGBoost Regressor to predict the budget for a given travel input setup using customized features.
- Used RandomizedSearchCV for tuning hyperparameters to improve prediction accuracy and model generalization.



- Fine-tuned Performance
- Better Accuracy
- Flexible and Powerful
- Efficient and Fast

# BUDGET PREDICTION – CODE EXECUTION FLOW

```
[ ] df = pd.concat([df, df_activities], axis=1)
df.drop(columns=['Activities'], inplace=True) # Drop original activities column

# Add Travel_Companion column and encode it
travel_companion_map = {'Solo': 0, 'Couple': 1, 'Family': 2, 'Friends': 3}
df['Travel_Companion'] = df['Travel_Companion'].map(travel_companion_map)

# Define Rating Ranges (0-1, 1-2, 2-3, 3-4, 4-5)
def categorize_rating(rating):
    return f'{int(rating)}-{int(rating) + 1}'

df['Rating_Range'] = df['Avg_Rating'].apply(categorize_rating)

[ ] # Select features and target variable
X = df.drop(columns=['Package_ID', 'Accommodation', 'Food & Transport', 'Budget (LKR)', 'Rating_Range'])
y = df['Budget (LKR)']

# Split dataset into training (80%) and testing (20%)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

[ ] # Initialize XGBoost regressor model
xgb_model = XGBRegressor(random_state=42)

# Train the model
xgb_model.fit(X_train, y_train)
```

```
if selected_packages:
    final_results = pd.DataFrame(selected_packages)
    final_results = final_results[['Location', 'Package_ID', 'Package_Type', 'Days', 'Food & Transport', 'Avg_Rating', 'Activities']]
]

# Format the predicted budget to two decimal places
final_results['Predicted_Budget'] = final_results['P']

total_duration = final_results['Days'].sum()
total_budget = round(total_budget, 2)
possible_combinations.append((final_results, total_d

return possible_combinations
```

```
[ ] # Make predictions
y_pred = xgb_model.predict(X_test)

# Evaluate the model
mae = mean_absolute_error(y_test, y_pred)
rmse = mean_squared_error(y_test, y_pred) ** 0.5
r2 = r2_score(y_test, y_pred)

# Display evaluation results
print(f"✅ XGBoost Model Evaluation:\nMAE: {mae}\nRMSE: {rmse}\nR2 Score: {r2}")

→ ✅ XGBoost Model Evaluation:
MAE: 2769.10 LKR
RMSE: 5242.96 LKR
```

```
XGBOOST
dict(X_test)

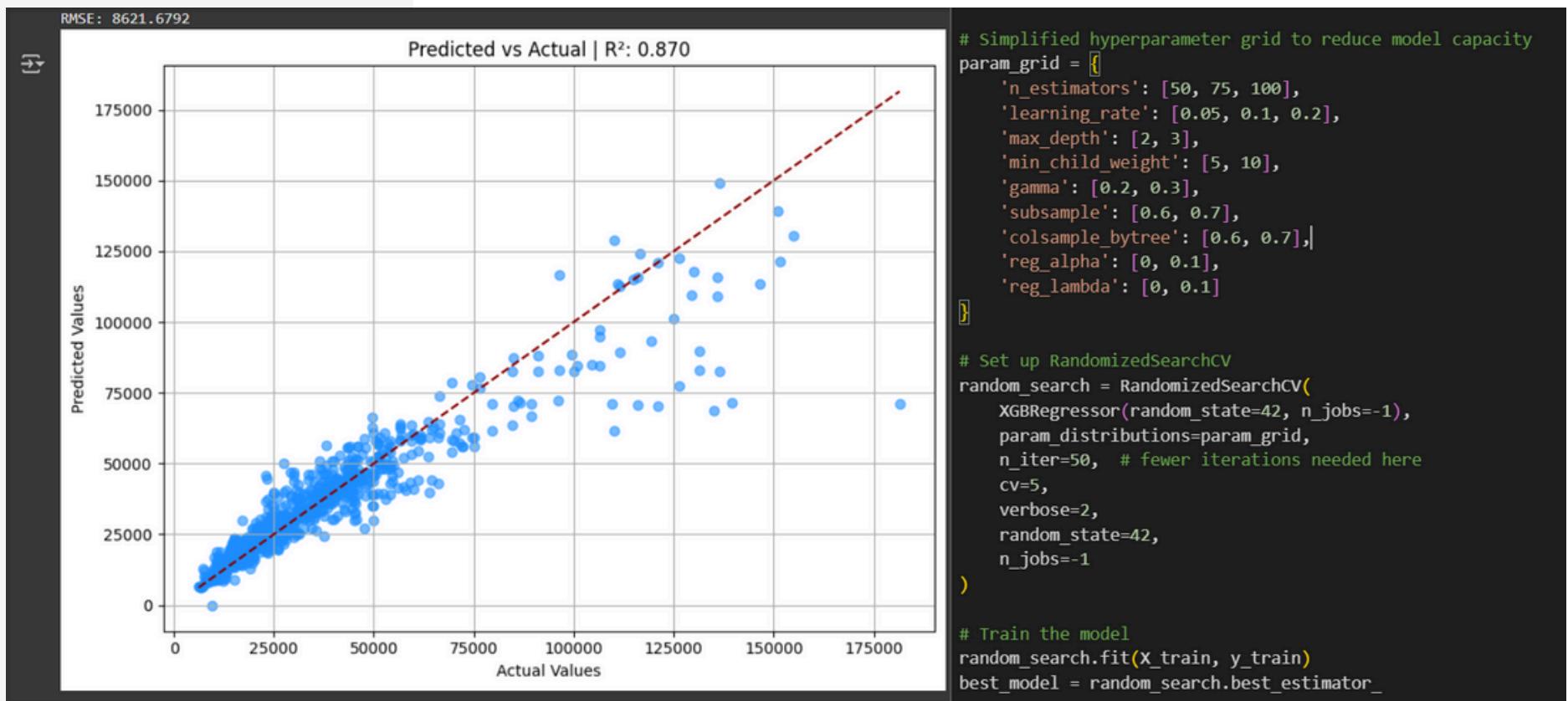
# evaluate
r2_tuned = r2_score(y_test, y_pred_tuned)
rmse = np.sqrt(mean_squared_error(y_test, y_pred_tuned)) # fixed RMSE calculation

# Output
print("\nBest Parameters Found:")
print(random_search.best_params_)
print(f"R² Score: {r2_tuned:.4f}")
print(f"RMSE: {rmse:.4f}")

# Plotting predicted vs actual
plt.figure(figsize=(8, 6))
plt.scatter(y_test, y_pred_tuned, alpha=0.6, color='dodgerblue')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], color='darkred', linestyle='--')
plt.title(f"Predicted vs Actual | R²: {r2_tuned:.3f}")
plt.xlabel("Actual Values")
plt.ylabel("Predicted Values")
plt.grid(True)
plt.tight_layout()
plt.show()
```

