# A DECISION TREE BASED RECOMMENDATION SYSTEM FOR TOURISTS

*A Seminar report submitted in partial fulfillment of the requirements* *for the award of the degree of*

**BACHELOR OF TECHNOLOGY**

in

**COMPUTER SCIENCE AND ENGINEERING**

*by*

**A.SAMYUKTHA (20S41A0506)**

**D.RAMU (20S41A0528)**

**A.ASHWINI (20S41A0507)**

**K.RANJITH KUMAR (20S41A0548)**



**Department of Computer Science & Engineering**

**VAAGESWARI COLLEGE OF ENGINEERING**

**(Affiliated to JNTU Hyderabad &Approved by AICTE New Delhi)**

**Ramakrishna colony, Karimnagar-505527**

**2023**

**Department of Computer Science & Engineering**

**VAAGESWARI COLLEGE OF ENGINEERING**

**(Affiliated to JNTU Hyderabad &Approved by AICTE New Delhi)**

**Ramakrishna colony, Karimnagar-505527**



**CERTIFICATE**

This is to certify that the seminar report entitled **“A DECISION TREE BASED RECOMMENDATION SYSTEM FOR TOURISTS”** submitted by the following students in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in CSE and is a bonafide record of the work performed by.

## A.SAMYUKTHA (20S41A0506)

## D.RAMU (20S41A0528)

## A.ASHWINI (20S41A0507)

## K.RANJITH KUMAR (20S41A0548)

## Internal guide Head of the department

## MRS.Y.SUSHEELA Dr.N.CHANDRAMOULI

**Associate professor Associate professor**

**Principal External examiner**

**Dr.CH.SRINIVAS**

**DECLARATION**

We hereby declare that the project titled “**A DECISION TREE BASED RECOMMENDATION SYSTEM FOR TOURISTS”** submitted to Vaageswari College of Engineering, affiliated to Jawaharlal Nehru Technological University Hyderabad (JNTUH) for the award of the Degree of Bachelor of Technology in CSE is a result of original research carried-out in this work. It is further declared that the report or any part thereof has not been previously submitted to any University or Institute for the award of degree or diploma.

1. **Name of the Student :SAMYUKTHA ANTAGIRI**

**Hall ticket Number : 20S41A0506**

1. **Name of the Student : RAMU DUMPATI**

**Hall ticket Number : 20S41A0528**

1. **Name of the Student : ASHWINI AREPALLI**

**Hall ticket Number : 20S41A0507**

1. **Name of the Student :RANJITH KASA**

**Hall ticket Number : 20S41A0548**

**Date:**

**ACKNOWLEDGEMENT**

We wish to pay our sincere thanks to **Dr.Ch. Srinivas**, Principal, Vaageswari College of Engineering, Karimnagar, for providing all required facilities and his support during the Seminar.

We would like to thank **Dr. N.Chandramouli,** Associate. Professor and HOD of the Computer Science and Engineering Department for his valuable suggestions during the seminar.

We sincerely extend our thanks to the project guide, **Mrs.Y.Susheela**, Associate Professor of Computer Science and Engineering Department, for sparing his valuable time in guiding the project work and giving feedback with many useful suggestions during the project work.

We are also conveying our heartfelt thanks to the Institute authority, Department, Library, and Laboratory staff of Vaageswari College of Engineering for their co-operation during our seminar. We thank our beloved friends for their help and encouragement regarding the concepts and presentation.

**A.SAMYUKTHA (20S41A0506)**

**D.RAMU (20S41A0528)**

**A.ASHWINI (20S41A0507)**

**K.RANJITH (20S41A0548)**

**INDEX**

**TOPICS Page No’s**

**ABSTRACT**

**CHAPTER-1: INTRODUCTION 1-2**

**CHAPTER-2: LITERATURE SURVEY 3-5**

**CHAPTER-3: SYSTEM ANALYSIS**

**3.1 Existing System 6**

**3.2 Proposed System 6**

**CHAPTER-4: SYSTEM REQUIREMENTS**

**4.1 Hardware Requirements 7**

**4.2 Software Requirements 7**

**CHAPTER-5: SYSTEM STUDY**

**5.1 Feasibility Study 8**

**5.2 Feasibility Analysis 8-9**

**CHAPTER-6: SYSTEM ARCHITECTURE**

**6.1 SYSTEM ARCHITECTURE 10**

**6.2 UML Diagrams 10-14**

**CHAPTER-7: INPUT AND OUTPUT DESIGN**

**7.1 Input Design 15**

**7.2 Output Design 16**

**CHAPTER-8: IMPLEMENTATION**

**8.1 MODULES 17**

**8.1.1 Module Description 17**

**CHAPTER-9: SOFTWARE ENVIRONMENT**

**9.1 PYTHON 18-36**

**9.2 Source Code 36-54**

**CHAPTER-10: RESULTS/DISCUSSIONS**

**10.1 System Test 55-61**

**10.2 Output Screens 62-64**

**CHAPTER-11: CONCLUSION 65**

**CHAPTER-12: REFERENCES 66-67**

**LIST OF FIGURES**

**FIGNO FIGURE NAME PAGE NO**

**1**

**DECISION TREE BASED RECOMMENDATION SYSTEM FOR TOURISTS**

**ABSTRACT**

Choosing a tourist destination from the information that is available on the Internet and through other sources is one of the most complex tasks for tourists when planning travel, both before and during travel. Previous Travel Recommendation Systems (TRSs) have attempted to solve this problem. However, some of the technical aspects such as system accuracy and the practical aspects such as usability and satisfaction have been neglected. To address this issue, it requires a full understanding of the tourists’ decision-making and novel models for their information search process. This paper proposes a novel human-centric TRS that recommends destinations to tourists in an unfamiliar city. It considers both technical and practical aspects using a real-world data set we collected. The system is developed using a two-step feature selection method to reduce number of inputs to the system and recommendations are provided by decision tree C4.5. The experimental results show that the proposed TRS can provide personalized recommendation on tourist destinations that satisfy the tourists.

**CHAPTER-1**

**INTRODUCTION**

A ​decision tree-based recommendation system for ​tourists is a ​predictive model that assists ​travelers in making informed decisions about their ​destinations, activities, and accommodations. This system utilizes decision tree algorithms to analyze user preferences, historical data, and other relevant factors to provide personalized recommendations.

The decision tree algorithm works by constructing a tree-like structure, where each node represents a decision based on specific features or attributes. At each node, the algorithm evaluates different criteria and splits the data into subsets based on these criteria. The process continues until the algorithm reaches the leaf nodes, which present the final recommendations to the users.

To develop a decision tree-based recommendation system for tourists, several steps are involved:

1. **Data Collection**: Relevant data is collected from various sources, such as travel websites, user reviews, and tourism databases. This data includes information about popular destinations, attractions, user preferences, ratings, and feedback.
2. **Feature Selection**: The collected data is transformed into meaningful features that can be used by the decision tree algorithm. These features may include location, budget, travel dates, accommodation preferences, activities of interest, transportation options, and more.
3. **Training the Decision Tree**: The decision tree algorithm is trained using the collected data. It learns from historical patterns and relationships between user preferences and recommended options. The algorithm determines the optimal splits and decisions based on the features and outcomes.
4. **Predicting Recommendations**: Once the decision tree is trained, it can utilize the input provided by users to predict their preferences and generate personalized recommendations. Users provide their preferences through a user interface, and the decision tree evaluates the input against the learned patterns to suggest suitable destinations, activities, accommodations, and other travel-related options.
5. **Evaluation and Refinement**: The recommendation system's performance is assessed using various evaluation metrics, such as accuracy, precision, and recall. If necessary, the system can be refined by incorporating new data, improving feature selection, or adjusting the decision tree algorithm parameters.

In summary, a decision tree-based recommendation system for tourists leverages the power of data analysis and machine learning algorithms to provide personalized and informed travel recommendations. By considering user preferences, historical data, and relevant factors, this system assists travelers in making well-informed decisions and optimizing their travel experiences.

.

# Proposed architecture for tourism recommender system.

# Fig1:Architecture for tourism recommendation system

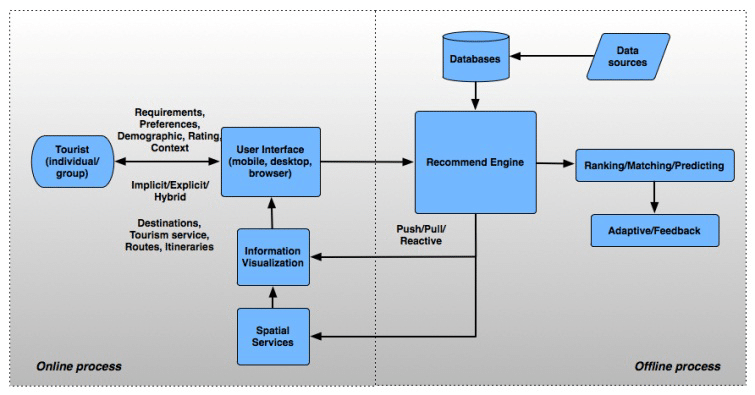
**A. Recommendation System**

A recommendation system (RS), a subset of Decision Support Systems (DSS), is a tool that can recommend an item based on the aggregated information of the user’s preferences . It supports users by providing valuable information to assist them in their decision-making processes based on their priorities and concerns . RS plays an important role and is common in many popular e-commerce websites, such as Amazon, Netflix, Pandora, etc. The e-commerce RSs suggest items to the user whichinvolve news, articles, people, URLs, and so on].

**B. Travel Recommendation Systems**

Tourism is a leisure activity that involves complex decision processes, for example, selecting destinations, attractions, activities, and services. Thus, TRS attract the attention of many researchers from the fields of both academics and industry.

Most TRSs present the result with the use of spatial web services such as the Google Map API



**FIG2:General framework of the travel recommendation system**

**C. Recommendation techniques**

According to, RS can be classified by the degree of personalization, including the usefulness and accuracy

of the recommendations. The degree of personalization can be defined from low to high, including non-

personalization, ephemeral personalization (short term), and persistent personalization.

