**Weekly Report 11 & 12**

Date:

## Implementation Details

**Code :**

GUI :

# Recommendation functions

def get\_user\_based\_recommendations(user\_id, num\_recommendations=5):

try:

# Create user-destination matrix

user\_dest\_matrix = userhistory\_df.pivot\_table(

index='UserID',

columns='DestinationID',

values='ExperienceRating',

fill\_value=0

)

# Calculate cosine similarity between users

user\_similarity = cosine\_similarity(user\_dest\_matrix)

user\_similarity\_df = pd.DataFrame(

user\_similarity,

index=user\_dest\_matrix.index,

columns=user\_dest\_matrix.index

)

# Get similar users (excluding the user themselves)

similar\_users = user\_similarity\_df[user\_id].sort\_values(ascending=False)[1:6].index

# Get destinations rated highly by similar users

similar\_users\_ratings = user\_dest\_matrix.loc[similar\_users]

avg\_ratings = similar\_users\_ratings.mean(axis=0)

# Filter out destinations already visited by the user

visited\_destinations = userhistory\_df[userhistory\_df['UserID'] == user\_id]['DestinationID'].unique()

recommendations = avg\_ratings[~avg\_ratings.index.isin(visited\_destinations)]

recommendations = recommendations.sort\_values(ascending=False).head(num\_recommendations)

return recommendations.index.tolist()

except Exception as e:

st.error(f"Error in recommendation generation: {str(e)}")

return []

def get\_popular\_recommendations(num\_recommendations=5):

return destinations\_df.sort\_values('Popularity', ascending=False)['DestinationID'].head(num\_recommendations).tolist()

def get\_content\_based\_recommendations(destination\_id, num\_recommendations=5):

try:

# Vectorize destination features

vectorizer = CountVectorizer()

features = destinations\_df['Type'] + ' ' + destinations\_df['State'] + ' ' + destinations\_df['BestTimeToVisit']

feature\_matrix = vectorizer.fit\_transform(features)

# Calculate cosine similarity between destinations

similarity\_matrix = cosine\_similarity(feature\_matrix)

# Get similar destinations

destination\_idx = destinations\_df[destinations\_df['DestinationID'] == destination\_id].index[0]

similar\_destinations = list(enumerate(similarity\_matrix[destination\_idx]))

similar\_destinations = sorted(similar\_destinations, key=lambda x: x[1], reverse=True)[1:num\_recommendations+1]

return [destinations\_df.iloc[i[0]]['DestinationID'] for i in similar\_destinations]

except Exception as e:

st.error(f"Error in content-based recommendations: {str(e)}")

return []

Data Cleaning :

def recommend\_destinations(user\_id, userhistory\_df, destinations\_df, cosine\_sim):

"""

Recommends top 5 destinations for a given user based on similarity scores.

Args:

- user\_id: ID of the user.

- userhistory\_df: User history DataFrame containing 'UserID' and 'DestinationID'.

- destinations\_df: Destinations DataFrame containing destination details.

- cosine\_sim: Cosine similarity matrix for destinations.

Returns:

- DataFrame with recommended destinations and their details.

"""

# Get the destinations the user has visited

visited\_destinations = userhistory\_df[userhistory\_df['UserID'] == user\_id]['DestinationID'].values

# Calculate similarity scores for visited destinations

similar\_scores = np.sum(cosine\_sim[visited\_destinations - 1], axis=0)

# Recommend the top 5 destinations the user hasn't visited yet

recommended\_destinations\_idx = np.argsort(similar\_scores)[::-1]

recommendations = []

for idx in recommended\_destinations\_idx:

if destinations\_df.iloc[idx]['DestinationID'] not in visited\_destinations:

# Append detailed information for each recommendation

recommendations.append(destinations\_df.iloc[idx][[

'DestinationID', 'Name', 'State', 'Type', 'Popularity', 'BestTimeToVisit'