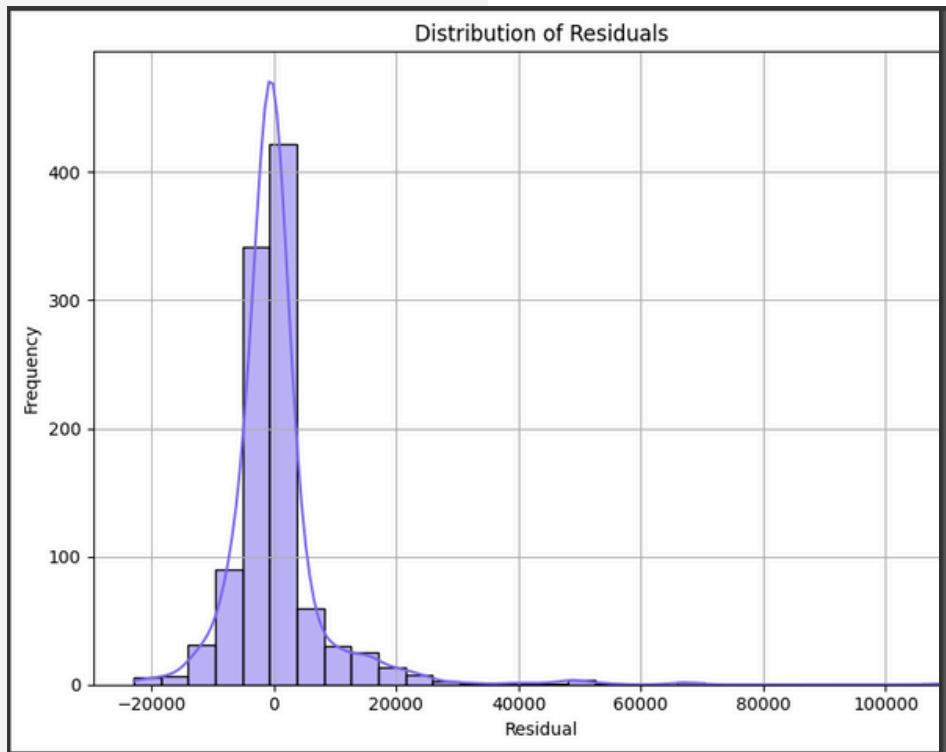
# ACCURACY & ANALYSIS

- Plotting



# ACCURACY & ANALYSIS

- Plot Histogram



```
import seaborn as sns
import matplotlib.pyplot as plt

# Calculate residuals
residuals = y_test - y_pred_tuned

# Plot histogram of residuals
plt.figure(figsize=(8, 6))
sns.histplot(residuals, kde=True, bins=30,
             plt.title("Distribution of Residuals")
             plt.xlabel("Residual")
             plt.ylabel("Frequency")
             plt.grid(True)
             plt.tight_layout()
             plt.show()
```