**ADVANTAGES OF DECISION TREE FOR TOURISM**

**A ​decision tree-based recommendation system for ​tourists offers several advantages:**

1. ​**Interpretable Recommendations:** Decision trees provide transparent and interpretable recommendations.
2. ​**Personalized Recommendations:** Decision trees can be tailored to individual tourists' preferences, enabling personalized recommendations based on their specific needs and interests.
3. ​**Scalability:** Decision trees can handle large amounts of data efficiently. With the increasing availability of tourism-related information and user-generated content, decision trees can handle a vast number of variables and provide recommendations based on multiple criteria simultaneously.
4. **Flexibility:** Decision trees can incorporate both categorical and numerical data, making them versatile for handling various types of tourist-related variables..
5. **Quick Decision-Making:** Decision trees excel at quickly making recommendations by evaluating different paths and options in a step-by-step manner.

**CHAPTER-2**

**LITERATURE SURVEY**

**Literature Survey**

The literature survey on decision tree-based recommendation systems for tourists involves an exploration of academic databases, including PubMed, IEEE Xplore, Springer, ScienceDirect, and Google Scholar, using keywords such as "decision tree," "recommendation system," and "tourism recommendation." Through a systematic review of titles, abstracts, and full texts, the survey aims to identify and synthesize relevant research. Additionally, consideration is given to recent publications, conference proceedings, and the exploration of citations and references to ensure a comprehensive overview of the field. The findings will be organized in a structured literature review, highlighting key methodologies, trends, and potential gaps in the existing literature.

Semantic network investigation is the focal point of this article as a technique for inspecting issues with the convenience of the Web for movement data search. Web convenience is characterized as how much the psychological models of data makers and shoppers, which depend on their cognizance of the  
design and content of data tracked down on the Web, are viable. The misalignment of mental models between the movement business promoters and travelers incited the Internet's lamentable comfort as a wellspring of development information. Using semantic network analysis, comparing these two  
types of mental models may not only reveal differences between them, but it may also provide advice on how to effectively distribute information via the Internet. The authors present preliminary results for the semantic network and use semantic network analysis to investigate the mental models that travel  
information providers use when they sell their destinations online.

**CHAPTER-3**

**SYSTEM ANALYSIS**

In this report, we will conduct a system analysis of a ​decision tree-based recommendation system specifically designed for ​tourists. We will explore the key components, functionalities, and advantages of this system, as well as potential challenges and limitations that need to be addressed for successful implementation.

**EXISTING SYSTEM**

A decision tree-based recommendation system for tourists is designed to provide personalized recommendations based on user preferences and other inputs. The system utilizes a decision tree model to generate recommendations by analyzing various factors. Here are the existing components of such a system:

1. **Approach:**
   * Typically, existing TRSs may utilize algorithms like collaborative filtering, content-based filtering, or hybrid methods to recommend tourist destinations.
2. **Technical Aspects:**
   * Accuracy: Existing systems may face challenges in achieving high accuracy due to the complexity of tourist preferences and evolving user behavior.
3. **Practical Aspects:**
   * Usability: Some existing TRSs might have limitations in terms of user-friendly interfaces, making it difficult for users to interact seamlessly.
4. **Data Utilization:**
   * Previous systems may not effectively utilize real-world data or may rely on limited datasets, impacting the diversity and relevance of recommendations.
5. **Decision-Making Model:**
   * The decision-making process in traditional TRSs may not fully align with the tourists' decision-making patterns, leading to less personalized recommendations.

**DISADVANTAGES OF EXISTING SYSTEM**

The existing system of a ​decision tree-based recommendation system for ​tourists has several disadvantages, including:

1. **Limited Complexity:** Decision trees have a relatively low capacity to handle ​complex relationships and ​dependencies among different ​variables. This limitation can result in oversimplified recommendations, especially when dealing with diverse and intricate tourist preferences and contexts.
2. **Lack of Continuous Learning**: Decision trees are static models that do not support continuous learning or adaptation to changing data.
3. **Overfitting:** Decision trees are prone to overfitting when the model becomes overly specific to the training data. This means that the recommendations may not generalize well to new or unseen instances, resulting in poor performance in real-world scenarios.
4. **Data Quality and Quantity:** Decision tree models heavily rely on the quality and quantity of the available data. Insufficient or noisy data can lead to biased or unreliable recommendations. Moreover, decision trees may struggle to handle missing or incomplete data, which is common in tourism datasets.
5. **Interpretability Challenges:** While decision trees are often praised for their interpretability, complex decision trees can become challenging to interpret and understand. As decision trees grow in size and depth, it becomes harder for users to grasp the rationale behind specific recommendations, reducing transparency and trust in the system.
6. **Inability to Capture Contextual Information:** Decision tree models generally focus on individual attributes or features without considering the broader context. In tourism, where recommendations heavily rely on contextual factors such as weather, time of year, traveler demographics, and local events, decision tree-based systems may overlook crucial information, leading to suboptimal recommendations.
7. **Scalability Issues**: Decision tree-based recommendation systems may face scalability challenges when dealing with large datasets or high-dimensional feature spaces. As the number of variables and instances increases, the computational complexity of building and traversing decision trees can become prohibitive, affecting system responsiveness.

To mitigate these disadvantages, researchers and practitioners are exploring hybrid approaches, such as combining decision trees with other machine learning techniques like ensemble methods or deep learning networks, to overcome these limitations and enhance the accuracy and effectiveness of tourism recommendation systems.

**PROPOSED SYSTEM**

The proposed decision tree-based recommendation system aims to overcome the limitations of existing tourist guidance systems by introducing a more sophisticated and personalized approach. This system utilizes decision tree algorithms to analyze a diverse set of input variables, including travel history, demographic information, and specific interests, enabling a nuanced understanding of individual preferences.

Unlike conventional systems that offer generic suggestions, the proposed model systematically processes these variables, creating a tailored decision-making path for each tourist. This personalized approach enhances the relevance and accuracy of recommendations, ensuring that users receive suggestions that align closely with their unique preferences and evolving interests.

The adaptability of decision tree models allows the system to learn from user interactions over time, continuously improving its recommendation accuracy. This dynamic aspect ensures that the system stays relevant and responsive to changes in a tourist's preferences, providing an enhanced and up-to-date experience.

Furthermore, the decision tree-based system addresses data privacy concerns by incorporating robust security measures in handling and processing user data. This emphasis on privacy safeguards user trust and aligns with ethical considerations in handling sensitive information.

In summary, the proposed system introduces a more advanced and personalized approach to tourist recommendations, leveraging decision tree algorithms to provide accurate, adaptable, and privacy-conscious suggestions, ultimately enhancing the overall travel experience for users.

**ADVANTAGES OF PROPOSED SYSTEM**

1**.personalization:**The system provides highly personalized recommendations by analyzing individual preferences, travel history, and specific interests. This tailored approach ensures that users receive suggestions aligned with their unique tastes, enhancing their overall travel experience.

2.**Accuracy:**Decision tree algorithms excel at handling complex datasets, leading to more accurate recommendations. By systematically evaluating a diverse set of input variables, the system can make informed decisions, improving the precision of tourist suggestions.

3.**Adaptability:**The system's dynamic nature allows it to adapt and evolve over time. As it learns from user interactions, it becomes more attuned to changes in preferences, ensuring that recommendations stay relevant and reflective of a tourist's evolving interests.

**4.Efficiency:**Decision tree models are computationally efficient, making them suitable for real-time recommendation scenarios. This efficiency contributes to a seamless and responsive user experience, particularly important in the fast-paced context of travel planning.

5.**Privacy Consideration**s:The proposed system incorporates robust security measures to address data privacy concerns. By ensuring the responsible handling and processing of user data, the system maintains user trust and complies with ethical standards in data management.

**6**.**User Satisfaction:** With its personalized and accurate recommendations, the system is designed to enhance user satisfaction. Tourists are more likely to discover and engage with activities and attractions that align with their interests, contributing to a more fulfilling and enjoyable travel experience.

In summary, the decision tree-based recommendation system stands out for its ability to deliver personalized, accurate, and adaptable suggestions while addressing privacy concerns, ultimately contributing to an improved and more satisfying tourist guidance experience.

**CHAPTER-4**

**SYSTEM REQUIREMENTS**

**HARDWARE REQUIREMENTS:**

* System : Pentium Dual Core.
* Hard Disk : 500 GB.
* Ram : 1GB.

**SOFTWARE REQUIREMENTS:**

* Operating system : Windows 11.or more
* Coding Language : Python

**CHAPTER-5**

**SYSTEM STUDY**

A decision tree-based recommendation system for tourists involves using a tree-like model to guide users towards recommended destinations or activities based on their preferences, behavior, and other relevant factors. Here's a system study outline for such a recommendation system:

**FEASABILITY STUDY**

## 5.1 FEASIBILITY STUDY:

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

## FEASIBILITY ANALYSIS

Three key considerations involved in the feasibility analysis are

* + - ECONOMICAL FEASIBILITY
    - TECHNICAL FEASIBILITY
    - SOCIAL FEASIBILITY

## ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

## TECHNICAL FEASABILITY

## The technical feasibility of a decision tree-based recommendation system for tourists depends on various factors, including the availability of data, computational resources, and the complexity of the system.

## Assessing these factors will help in determining the technical feasibility of implementing a decision tree-based recommendation system for tourists. Decision trees are often a feasible choice for recommendation systems due to their simplicity, interpretability, and lower computational requirements, making them suitable for various practical applications.

## SOCIAL FEASIBILITY

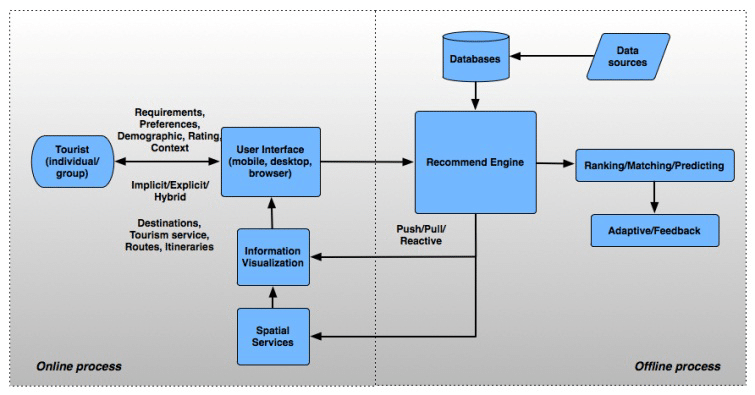
The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

**CHAPTER-6**

**SYSTEM DESIGN**

**SYSTEM ARCHITECTURE**

The system design of a decision tree-based recommendation system for tourists involves outlining the architecture, components, and processes required for the system to function effectively.