]].to\_dict())

if len(recommendations) >= 100:

break

# Convert recommendations to a DataFrame

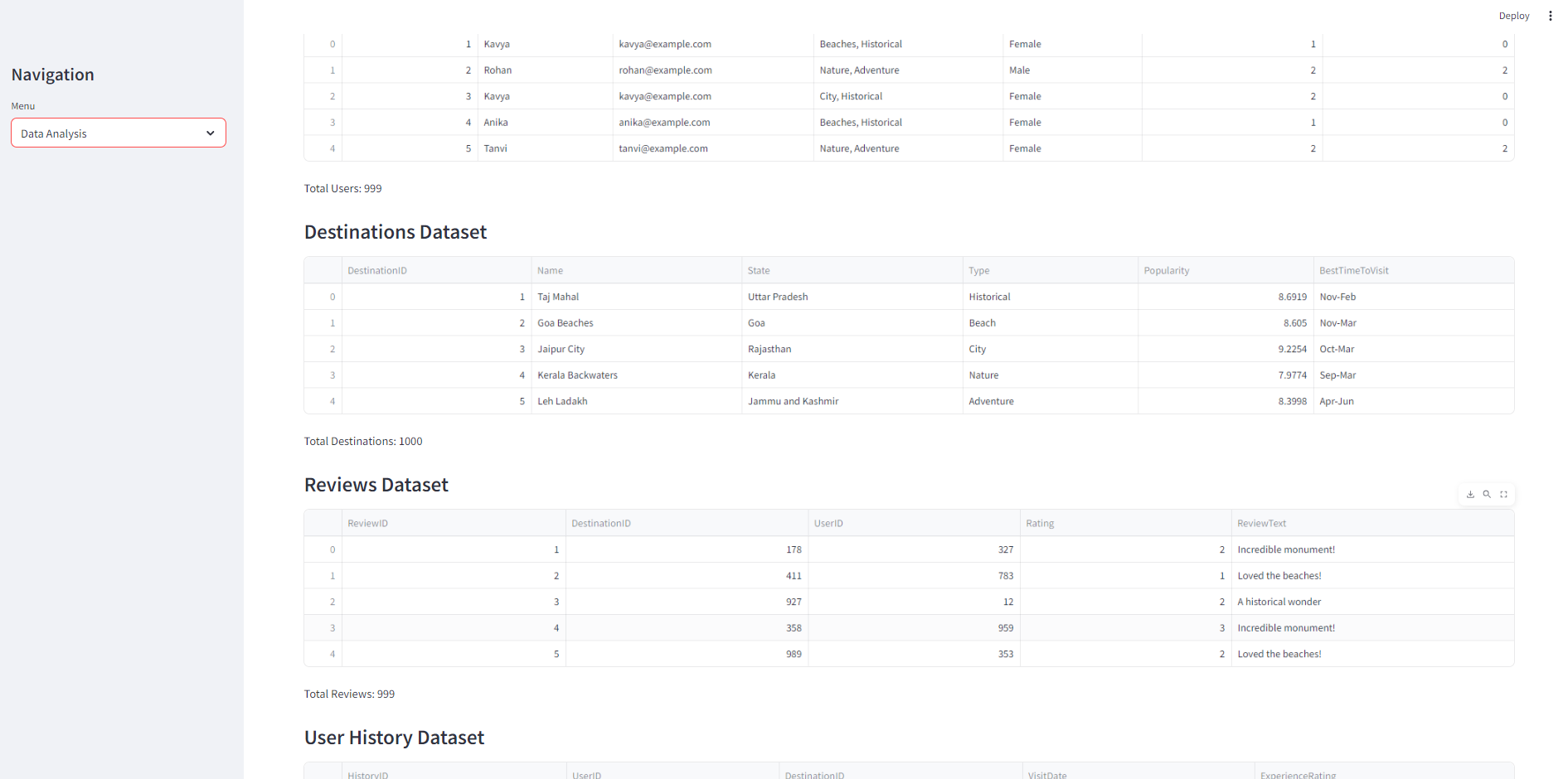