# BUDGET PREDICTION & ITERATIVE RESULTS

"HERE'S HOW OUR BUDGET SYSTEM FITS IN. IT PULLS LIVE COST DATA, APPLIES PREDICTION LOGIC, AND RETURNS AN OPTIMIZED, ADAPTIVE BUDGET PLAN BASED ON USER INPUTS AND PREFERENCES."

```
locations = ["LOC_11", "LOC_42", "LOC_6"]
package = "Moderate"
total_days = 8
rating_range = "4-5"
travel_companion = "solo"

predicted_packages = predict_budget_multiple_options(locations, package, total_days, rating_range, travel_companion, df, best_model, le_location, le_package)

# Display the results
for i, (packages, total_duration, total_budget) in enumerate(predicted_packages, 1):
    print("+" * 72)
    print(f"{'*' * 72} Trip Plan {str(i)} {'*' * 72}") # Center-aligned title
    print("+" * 72)

    # Display the package details
    print(packages)
    |
    # Display travel companion
    print("-" * 72)
    print(f"💡 Travel Companion: {travel_companion}")

    print("-" * 72)
    print(f"⌚ Total Duration: {total_duration} days")
    print(f"💰 Total Budget: {total_budget:.2f} LKR")
    print("-" * 72, "\n")
    print("\n")
```

Trip Plan 1					
Location	Package_ID	Package_Type	Days	Accommodation	
697	LOC_42	PACK_42Z2-4	Moderate	2.0	Seaside Guesthouse
696	LOC_42	PACK_42Z2-1	Moderate	1.0	Boutique Villa
101	LOC_6	PACK_6Z2-3	Moderate	2.0	4-Star Resort

Food & Transport	Avg_Rating
697 Full Board, Shared Transport	4.0
696 Half Board, Shared Transport	4.1
101 Full Board, Private Transport	4.3

Activities	Predicted_Budget
697 [Morning Safari, Nature Walk, Spot Flamingos]	25405.94
696 [Birdwatching, Explore Flora & Fauna, Lagoon T...	20413.94
101 [Boat Ride to Trincomalee, Whale Watching (Sea...	51253.43

💡 Travel Companion: Solo

⌚ Total Duration: 5.0 days

💰 Total Budget: 97073.31 LKR

Trip Plan 2					
Location	Package_ID	Package_Type	Days	Accommodation	
200	LOC_11	PACK_11Z2-8	Moderate	3.0	Beachfront Resort
697	LOC_42	PACK_42Z2-4	Moderate	2.0	Seaside Guesthouse
103	LOC_6	PACK_6Z2-5	Moderate	2.0	Beachfront Hotel

Food & Transport	Avg_Rating
200 Full Board, Private Transport	4.0
697 Full Board, Shared Transport	4.0
103 Full Board, Private Transport	4.1

Activities	Predicted_Budget
200 [Relax on Kalkudah Beach, Watersports at Pasik...	26724.68
697 [Morning Safari, Nature Walk, Spot Flamingos]	25405.94
103 [Diving, Underwater Photography]	43833.84

💡 Travel Companion: Solo

⌚ Total Duration: 7.0 days

💰 Total Budget: 95964.38 LKR

Trip Plan 3					
Location	Package_ID	Package_Type	Days	Accommodation	
200	LOC_11	PACK_11Z2-8	Moderate	3.0	Beachfront Resort
696	LOC_42	PACK_42Z2-1	Moderate	1.0	Boutique Villa
103	LOC_6	PACK_6Z2-5	Moderate	2.0	Beachfront Hotel

Food & Transport	Avg_Rating
200 Full Board, Private Transport	4.0
696 Half Board, Shared Transport	4.1
103 Full Board, Private Transport	4.1

Activities	Predicted_Budget
200 [Relax on Kalkudah Beach, Watersports at Pasik...	23317.40
696 [Birdwatching, Explore Flora & Fauna, Lagoon T...	16623.75
103 [Diving, Underwater Photography]	28720.92

💡 Travel Companion: Solo

⌚ Total Duration: 6.0 days

💰 Total Budget: 68662.07 LKR



# COMPLETION OF THE PROJECT

90%

# KEY FUNCTIONALITIES

- **AI-Powered Predictions** – Generate multiple trip plan options based on user-selected preferences (locations, days, package, ratings, companion).
- **Iterative Plan Options** – Provide alternative detailed itineraries with budget, activities, and duration breakdowns.
- **User Plan Selection** – Enable users to review, choose, and confirm one preferred plan.
- **Plan Management** – Save confirmed plans in the system for later retrieval, with options to view or delete.
- **Chatbot Assistance** – AI chatbot supports users in customizing or refining plans when system-generated ones don't meet needs.
- **Comprehensive API** – FastAPI endpoints for prediction, confirmation, retrieval, deletion, and chatbot integration.
- **PDF Exporting** – Download finalized plans as professional PDF reports for easy sharing.