**Fig2:Architecture for travel recommendation system**

This system design provides a comprehensive overview of the decision tree-based recommendation system for tourists, covering key components, interactions, and considerations for implementation. The design can be adapted based on specific requirements, technologies, and constraints of the target environment.

**UML DIAGRAMS**

Unified Modeling Language (UML) is a standardized modeling language used in software engineering for visualizing, specifying, constructing, and documenting the artifacts of a system. UML diagrams provide a way to represent the various aspects of a system, such as its structure, behavior, and interactions. There are several types of UML diagrams, each serving a specific purpose.

1. The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.
2. The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.
3. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.
4. The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

**USE CASE DIAGRAM**

A use case diagram is used to represent the dynamic behavior of a system. It encapsulates the system's functionality by incorporating use cases, actors, and their relationships. It models the tasks, services, and functions required by a system/subsystem of an application. It depicts the high-level functionality of a system and also tells how the user handles a system. The main purpose of a use case diagram is to portray the dynamic aspect of a system. It accumulates the system's requirement, which includes both internal as well as external influences. It invokes persons, use cases, and several things that invoke the actors and elements accountable for the implementation of use case diagrams. It represents how an entity from the external environment can interact with a part of the system.



**CLASS DIAGRAM**

The class diagram depicts a static view of an application. It represents the types of objects residing in the system and the relationships between them. A class consists of its objects, and also it may inherit from other classes. A class diagram is used to visualize, describe, document various different aspects of the system, and also construct executable software code.

It shows the attributes, classes, functions, and relationships to give an overview of the software system. It constitutes class names, attributes, and functions in a separate compartment that helps in software development. Since it is a collection of classes, interfaces, associations, collaborations, and constraints, it is termed as a structural diagram.



**SEQUENCE DIAGRAM**

[UML](https://en.wikipedia.org/wiki/Unified_Modeling_Language) Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration. Sequence Diagrams are time focus and they show the order of the interaction visually by using the vertical axis of the diagram to represent time what messages are sent and when.

* Model high-level interaction between active objects in a system
* Model the interaction between object instances within a collaboration that realizes a use case
* Model the interaction between objects within a collaboration that realizes an operation
* Either model generic interactions (showing all possible paths through the interaction) or specific instances of a interaction (showing just one path through the interaction).



**COLLABORATION DIAGRAM**

The collaboration diagram is used to show the relationship between the objects in a system. Both the sequence and the collaboration diagrams represent the same information but differently. Instead of showing the flow of messages, it depicts the architecture of the object residing in the system as it is based on object-oriented programming. An object consists of several features. Multiple objects present in the system are connected to each other. The collaboration diagram, which is also known as a communication diagram, is used to portray the object's architecture in the system.

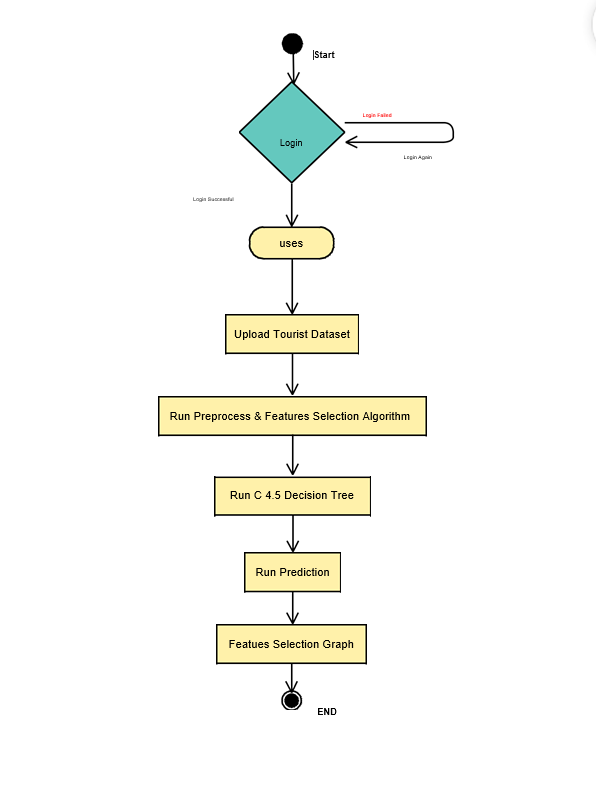
The collaborations are used when it is essential to depict the relationship between the object. Both the sequence and collaboration diagrams represent the same information, but the way of portraying it quite different. The collaboration diagrams are best suited for analyzing use cases.



**ACTIVITY DIAGRAM**

In UML, the activity diagram is used to demonstrate the flow of control within the system rather than the implementation. It models the concurrent and sequential activities.

The activity diagram helps in envisioning the workflow from one activity to another. It put emphasis on the condition of flow and the order in which it occurs. The flow can be sequential, branched, or concurrent, and to deal with such kinds of flows, the activity diagram has come up with a fork, join, etc.It is also termed as an object-oriented flowchart. It encompasses activities composed of a set of actions or operations that are



**COMPONENT DIAGRAM**

A component diagram is used to break down a large object-oriented system into the smaller components, so as to make them more manageable. It models the physical view of a system such as executables, files, libraries, etc. that resides within the node.

It visualizes the relationships as well as the organization between the components present in the system. It helps in forming an executable system. A component is a single unit of the system, which is replaceable and executable. The implementation details of a component are hidden, and it necessitates an interface to execute a function. It is like a black box whose behavior is explained by the provided and required interfaces.



**DEPLOYMENT DIAGRAM**

A UML deployment diagram depicts a static view of the run-time configuration of hardware nodes and the software components that run on those nodes. UML deployment diagrams show the hardware for your system, the software that is installed on that hardware, and the middleware used to connect the disparate machines to one another.

The main purpose of the deployment diagram is to represent how software is installed on the hardware component. It depicts in what manner a software interacts with hardware to perform its execution.

Both the deployment diagram and the component diagram are closely interrelated to each other as they focus on software and hardware components.



# 

# CHAPTER-7

# INPUT OUTPUT DESIGNS

## 7.1INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

* + - What data should be given as input?
    - How the data should be arranged or coded?
    - The dialog to guide the operating personnel in providing input.
    - Methods for preparing input validations and steps to follow when error occur.

### **INPUT OBJECTIVES**

Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

1. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.
2. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow.

### 

### **OUTPUT DESIGN**

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system’s relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.
2. Select methods for presenting information.
3. Create document, report, or other formats that contain information produced by the system. The output form of an information system should accomplish one or more of the following objectives.
   * Convey information about past activities, current status or projections of the
   * Future.
   * Signal important events, opportunities, problems, or warnings.
   * Trigger an action.
   * Confirm an action.

**CHAPTER-8**

**IMPLEMENTATION**

**MODULE**

USER

**MODULES DESCRIPTION:**

In this paper author is implementing C4.5 decision tree algorithm with MRMR features selection to recommend travel areas to tourist by using dataset from past tourist experiences. All existing algorithms such as collaborative or content filtering algorithms uses current user past experience data to recommend him new locations. These algorithms will not work if this current user has no past experiences data. Below are the dataset columns or attributes taken from previous users to build model. This data set is populated by crawling TripAdvisor.com. Reviews on destinations in 10 categories mentioned across East Europe are considered. Each traveller rating is mapped as Excellent (4), Very Good (3), Average (2), Poor (1), and Terrible (0) and average rating is used against each category per user. Dataset columns and values userid,art\_galleries,dance\_clubs,juice\_bars,restaurants,museums,resorts,parks\_picnic,spots,beaches,theaters,religious\_institutio ns,location Above are the column names and below are the column values User 1,0.93,1.8,2.29,0.62,0.8,2.42,3.19,2.79,1.8 2,2.42,Amsterdam\_Heining\_2

User 2,1.02,2.2,2.66,0.64,1.42,3.18,3.21,2.63,1. 86,2.32,Amsterdam\_Jachthaven\_ijbur User 3,1.22,0.8,0.54,0.53,0.24,1.54,3.18,2.8,1.3 1,2.5,Amsterdam\_Bert\_Haanstra\_Kad User 4,0.45,1.8,0.29,0.57,0.46,1.52,3.18,2.96,1. 57,2.86,Amsterdam\_Ruigoord\_Ker In above values first column is USER\_ID and second column is ART\_GALLERIES and third is DANCE CLUB etc and for each column user had given rating from 4 to 0 and 4 means Excellent service. Now using above values we can build C4.5 decision tree and prediction will be done using below test values 'User 122',0.93,1.8,2.29,0.62,0.8,2.42,3.19,2.79, 1.82,2.42,? 'User 222',1.02,2.2,2.66,0.64,1.42,3.18,3.21,2.63 ,1.86,2.32,? 'User 3222',1.22,0.8,0.54,0.53,0.24,1.54,3.18,2.8 ,1.31,2.5,? 'User 4222',0.45,1.8,0.29,0.57,0.46,1.52,3.18,2.9 6,1.57,2.86,? 'User 522',0.51,1.2,1.18,0.57,1.54,2.02,3.18,2.78 ,1.18,2.54,? In above test values new user has given values to look for location which has above service rating but new user don’t know which location provides such services so he will put question mark and when we upload above test values to decision tree then it will take decision and predict best location and inform to user.

**CHAPTER-9**

**SOFTWARE ENVIRONMENT**

**What is Python?**

* Python is a High level, structured, open-source programming language that can be used for a wide variety of programming tasks.
* Python within itself is an interpreted programming language that is automatically compiled into bytecode before execution.
* It is also a dynamically typed language that includes (but does not require one to use) object-oriented features.
* NASA has used Python for its software systems and has adopted it as the standard scripting language for its Integrated Planning System.
* Python is also extensively used by Google to implement many components of its Web Crawler and Search Engine & Yahoo! for managing its discussion groups.

**History of Python**

* Python was created by Guido Van Rossum.
* The design began in the late 1980s and was first released in February 1991.

**Why the name Python?**

No. It wasn't named after a dangerous snake. Rossum was fan of a comedy series from late 70s. The name "Python" was adopted from the same series "Monty Python's Flying Circus".