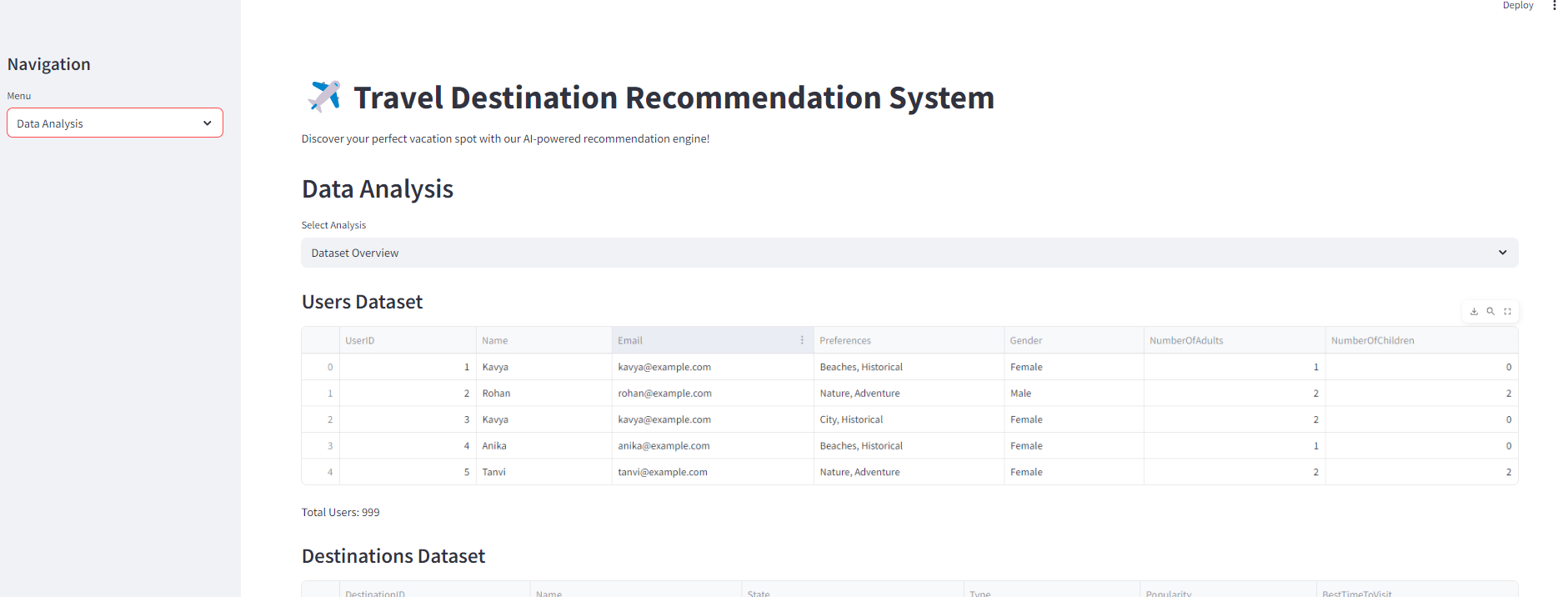
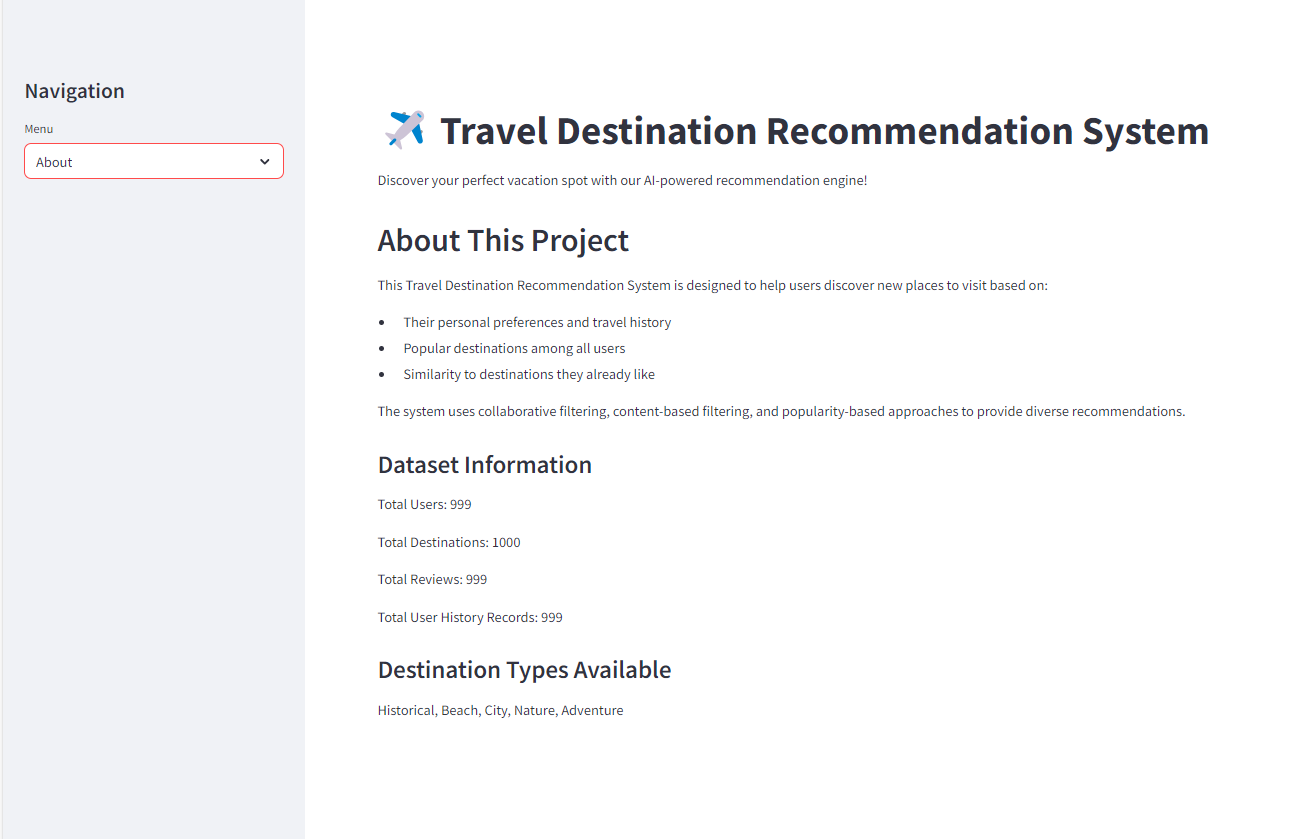
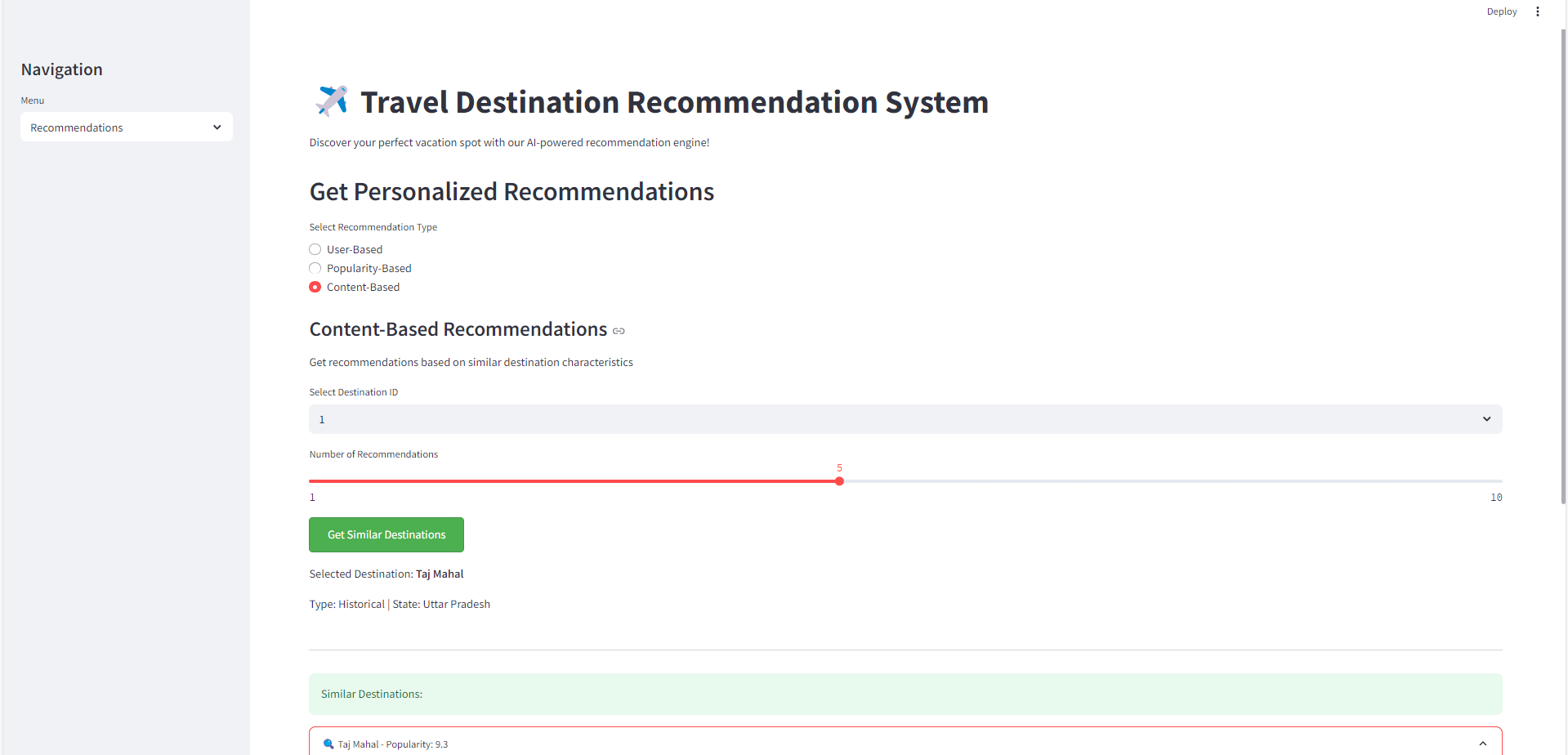