# FAST API - IMPLEMENTATION

```
# Predict trip plan
@app.post("/predict", summary="Predict trip budget options")
async def predict_trip(req: TripRequest):
    try:
        results = predict_budget_multiple_options(
            locations=req.locations,
            package=req.package,
            total_days=req.total_days,
            rating_range=req.rating_range,
            travel_companion=req.travel_companion,
        )

        data_to_save = req.dict()
        data_to_save["predictions"] = [
            {
                "plan": plan.get("plan", []),
                "total_days": plan.get("total_days"),
                "total_budget": plan.get("total_budget"),
                "travel_companion": plan.get("travel_companion"),
            }
            for plan in results
        ]
        predictions_collection.insert_one(data_to_save)

        return {"combinations": data_to_save["predictions"]}

    except Exception as e:
        print("ERROR in /predict:", traceback.format_exc())
        raise HTTPException(status_code=500, detail=f"Prediction failed: {str(e)}")

# Confirm selected trip plan
@app.post("/confirm", summary="Confirm a selected trip plan")
async def confirm_plan(req: ConfirmPlanRequest):
    try:
        doc = {
            "plan_number": req.plan_number,
            "package_ids": req.package_ids,
            "confirmed_at": req.confirmed_at.isoformat(),
            "user_id": req.user_id,
            "full_plan": req.full_plan,
        }

        result = confirmed_plans_collection.insert_one(doc)

        return {
            "message": "Plan confirmed successfully",
            "id": str(result.inserted_id)
        }

    except Exception as e:
        print("ERROR in /confirm:", traceback.format_exc())
        raise HTTPException(status_code=500, detail=f"Failed to confirm plan: {str(e)}")
```

PREDICT TRIP PLAN

CONFIRM PLAN

```
# ChatBot using Cohere
You, last month | 1 author (You)
class ChatRequest(BaseModel):
    message: str

You, last month | 1 author (You)
class ChatResponse(BaseModel):
    reply: str

@app.post("/chatbot", response_model=ChatResponse, summary="Chatbot response using Cohere")
async def chatbot_endpoint(chat_request: ChatRequest):
    user_message = chat_request.message

    try:
        response = co.chat(
            message=user_message,
            model="command-r-plus",
            temperature=0.7,
        )

        return ChatResponse(reply=response.text.strip())

    except Exception as e:
        print("Cohere Error:", traceback.format_exc())
        raise HTTPException(status_code=500, detail=f"Chatbot error: {str(e)}")
```

CHAT BOT  
IMPLEMENTATION

```
# Get all confirmed plans
@app.get("/confirmed-plans", summary="Get all confirmed plans")
async def get_confirmed_plans(user_id: str = None):
    try:
        query = {}
        if user_id:
            query["user_id"] = user_id

        plans = list(confirmed_plans_collection.find(query))

        for p in plans:
            p["_id"] = str(p["_id"])
            if "confirmed_at" in p and hasattr(p["confirmed_at"], "isoformat"):
                p["confirmed_at"] = p["confirmed_at"].isoformat()

        return {"confirmed_plans": plans}

    except Exception as e:
        print("ERROR in /confirmed-plans:", traceback.format_exc())
        raise HTTPException(status_code=500, detail=f"Failed to fetch confirmed plans: {str(e)}")
```

GET ALL PLANS

# POSTMAN API TEST

## BUDGET PREDICTION POST CALL

The screenshot shows a Postman test result for a 'BUDGET PREDICTION POST CALL'. The request body contains JSON data for travel parameters. The response status is 200 OK, with a duration of 4.77 s and a size of 4.35 KB. The response body is a JSON object containing a travel plan with details like location, package ID, days, accommodation, food & transport, average rating, activities, and predicted budget.

```
1 {  
2   "locations": ["LOC_11", "LOC_42", "LOC_6"],  
3   "package": "Moderate",  
4   "total_days": 10,  
5   "rating_range": "3.0-5.0",  
6   "travel_companion": "Family"  
7 }  
  
200 OK • 4.77 s • 4.35 KB • 🔍 Save Response ⚙️  
  
Body Cookies Headers (4) Test Results ⚙️  
{ } JSON ▾ ▶ Preview ⚙️ Visualize ⚙️  
  
4   "plan": [  
5     {  
6       "Location": "LOC_11",  
7       "Package_ID": "PACK_11Z2-8",  
8       "Package_Type": "Moderate",  
9       "Days": 3.0,  
10      "Accommodation": "Beachfront Resort",  
11      "Food & Transport": "Full Board, Private Transport",  
12      "Avg_Rating": 4.0,  
13      "Activities": [  
14        "Relax on Kalkudah Beach",  
15        "Watersports at Pasikuda Bay"  
16      ],  
17      "Predicted_Budget": "34367.29"  
18 ]  
19  
20
```