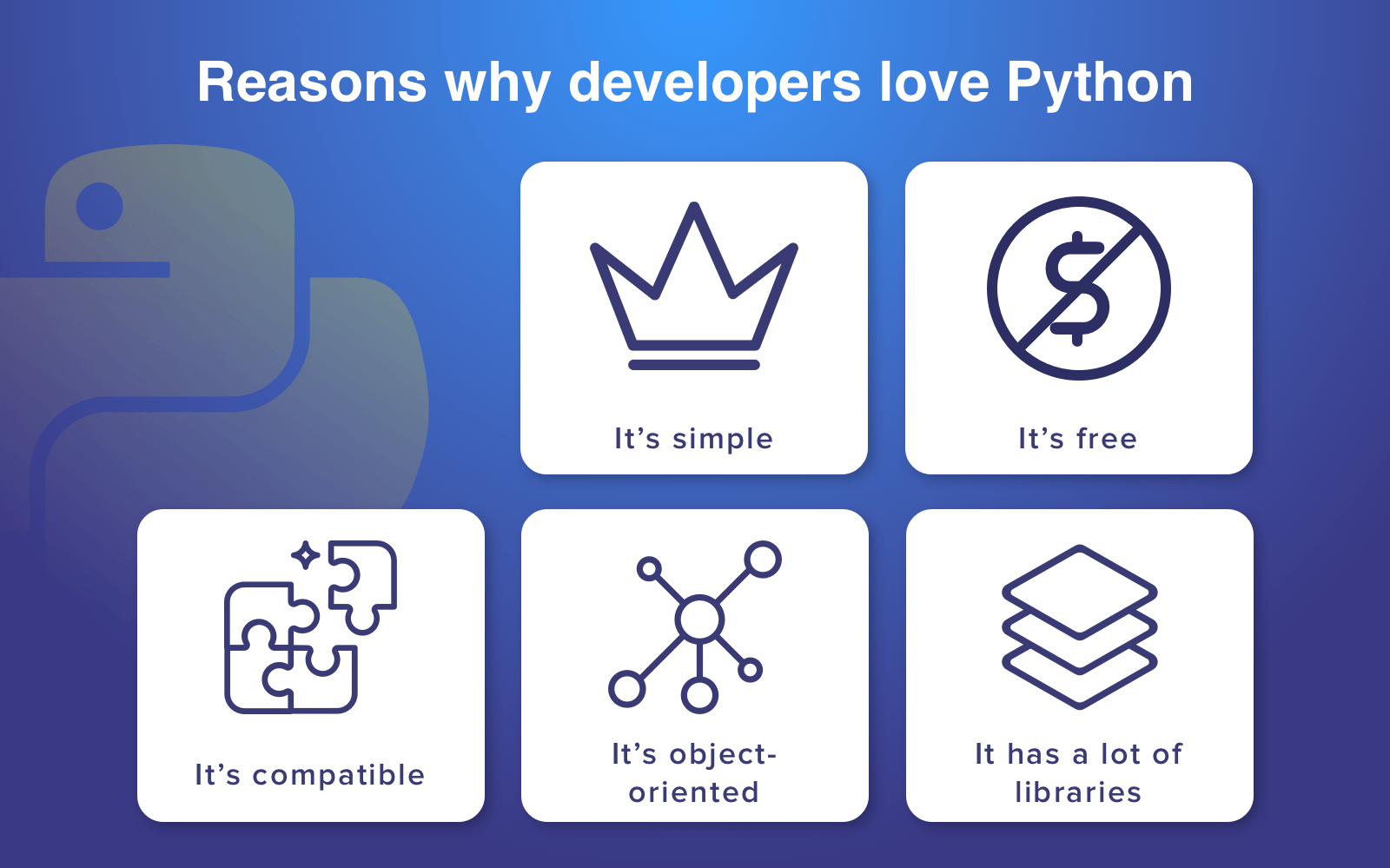
**Python Version History**

Implementation started - December 1989

Internal releases – 1990

|  |  |
| --- | --- |
| **Version No.** | **Date of Released** |
| 0.9 | February 20, 1991 |
| 1.0 | January, 1994 |
| 2.0 | October 16, 2000 |
| 3.0 | December 3, 2008 |
| 3.1 | June 27, 2009 |
| 3.2 | February 20, 2011 |
| 3.3 | September 29, 2012 |
| 3.4 | March 16, 2014 |
| 3.5 | September 13, 2015 |
| 3.6 | December 23, 2016 |
| 3.7 | June 27, 2018 |

**Features of Python Programming**



1. **A simple language which is easier to learn**

* Python has a very simple and elegant syntax.
* It's much easier to read and write Python programs compared to other languages like: C++, Java, C#.
* Python makes programming fun and allows you to focus on the solution rather than syntax.
* If you are a newbie, it's a great choice to start your journey with Python.

1. **Free and open-source**

* You can freely use and distribute Python, even for commercial use.
* Not only you can use and distribute software’s written in it, you can even make changes to the Python's source code.
* Python has a large community constantly improving it in each iteration.

1. **Portability**

* You can move Python programs from one platform to another and run it without any changes.
* It runs seamlessly on almost all platforms including Windows, Mac OS and Linux.

1. **Extensible and Embeddable**

* Suppose an application requires high performance. You can easily combine pieces of C/C++ or other languages with Python code.
* This will give your application high performance as well as scripting capabilities which other languages may not provide out of the box.

1. **A high-level, interpreted language**

* Unlike C/C++, you don't have to worry about daunting tasks like memory management, garbage collection and so on.
* Likewise, when you run Python code, it automatically converts your code to the language your computer understands. You don't need to worry about any lower-level operations.

1. **Large standard libraries to solve common tasks**

* Python has several standard libraries which makes life of a programmer much easier since you don't have to write all the code yourself.
* For example: Need to connect MySQL database on a Web server? You can use MySQL dB library using import MySQL db.
* Standard libraries in Python are well tested and used by hundreds of people. So, you can be sure that it won't break your application.

1. **Object-oriented**

* Everything in Python is an object. Object oriented programming (OOP) helps you solve a complex problem intuitively.
* With OOP, you can divide these complex problems into smaller sets by creating objects.

**4 Reasons to Choose Python as First Language**

1. **Simple Elegant Syntax**

* Programming in Python is fun. It's easier to understand and write Python code. **Why?** The syntax feels natural. Take this source code for an example:

a = 2

b = 3

sum = a + b

print(sum)

* Even if you have never programmed before, you can easily guess that this program adds two numbers and prints it.

1. **Not overly strict**

* You don't need to define the type of a variable in Python. Also, it's not necessary to add semicolon at the end of the statement.
* Python enforces you to follow good practices (like proper indentation). These small things can make learning much easier for beginners.

1. **Expressiveness of the language**

* Python allows you to write programs having greater functionality with fewer lines of code. Here's a link to the source code of Tic-tac-toe game with a graphical interface and a smart computer opponent in less than 500 lines of code. This is just an example. You will be amazed how much you can do with Python once you learn the basics.

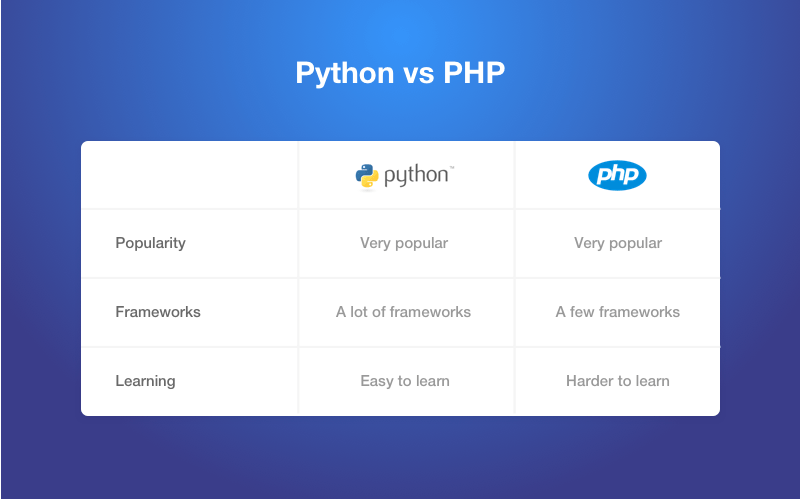
1. **Great Community and Support**

* Python has a large supporting community. There are numerous active forums online which can be handy if you are stuck. Some of them are:
  + - Google Forum for Python
    - Python Questions - Stack Overflow

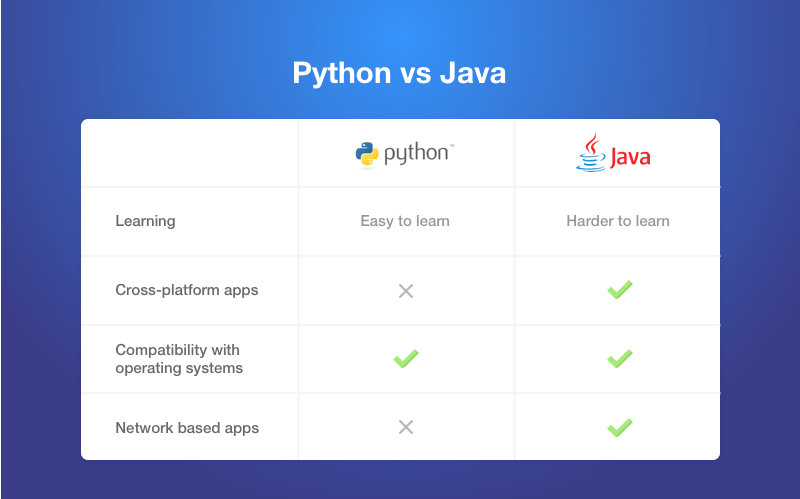
## Python vs PHP

From the development point of view, PHP is a web-oriented language.