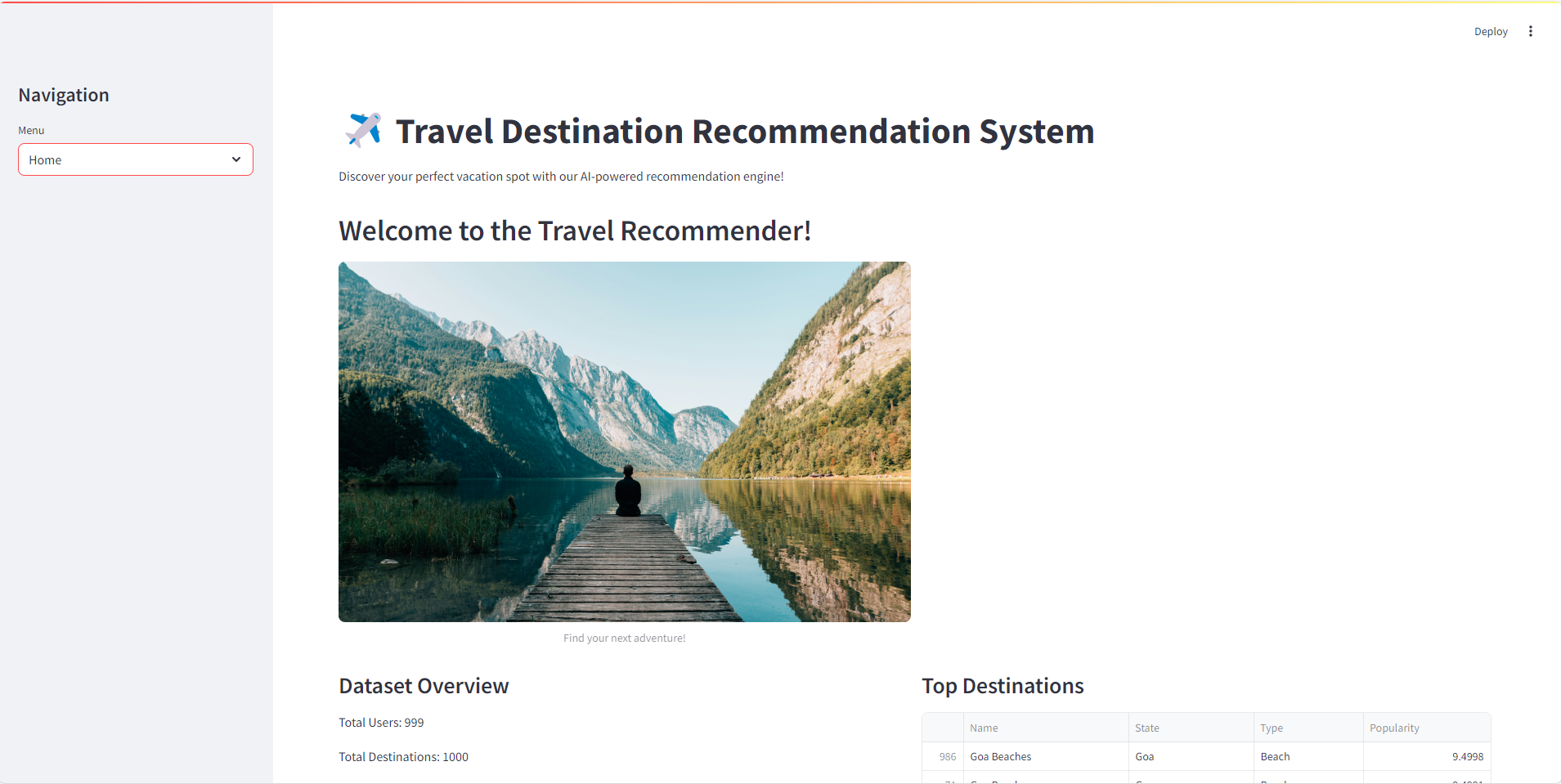
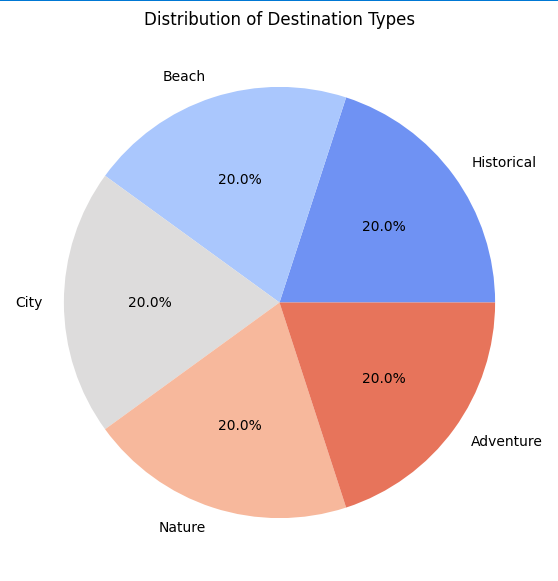
return pd.DataFrame(recommendations)

# Example: Recommend destinations for user with ID 1

recommended\_destinations = recommend\_destinations(5, userhistory\_df, destinations\_df, cosine\_sim)

# Display recommendations

recommended\_destinationsata



Submitted By-  
Gauri Uday Gotad (23030421995506)

Sign of Faculty Incharge