## CONFIRMED PLAN POST CALL

The screenshot shows a Postman test result for a 'CONFIRMED PLAN POST CALL'. The request body contains JSON data for confirming a travel plan. The response status is 200 OK, with a duration of 100 ms and a size of 198 B. The response body is a JSON object with a confirmation message and a unique ID.

```
1 {  
2   "user_id": "user123",  
3   "plan_number": 0,  
4   "package_ids": ["PACK_11Z2-1", "PACK_42Z2-3", "PACK_42Z2-1"],  
5   "confirmed_at": "2025-08-02T12:00:00Z"  
6 }  
7  
  
200 OK • 100 ms • 198 B • 🔍 Save Response ⚙️  
  
Body Cookies Headers (4) Test Results ⚙️  
{ } JSON ▾ ▶ Preview ⚙️ Visualize ⚙️  
  
1 {  
2   "message": "Plan confirmed successfully",  
3   "id": "68c658a586e89b58269db795"  
4 }  
5  
6
```

# POSTMAN API TEST

## BUDGET PREDICTION GET CALL

Body Cookies Headers (4) Test Results ⏱

200 OK • 5 ms • 204 B • Save Response ⏮

{ } JSON ▾ ▶ Preview ⏷ Visualize ▾

```
1 {  
2   "message": "Trip Budget Prediction API is up and running with Cohere chatbot!"  
3 }
```

## CHATBOT POST CALL

Request Body

```
1 {  
2   "message": "Can you help me plan my trip?"  
3 }  
4
```

Body Cookies Headers (4) Test Results ⏱

200 OK • 7.80 s • 1.24 KB • Save Response ⏮

{ } JSON ▾ ▶ Preview ⏷ Visualize ▾

```
1 {  
2   "reply": "Sure! I'd be happy to help you plan your trip. Please provide me with some details about your travel preferences, budget, and any specific destinations or activities you are interested in. Here are some questions to get started:\n\n1. What type of trip are you planning? (e.g. beach vacation, city exploration, outdoor adventure, cultural trip)\n2. When would you like to travel? (e.g. specific dates or flexibility)\n3. How long do you plan to travel for?\n4. What is your approximate budget for the trip?\n5. Are there any specific destinations or regions you are interested in visiting?\n6. What type of accommodations do you prefer? (e.g. hotels, hostels, Airbnbs, camping)\n7. What activities or experiences are you interested in? (e.g. sightseeing, hiking, food tours, museums)\n8. Do you have any specific travel constraints or preferences (e.g. dietary restrictions, mobility concerns, fear of flying)?\n\nBased on your responses, I can provide tailored recommendations and create a customized itinerary for your trip, including transportation, accommodations, and activities. Let's work together to plan an amazing journey!"  
3 }
```

Schema ⏷

# WORKING BUDGET PREDICTION

## USER INTERACTION PAGE

### Plan Your Ideal Trip

Fill in your trip preferences to receive optimized budget predictions tailored just for you.