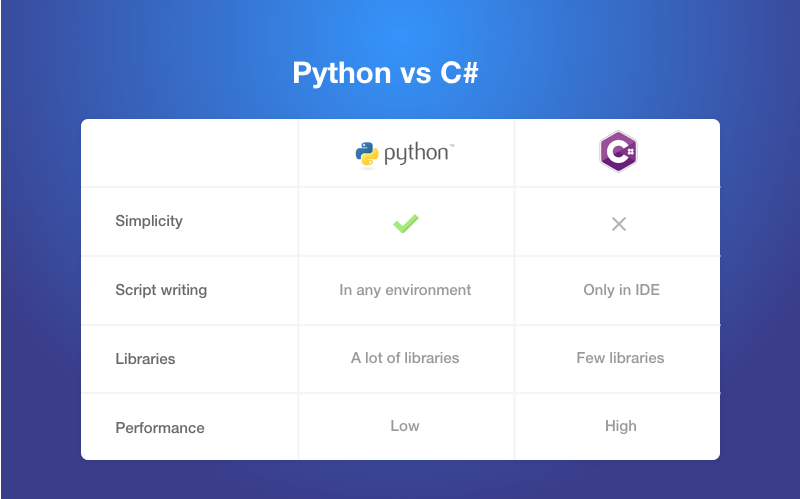
Choosing between **Python or PHP for web applications** pay attention to these characteristics:



**Python vs Java**

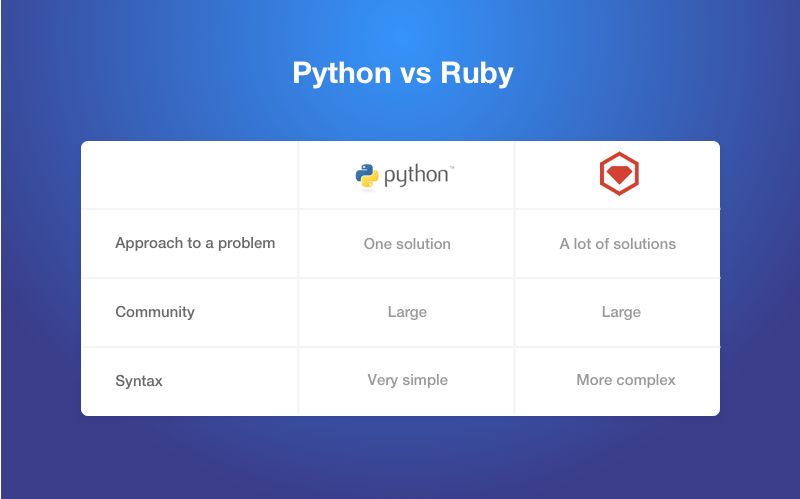


**Python vs C#**



**Difference between Ruby and Python**

In terms of the first language, **Ruby and Python** are the most popular ones. Ruby is extremely popular technology for building websites. Among the most famous are Twitter (the early version), Basecamp, Github, Airbnb, Slideshare and Groupon.



**Installing and Running Python in Windows**

1. Go to Download Python page on the official site and click **Download Python 3.7** (You may see different version name).
2. When the download is completed, double-click the file and follow the instructions to install it.

When Python is installed, a program called IDLE is also installed along with it. It provides graphical user interface to work with Python.

1. Open IDLE, copy the following code below and press enter.
2. print("Hello, World!")
3. To create a file in IDLE, go to **File > New Window** (Shortcut: **Ctrl+N**).
4. Write Python code (you can copy the code below for now) and save (Shortcut: **Ctrl+S**) with **.py** file extension like: hello.py or your-first-program.py

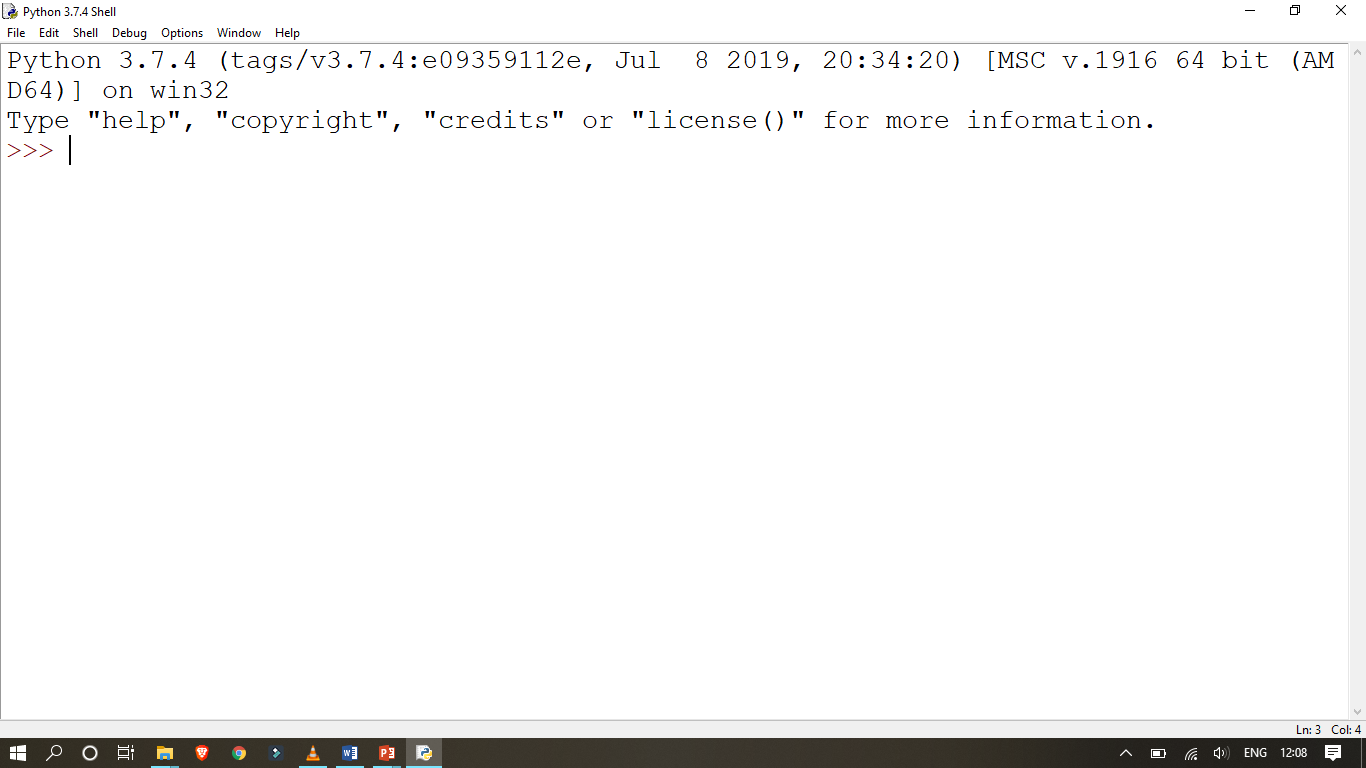
print("Hello, World!")

1. Go to **Run > Run module** (Shortcut: **F5**) and you can see the output. Congratulations, you've successfully run your first Python program.

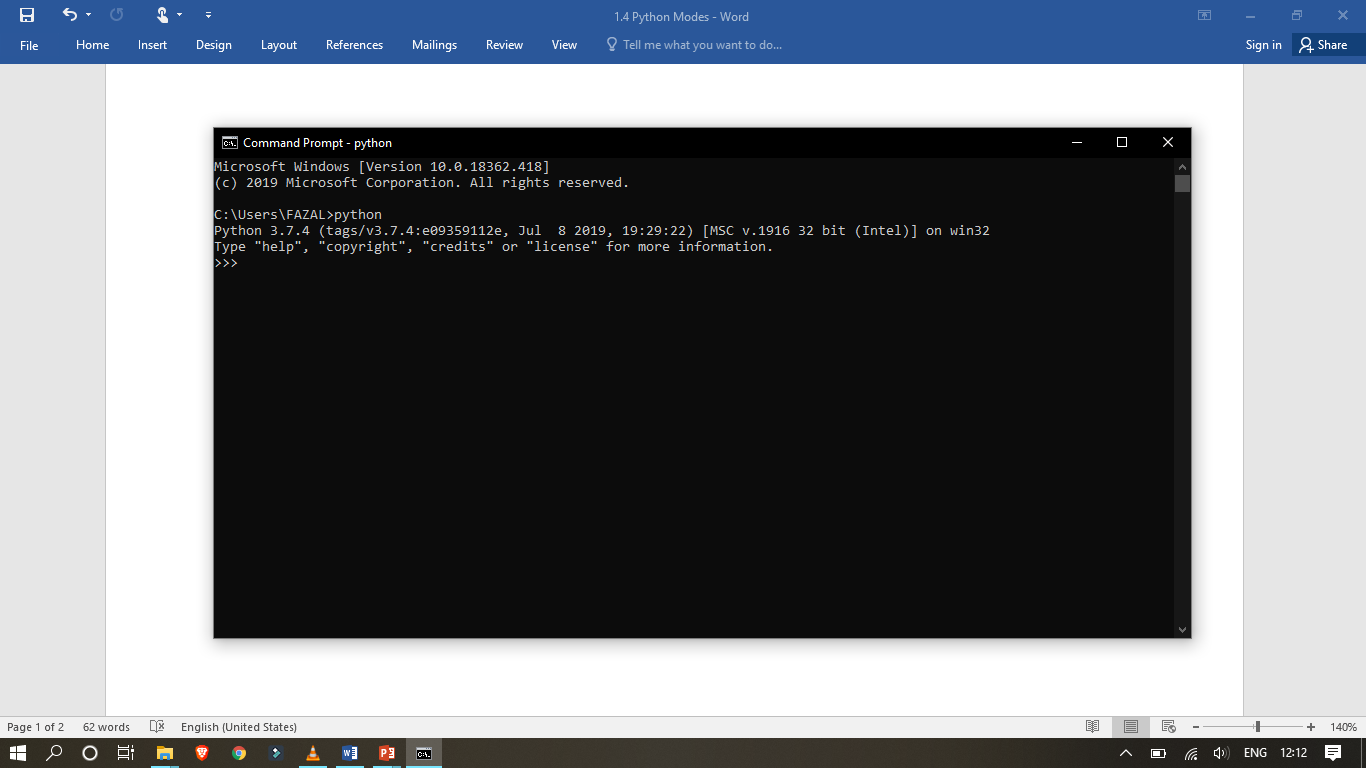
**PYTHON HAS TWO BASIC MODES:**

**Interactive mode:** is a command line shell which gives immediate output for each statement, while running previously statements in **active** memory.