**Locations**  
LOC\_11, LOC\_42, LOC\_67  
Use location IDs, separated by commas.

**Package Type**  
Moderate

**Total Days**  
5

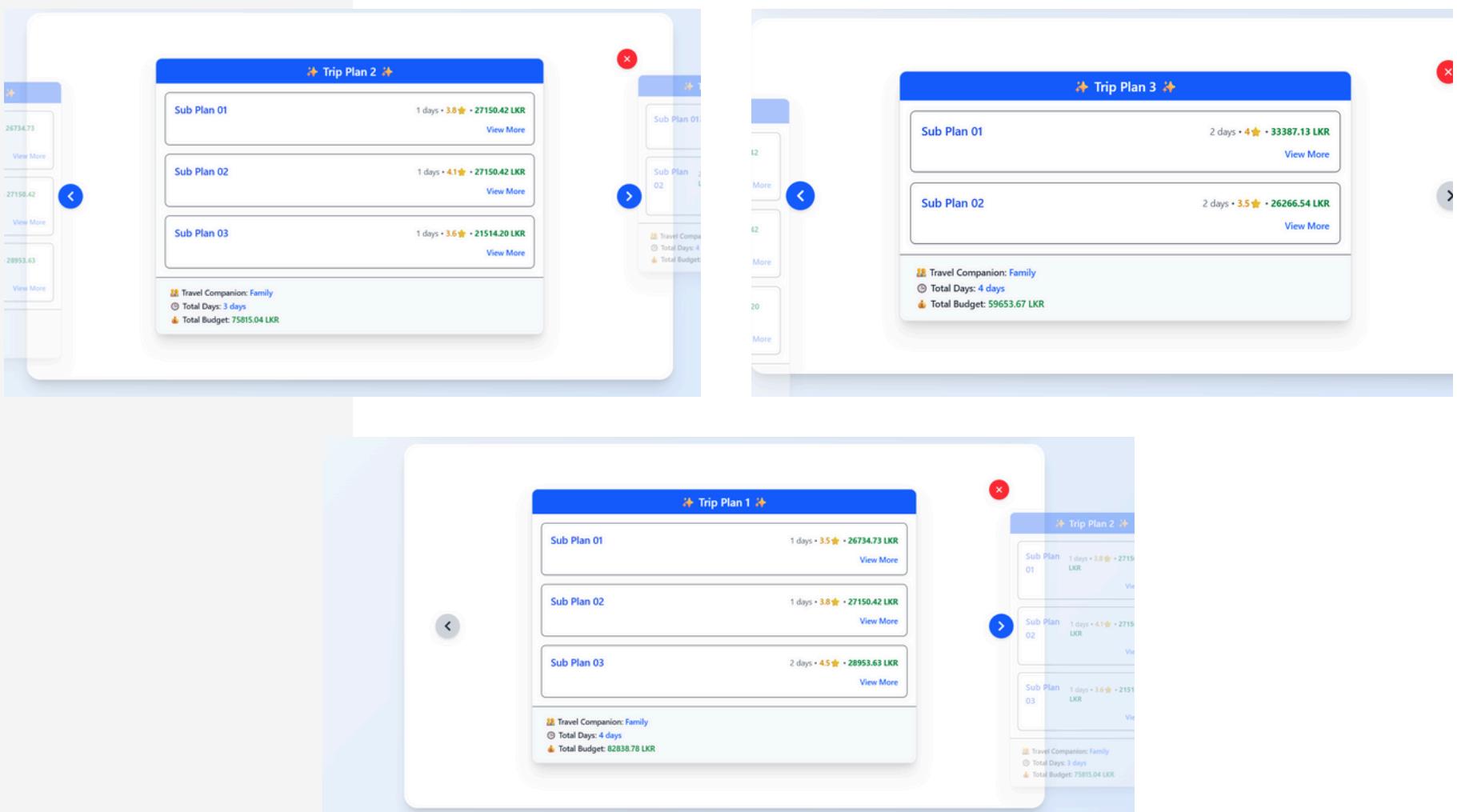
**Rating Range**  
Min: 3.0 Max: 5.0

**Travel Companion**  
Family

**Buttons**  
Fill Sample Reset Predict Budget

# WORKING BUDGET PREDICTION

## ITERATIVE BUDGET PLANS



# WORKING BUDGET PREDICTION

## CONFIRMED PLAN & CHATBOT VIEW

The image displays a user interface for a travel budget prediction system. On the left, a sidebar shows budget components: 5734.73, 7150.42, and 8953.63, each with a 'View More' button. A blue circular arrow icon is positioned between the first two items. In the center, a large blue-bordered box titled 'Trip Plan 2' lists three sub-plans:

- Sub Plan 01**: 1 days • 3.8★ • 27150.42 LKR (with 'View More' link)
- Sub Plan 02**: 1 days • 4.1★ • 27150.42 LKR (with 'View More' link)
- Sub Plan 03**: 1 days • 3.6★ • 21514.20 LKR (with 'View More' link)

Below these, travel details are listed: Travel Companion: Family, Total Days: 3 days, and Total Budget: 75815.04 LKR. At the bottom are 'Confirm Plan' and 'Cancel' buttons.

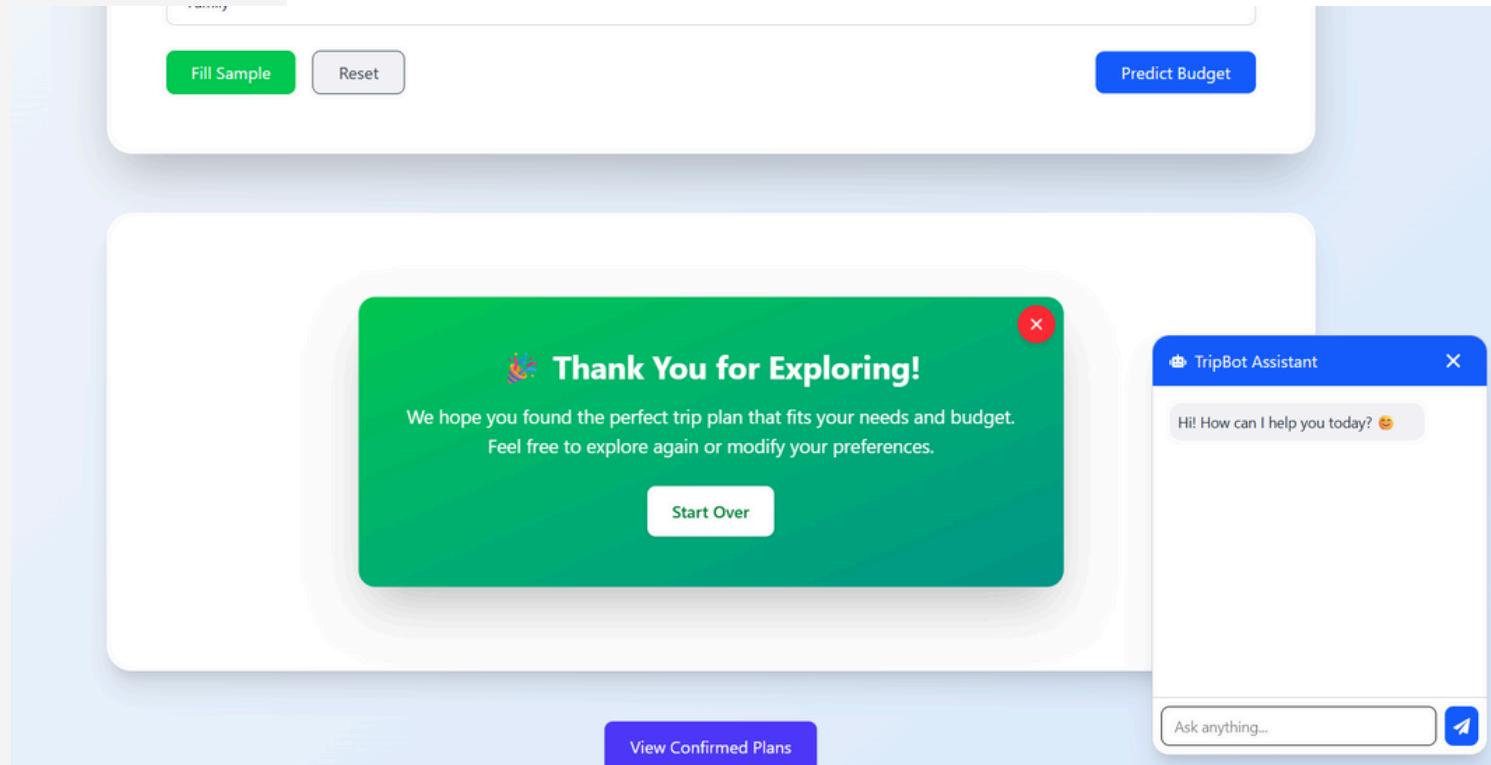
To the right, a vertical stack of cards shows 'Sub 01' and 'Sub 02'. A red 'X' icon is at the top of the stack. Below the stack is a small card with travel icons.

At the bottom center, a green box displays a success message: "Thank You for Exploring! You have successfully selected Trip Plan 2." A purple 'View Confirmed Plans' button is below it.

On the far right, a 'TripBot Assistant' chatbot window is open. It shows a message: "Thank you for selecting Trip Plan 2! Here's how the system works: Based on your preferences and predicted budget, we've created the most optimized travel combo for you. 🎉 Are you satisfied with this plan? Feel free to share any thoughts or feedback!" An input field says "Ask anything..." and has a send icon.

# WORKING BUDGET PREDICTION

## CANCELLED PLAN & CHATBOT VIEW



# WORKING BUDGET PREDICTION

## CONFIRMED PLANS SCREEN



### Confirmed Trip Plans

#### Trip Plan #2

Confirmed At: 8/25/2025, 10:39:38 PM

Packages Included: PACK\_11Z2-3, PACK\_42Z2-1, PACK\_67Z2-1

[Download PDF](#)[View Details](#)

#### Trip Plan #1

Confirmed At: 8/2/2025, 5:30:00 PM

Packages Included: PACK\_11Z2-1, PACK\_42Z2-3, PACK\_42Z2-1

[Download PDF](#)[View Details](#)

#### Trip Plan #2

Confirmed At: 9/14/2025, 12:31:03 PM

Packages Included: PACK\_42Z2-6, PACK\_42Z2-1, PACK\_67Z2-1

[Download PDF](#)[View Details](#)

## VIEW PLANS SCREEN

#### Package Summary

Package IDs: PACK\_42Z2-6, PACK\_42Z2-1, PACK\_67Z2-1

#### Full Plan Details

Sub Plan 01 | PACK\_42Z2-6 (Moderate)