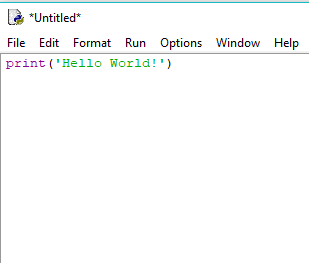
This mode is also referred as REPL (Read Evaluate Print Loop)



**We can start an interactive session from Command Prompt Directly.**



**Normal mode:** is where the scripted python file (.py) run in the Python interpreter.



**PYTHON PROGRAM TO ADD TWO NUMBERS**

# This program adds two numbers

num1 = 1.5

num2 = 6.3

# Add two numbers

sum = float(num1) + float(num2)

# Display the sum

print('The sum of {0} and {1} is {2}'.format(num1, num2, sum))

**ADD TWO NUMBERS PROVIDED BY THE USER**

# Store input numbers

num1 = input('Enter first number: ')

num2 = input('Enter second number: ')

# Add two numbers

sum = float(num1) + float(num2)

# Display the sum

print('The sum of {0} and {1} is {2}'.format(num1, num2, sum))

We use the built-in function input() to take the input.

input() returns a string, so we convert it into number using the float() function.

**Python Quick start**

Python is an interpreted programming language, this means that as a developer you write Python (.py) files in a text editor and then put those files into the python interpreter to be executed.

The way to run a python file is like this on the command line:

C:\Users\*Your Name*>python helloworld.py

Where "helloworld.py" is the name of your python file.

Let's write our first Python file, called helloworld.py, which can be done in any text editor.

helloworld.py

print("Hello, World!")

Simple as that. Save your file. Open your command line, navigate to the directory where you saved your file, and run:

C:\Users\*Your Name*>python helloworld.py

The output should read:

Hello, World!

Congratulations, you have written and executed your first Python program.

**The Python Command Line**

To test a short amount of code in python sometimes it is quickest and easiest not to write the code in a file. This is made possible because Python can be run as a command line itself.

Type the following on the Windows, Mac or Linux command line:

C:\Users\*Your Name*>python

From there you can write any python, including our hello world example from earlier in the tutorial:

C:\Users\*YourName*>python  
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] onwin 32 Type "help", "copyright", "credits" or "license" for more information.

>>> print("Hello, World!")

Which will write "Hello, World!" in the command line:

Whenever you are done in the python command line, you can simply type the following to quit the python command line interface:

exit()

**Python - GUI Programming (Tkinter)**

Python provides various options for developing graphical user interfaces (GUIs). Most important are listed below.

* **Tkinter** − Tkinter is the Python interface to the Tk GUI toolkit installed with Python.
* **wxPython** − This is an open-source Python interface for wxWindows
* **JPython** − JPython is a Python port for Java which gives Python scripts seamless access to Java class libraries on the local machine

**Tkinter Programming**

Tkinter is the standard GUI library for Python.

Python when combined with Tkinter provides a fast and easy way to create GUI applications.

Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.

Creating a GUI application using Tkinter, we need to do is perform the following steps

* Import the *tkinter* module.
* Create the GUI application main window.
* Add one or more widgets to the GUI application.
* Enter the main event loop to take action against each event triggered by the user.

**Tkinter Widgets**

tkinter provides various controls, such as buttons, labels and text boxes used in a GUI application. These controls are commonly called widgets.

widgets in Tkinter.

|  |  |
| --- | --- |
| **Sr.No.** | **Operator & Description** |
| **1** | Button  The Button widget is used to perform some action when clicked. |
| **2** | Canvas  The Canvas widget is used to draw shapes, such as lines, ovals, polygons and rectangles, in your application. |
| **3** | Checkbutton  The Checkbutton widget is used to display a number of options as checkboxes. The user can select multiple options at a time. |
| **4** | Entry  The Entry widget is used to display a single-line text field for accepting values from a user. |
| **5** | Frame  The Frame widget is used as a container widget to organize other widgets. |
| **6** | Label  The Label widget is used to provide a single-line caption for other widgets. It can also contain images. |
| **7** | Listbox  The Listbox widget is used to provide a list of options to a user. |
| **8** | Menubutton  The Menubutton widget is used to display menus in your application. |
| **9** | Menu  The Menu widget is used to provide various commands to a user. These commands are contained inside Menubutton. |
| **10** | Message  The Message widget is used to display multiline text fields for accepting values from a user. |
| **11** | Radiobutton  The Radiobutton widget is used to display several options as radio buttons. The user can select only one option at a time. |
| **12** | Scale  The Scale widget is used to provide a slider widget. |
| **13** | Scrollbar  The Scrollbar widget is used to add scrolling capability to various widgets, such as list boxes. |
| **14** | Text  The Text widget is used to display text in multiple lines. |
| **15** | Toplevel  The Toplevel widget is used to provide a separate window container. |
| **16** | Spinbox  The Spinbox widget is a variant of the standard Tkinter Entry widget, which can be used to select from a fixed number of values. |
| **17** | PanedWindow  A PanedWindow is a container widget that may contain any number of panes, arranged horizontally or vertically. |
| **18** | LabelFrame  A labelframe is a simple container widget. Its primary purpose is to act as a spacer or container for complex window layouts. |
| **19** | MessageBox  This module is used to display message boxes in your applications. |

Let us study these widgets in detail −

**Standard attributes**

Let us look at how some of their common attributes. such as sizes, colors and fonts are specified.

* Dimensions
* Colors
* Fonts
* Anchors
* Relief styles
* Bitmaps
* Cursors

**Geometry Management**

All Tkinter widgets have access to specific geometry management methods, which have the purpose of organizing widgets throughout the parent widget area.

Tkinter contains the following geometry manager classes: pack, grid, and place.

* **The *pack()* Method** − organizes widgets in blocks before placing them in the parent widget.
* **The *grid()* Method** − organizes widgets in a table-like structure in the parent widget.
* **The *place()* Method** − organizes widgets by placing them in a specific position in the parent widget.

**Why Django?**

Django is a Web framework written in Python.

A Web framework is a software that supports the development of dynamic Web sites, applications, and services.

It provides a set of tools and functionalities that solves many common problems associated with Web development, such as security features, database access, sessions, template processing, URL routing, internationalization, localization, and much more.

Using a Web framework, such as Django, enables us to develop secure and reliable Web applications very quickly in a standardized way.

The development of Django is supported by the [Django Software Foundation](https://www.djangoproject.com/foundation/), and it’s sponsored by companies like JetBrains and Instagram.

##### Who’s Using Django?

It’s good to know who is using Django out there, so to have an idea what you can do with it. Among the biggest Web sites using Django we have: [Instagram](https://instagram.com/), [Disqus](https://disqus.com/), [Mozilla](https://www.mozilla.org/), [Bitbucket](https://bitbucket.org/), [Last.fm](https://www.last.fm/), [National Geographic](http://www.nationalgeographic.com/).

#### Installation

The first thing we need to do is install some programs on our machine so to be able to start playing with Django. The basic setup consists of installing

* **Python**
* **Virtualenv**
* **Django**

Using virtual environments is not mandatory, but it’s highly recommended.

##### Installing Virtualenv

we are going to use **pip**, a tool to manage and install Python packages, to install **virtualenv**.

In the Command Prompt, execute the command below:

pip install virtualenv

From now on, everything we install, including Django itself, will be installed inside a Virtual Environment.

mkdir myproject

cd myproject

This folder is the higher level directory that will store all the files and things related to our Django project, including its virtual environment.

let’s start by creating our very first virtual environment and installing Django.

Inside the **myproj** folder:

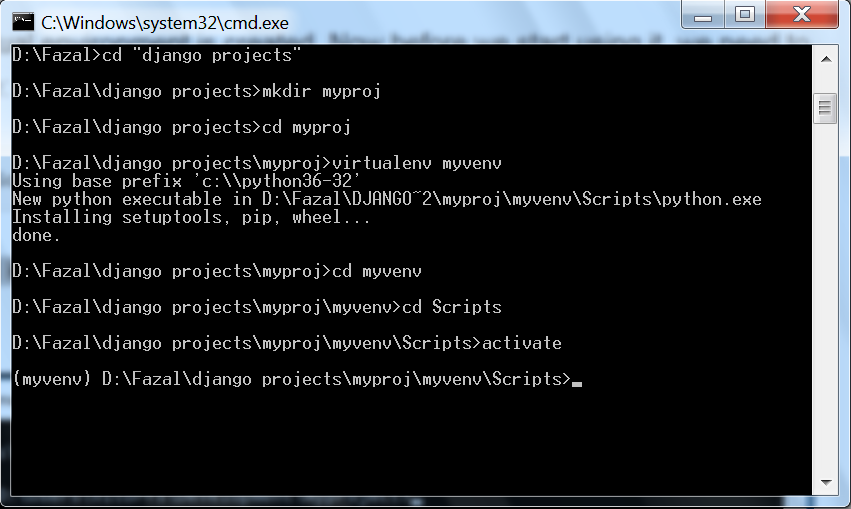
virtualenv myvenv

Our virtual environment is created.

Now before we start using it, we need to activate:

myvenv\Scripts\activate

You will know it worked if you see **(venv)** in front of the command line, like this:



to deactivate the **venv** run the command below:

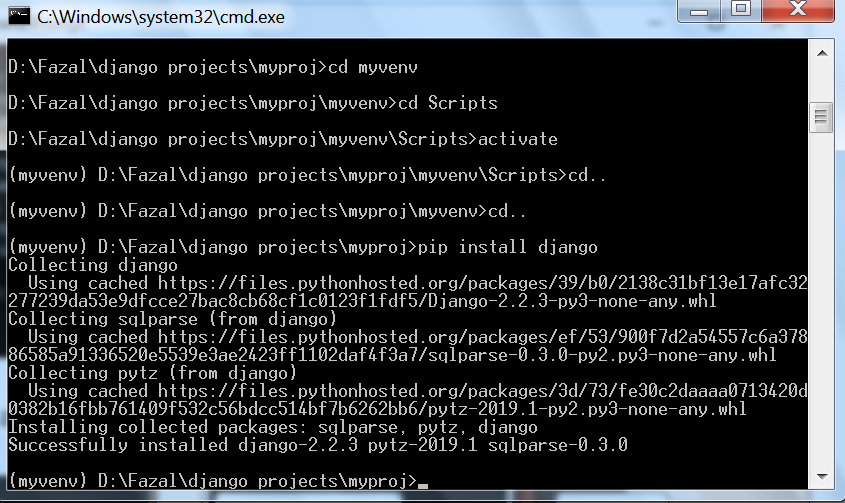
venv\Scripts\deactivate.bat

But let’s keep it activated for the next steps.