Days: 1

Accommodation: Riverside Lodge

Food & Transport: Breakfast Only, Shared Transport

Average Rating: 3.8

Predicted Budget: 27150.42

##### Activities:

- Half-Day Safari
- Lagoon Exploration
- Visit Flamingo Colonies

Sub Plan 02 | PACK\_42Z2-1 (Moderate)

Days: 1

Accommodation: Boutique Villa

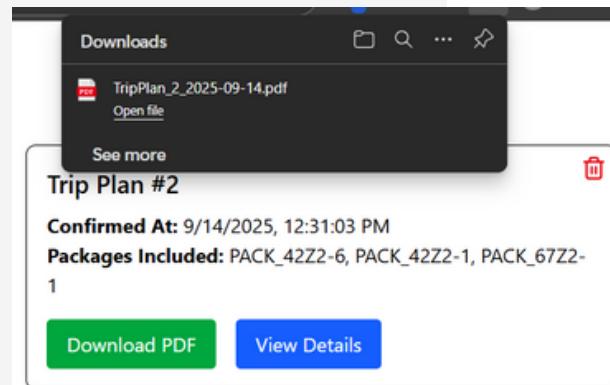
Food & Transport: Half Board, Shared Transport

Average Rating: 4.1

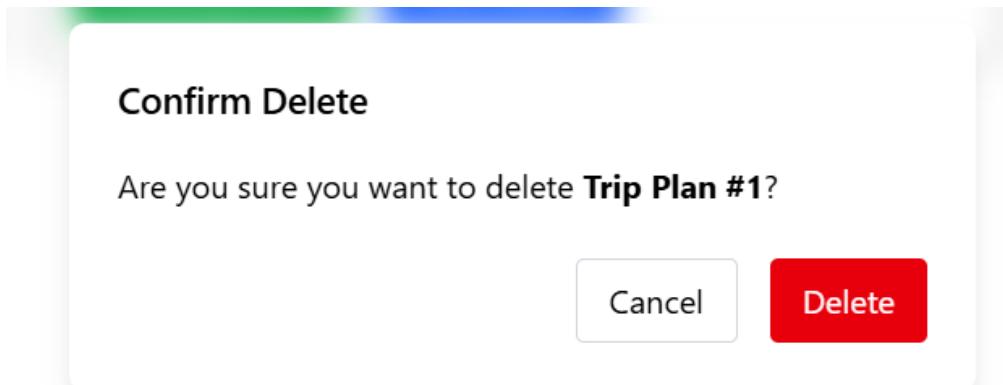
Predicted Budget: 27150.42

# WORKING BUDGET PREDICTION

## PDF VIEW OF SELECTED PLAN



## DELETE SCREEN VIEW



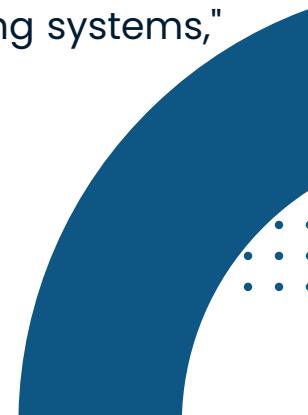
### Trip Plan #2

Confirmed At: 9/14/2025, 12:31:03 PM

Package IDs: PACK\_42Z2-6, PACK\_42Z2-1, PACK\_67Z2-1

**Full Plan Details:**

Plan:	Sub Plan 01	Package:	PACK_42Z2-6 (Moderate)
Days:	1		
Accommodation:	Riverside Lodge		
Food & Transport:	Breakfast Only, Shared Transport		
Average Rating:	3.8		
Predicted Budget:	LKR 27,150.42		
Activities:	<ul style="list-style-type: none"><li>• Half-Day Safari</li><li>• Lagoon Exploration</li><li>• Visit Flamingo Colonies</li></ul>		



# REFERENCES



- [1] J. Anderson, M. Smith, and L. Roberts, "Limitations in static budgeting models: Examining traditional travel budgeting tools," *Journal of Travel Management*, vol. 34, no. 2, pp. 45–62, 2020.
- [2] H. Tan, C. Lee, and S. Tan, "Role of real-time data in budget predictions: Integrating APIs for travel budgeting systems," *International Journal of Data Analytics*, vol. 19, no. 1, pp. 30–42, 2021.
- [3] X. Zhang, L. Wang, and Q. Chen, "Importance of user-centric designs in travel budgeting tools," *Journal of User Experience Design*, vol. 12, no. 4, pp. 58–75, 2022.
- [4] J. Smith and K. Thomas, "Machine learning for travel budgeting: Enhancing cost predictions with predictive algorithms," *Journal of Artificial Intelligence in Tourism*, vol. 16, no. 3, pp. 215–228, 2021.
- [5] M. Green and D. Turner, "Real-time data integration: A new frontier in travel budgeting systems," *Journal of Travel Technology*, vol. 21, no. 1, pp. 50–65, 2020.



THANK YOU