##### Installing Django

Now that we have the **venv** activated, run the following command to install Django:

pip install django



#### Starting a New Project

To start a new Django project, run the command below:

django-admin startproject myproject

The command-line utility **django-admin** is automatically installed with Django.

After we run the command above, it will generate the base folder structure for a Django project.

Our initial project structure is composed of five files:

* **manage.py**: a shortcut to use the **django-admin** command-line utility. It’s used to run management commands related to our project.

We will use it to run the development server, run tests, create migrations and much more.

* **\_\_init\_\_.py**: this empty file tells Python that this folder is a Python package.
* **settings.py**: this file contains all the project’s configuration.
* **urls.py**: this file is responsible for mapping the routes and paths in our project.

For example, if you want to show something in the URL /about/, you have to map it here first.

* **wsgi.py**: this file is a simple gateway interface used for deployment.

You don’t have to bother about it. Just let it be for now.

Django comes with a simple web server installed.

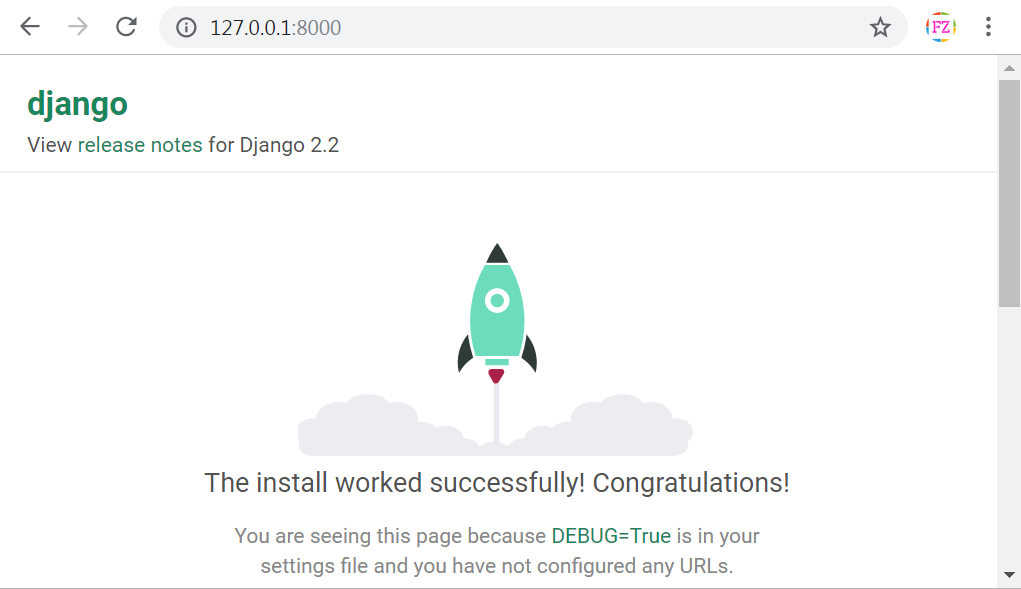
It’s very convenient during the development, so we don’t have to install anything else to run the project locally.

We can test it by executing the command:

python manage.py runserver

For now, you can ignore the migration errors; we will get to that later.

Now open the following URL in a Web browser: **http://127.0.0.1:8000** and you should see the following page:



Hit CTRL + BREAK to stop the development server.

#### Django Apps

In the Django philosophy we have two important concepts:

* **app**: is a Web application that does something.

An app usually is composed of a set of models (database tables), views, templates, tests.

* **project**: is a collection of configurations and apps.

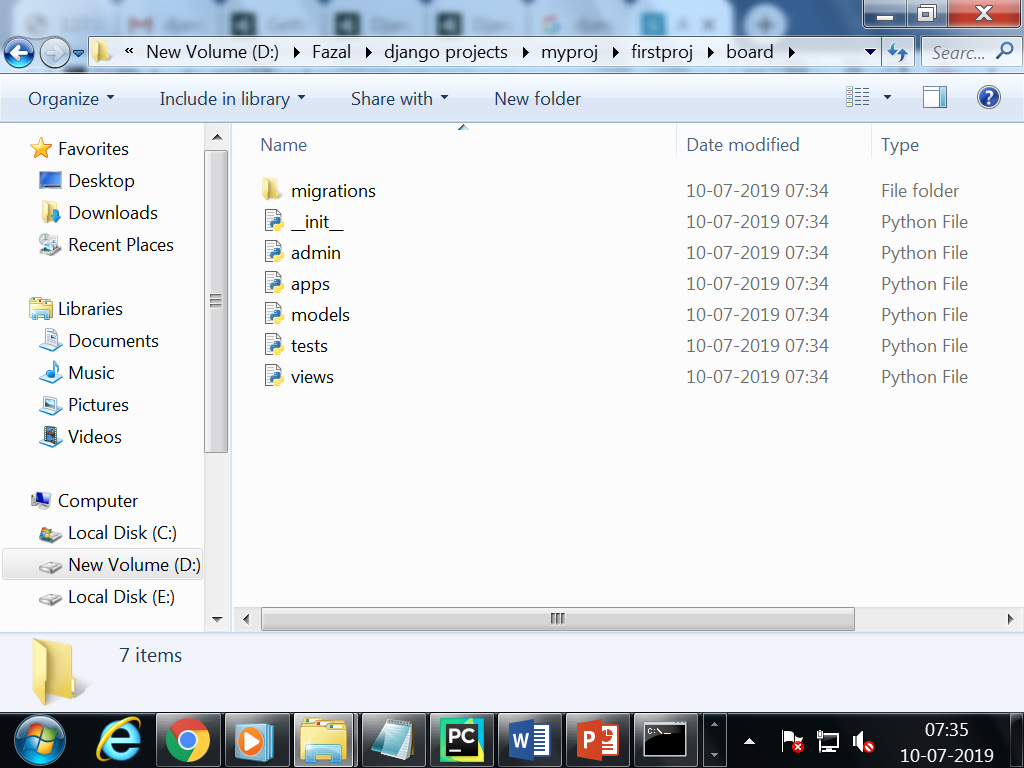
One project can be composed of multiple apps, or a single app.

It’s important to note that you can’t run a Django **app** without a **project**. Simple websites like a blog can be written entirely inside a single app, which could be named **blog** or **weblog** for example.

let’s create a simple Web Forum or Discussion Board. To create our first app, go to the directory where the **manage.py** file is and executes the following command:

django-admin startapp boards

Notice that we used the command **startapp** this time.



So, let’s first explore what each file does:

* **migrations/**: here Django store some files to keep track of the changes you create in the **models.py** file, so to keep the database and the **models.py** synchronized.
* **admin.py**: this is a configuration file for a built-in Django app called **Django Admin**.
* **apps.py**: this is a configuration file of the app itself.
* **models.py**: here is where we define the entities of our Web application. The models are translated automatically by Django into database tables.
* **tests.py**: this file is used to write unit tests for the app.
* **views.py**: this is the file where we handle the request/response cycle of our Web application.

Now that we created our first app, let’s configure our project to *use* it.

To do that, open the **settings.py** and try to find the INSTALLED\_APPS variable:

**settings.py**

INSTALLED\_APPS **=** [

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

'django.contrib.staticfiles'**,**

]

As you can see, Django already come with 6 built-in apps installed. They offer common functionalities that most Web applications need, like authentication, sessions, static files management (images, javascripts, css, etc.) and so on.

#### Hello, World!

Let’s write our first **view**. We will explore it in great detail in the next tutorial. But for now, let’s just experiment how it looks like to create a new page with Django.

Open the **views.py** file inside the **boards** app, and add the following code:

**views.py**

from django.http import HttpResponse

**def** **home**(request):

**return** HttpResponse('Hello, World!')

Views are Python functions that receive an HttpRequest object and returns an HttpResponse object. Receive a request as a parameter and returns a response as a result. That’s the flow you have to keep in mind!

So, here we defined a simple view called **home** which simply returns a message saying **Hello, World!**.

Now we have to tell Django when to serve this view. It’s done inside the **urls.py** file:

**urls.py**

from django.conf.urls import url

from django.contrib import admin

from boards import views

urlpatterns **=** [

url(r'^$', views**.**home, name**=**'home'),

url(r'^admin/', admin**.**site**.**urls),

]

If you compare the snippet above with your **urls.py** file, you will notice I added the following new line: url(r'^$', views.home, name='home') and imported the **views** module from our app **boards** using from boards import views.

As I mentioned before, we will explore those concepts in great detail later on.

But for now, Django works with **regex** to match the requested URL. For our **home** view, I’m using the ^$ regex, which will match an empty path, which is the homepage (this url: **http://127.0.0.1:8000**). If I wanted to match the URL **http://127.0.0.1:8000/homepage/**, my url would be:

url(r'^homepage/$', views.home, name='home').

Let’s see what happen:

python manage.py runserver

In a Web browser, open the http://127.0.0.1:8000 URL:

**9.2SOURCE CODE :**

**9.2.0MAIN.PY:**

import matplotlib.pyplot as plt

from tkinter import messagebox

from tkinter import \*

from tkinter.filedialog import askopenfilename

from tkinter import simpledialog

import tkinter

from tkinter import filedialog

import os

import re

import numpy as np

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn import metrics

from sklearn.tree import DecisionTreeClassifier

from sklearn.tree import export\_text

from sklearn.feature\_selection import RFE

root = tkinter.Tk()

root.title("A Decision Tree based Recommendation System for Tourists")

root.geometry("1200x850")

global filename

feature\_cols = ['userid','art\_galleries','dance\_clubs','juice\_bars','restaurants','museums','resorts','parks\_picnic\_spots','beaches','theaters','religious\_institutions']

global clf

global rfe

global X\_train

global y\_train

global fit

def upload():

    global filename

    filename = filedialog.askopenfilename(initialdir="dataset")

    pathlabel.config(text=filename)

    text.delete('1.0', END)

    with open(filename, "r") as file:

      for line in file:

         line = line.strip('\n')

         text.insert(END,line+"\n")

def featureSelection():

    global clf

    global rfe

    global fit

    global X\_train

    global y\_train

    dataset = pd.read\_csv(filename)

    dataset.head()

    y = dataset['location']

    X = dataset.drop(['location'], axis = 1)

    X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.1, random\_state=0)

    clf = DecisionTreeClassifier()

    rfe = RFE(clf, 3)

    fit = rfe.fit(X\_train,y\_train)

    text.delete('1.0', END)

    text.insert(END,"Total number of features : "+str(len(feature\_cols))+"\n")

    text.insert(END,"Selected number of features : "+str(fit.n\_features\_)+"\n")

    text.insert(END,"Selected number of features : "+str(fit.support\_)+"\n")

def decisionTree():

    global clf

    global X\_train

    global y\_train

    clf.fit(X\_train,y\_train)

    text.delete('1.0', END)

    r = export\_text(clf, feature\_names=feature\_cols)

    text.insert(END,"Selected number of features : "+str(r)+"\n")

def predict():

    global clf

    testname = filedialog.askopenfilename(initialdir="dataset")

    text.delete('1.0', END)

    with open(testname, "r") as file:

      for line in file:

         line = line.strip('\n')

         text.insert(END,line+"\n")

    test = pd.read\_csv(testname)

    y\_pred = clf.predict(test)

    text.insert(END,"\nPredicted Location : "+str(y\_pred)+"\n")

def graph():

    global fit

    height = [len(feature\_cols), fit.n\_features\_]

    bars = ('Total Features', 'Selected Features')

    y\_pos = np.arange(len(bars))

    plt.bar(y\_pos, height)

    plt.xticks(y\_pos, bars)

    plt.show()

font = ('times', 18, 'bold')

title = Label(root, text='A Decision Tree based Recommendation System for Tourists')

title.config(bg='wheat', fg='red')

title.config(font=font)

title.config(height=3, width=80)

title.place(x=5,y=5)

font1 = ('times', 14, 'bold')

upload = Button(root, text="Upload Tourist Dataset", command=upload)

upload.place(x=50,y=100)

upload.config(font=font1)

pathlabel = Label(root)

pathlabel.config(bg='blue', fg='white')

pathlabel.config(font=font1)

pathlabel.place(x=300,y=100)

normal = Button(root, text="Run Preprocess & Features Selection Algorithm", command=featureSelection)

normal.place(x=50,y=150)

normal.config(font=font1)

decisionbutton = Button(root, text="Run C4.5 Decision Tree", command=decisionTree)

decisionbutton.place(x=50,y=200)

decisionbutton.config(font=font1)

predictbutton = Button(root, text="Tourist Recommendation", command=predict)

predictbutton.place(x=50,y=250)

predictbutton.config(font=font1)

rungraph = Button(root, text="Features Selection Graph", command=graph)

rungraph.place(x=50,y=300)

rungraph.config(font=font1)

text=Text(root,height=25,width=80)

scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=550,y=100)

text.config(font=font1)

root.mainloop()

**9.2.1TEST.PY:**

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn import metrics

from sklearn.tree import DecisionTreeClassifier

from sklearn.tree import export\_text

from sklearn.feature\_selection import RFE

dataset = pd.read\_csv('dataset.txt')

dataset.head()

test = pd.read\_csv('test.txt')

feature\_cols = ['userid','art\_galleries','dance\_clubs','juice\_bars','restaurants','museums','resorts','parks\_picnic\_spots','beaches','theaters','religious\_institutions']

y = dataset['location']

X = dataset.drop(['location'], axis = 1)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.1, random\_state=0)

clf = DecisionTreeClassifier()

rfe = RFE(clf, 3)

# Train Decision Tree Classifer

fit = rfe.fit(X\_train,y\_train)

#Predict the response for test dataset

y\_pred = rfe.predict(test)

print("predicted : "+str(y\_pred))

print("Num Features: %d" % fit.n\_features\_)

print("Selected Features: %s" % fit.support\_)

print("Feature Ranking: %s" % fit.ranking\_)

#r = export\_text(clf, feature\_names=feature\_cols)

#print(r)

**CHAPTER-10**

**10.1 SYSTEM TEST**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

#### TYPES OF TESTS:

**Unit testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately

### **Functional test**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted. Invalid Input : identified classes of invalid input must be rejected. Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised. Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

#### System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**OUTPUT SCREENS**

**A Decision Tree based Recommendation System for Tourists**

In this paper author is implementing C4.5 decision tree algorithm with MRMR features selection to recommend travel areas to tourist by using dataset from past tourist experiences. All existing algorithms such as collaborative or content filtering algorithms uses current user past experience data to recommend him new locations. These algorithms will not work if this current user has no past experiences data.

To overcome from above problem author is asking to use C4.5 decision tree algorithms which take experiences of previous users and then build a model and if new user enter his requirements then decision tree will predict best location based on his given input. Decision tree don’t need new users past experience data.

To implement decision tree model we need to have dataset and this dataset sometime will have empty or garbage values and this values will put bad effect on decision tree model so we can remove such empty or garbage values by applying pre-process techniques.

Sometime to predict or build model no need to use all columns (attributes) values from dataset and this unnecessary attributes can be remove by apply features selection algorithms and here we are using MRMR features selection algorithms to remove unnecessary attributes to reduce execution time of building model and to increase system accuracy.

Below are the dataset columns or attributes taken from previous users to build model.

This data set is populated by crawling TripAdvisor.com. Reviews on destinations in 10 categories mentioned across East Europe are considered. Each traveller rating is mapped as Excellent (4), Very Good (3), Average (2), Poor (1), and Terrible (0) and average rating is used against each category per user.

Dataset columns and values

userid,art\_galleries,dance\_clubs,juice\_bars,restaurants,museums,resorts,parks\_picnic\_spots,beaches,theaters,religious\_institutions,location

Above are the column names and below are the column values

User 1,0.93,1.8,2.29,0.62,0.8,2.42,3.19,2.79,1.82,2.42,Amsterdam\_Heining\_2

User 2,1.02,2.2,2.66,0.64,1.42,3.18,3.21,2.63,1.86,2.32,Amsterdam\_Jachthaven\_ijbur

User 3,1.22,0.8,0.54,0.53,0.24,1.54,3.18,2.8,1.31,2.5,Amsterdam\_Bert\_Haanstra\_Kad

User 4,0.45,1.8,0.29,0.57,0.46,1.52,3.18,2.96,1.57,2.86,Amsterdam\_Ruigoord\_Ker

In above values first column is USER\_ID and second column is ART\_GALLERIES and third is DANCE CLUB etc and for each column user had given rating from 4 to 0 and 4 means Excellent service.

Now using above values we can build C4.5 decision tree and prediction will be done using below test values

'User 122',0.93,1.8,2.29,0.62,0.8,2.42,3.19,2.79,1.82,2.42,?

'User 222',1.02,2.2,2.66,0.64,1.42,3.18,3.21,2.63,1.86,2.32,?

'User 3222',1.22,0.8,0.54,0.53,0.24,1.54,3.18,2.8,1.31,2.5,?

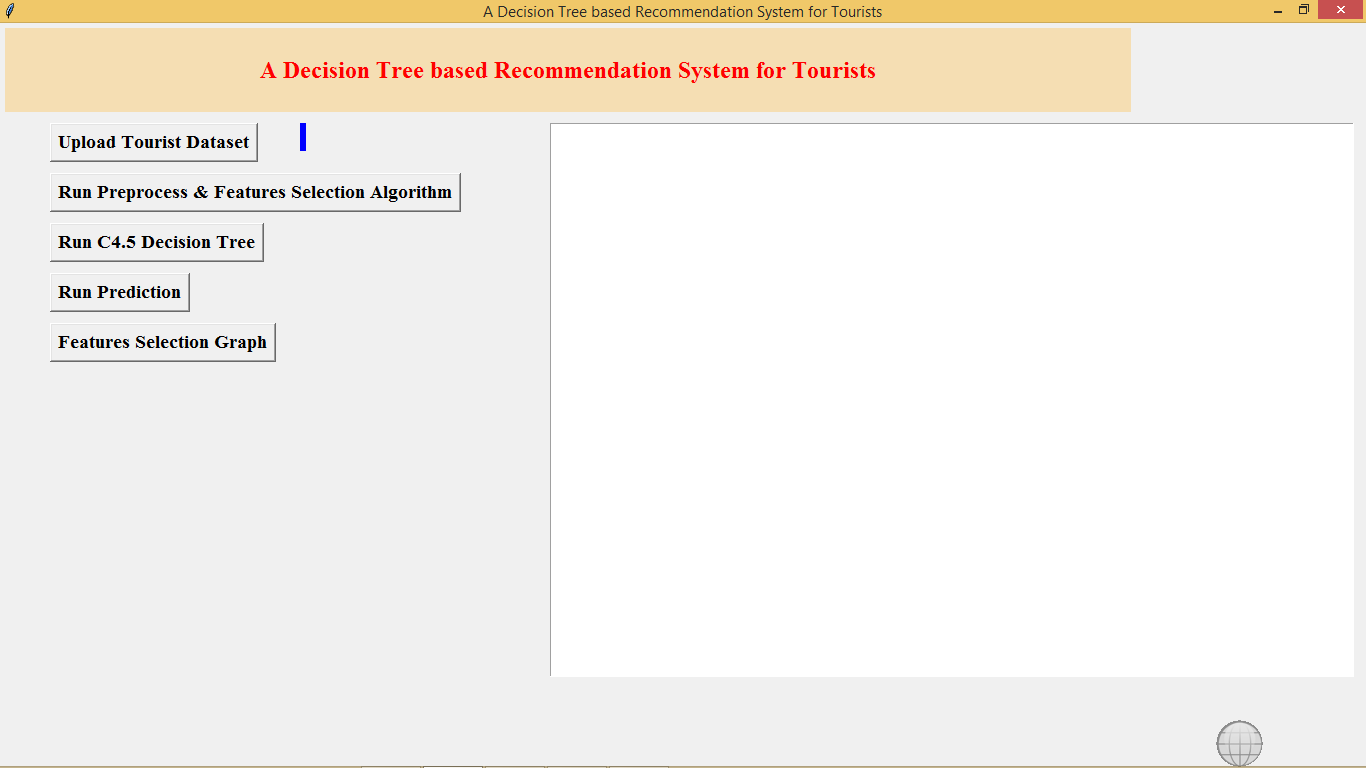
'User 4222',0.45,1.8,0.29,0.57,0.46,1.52,3.18,2.96,1.57,2.86,?

'User 522',0.51,1.2,1.18,0.57,1.54,2.02,3.18,2.78,1.18,2.54,?

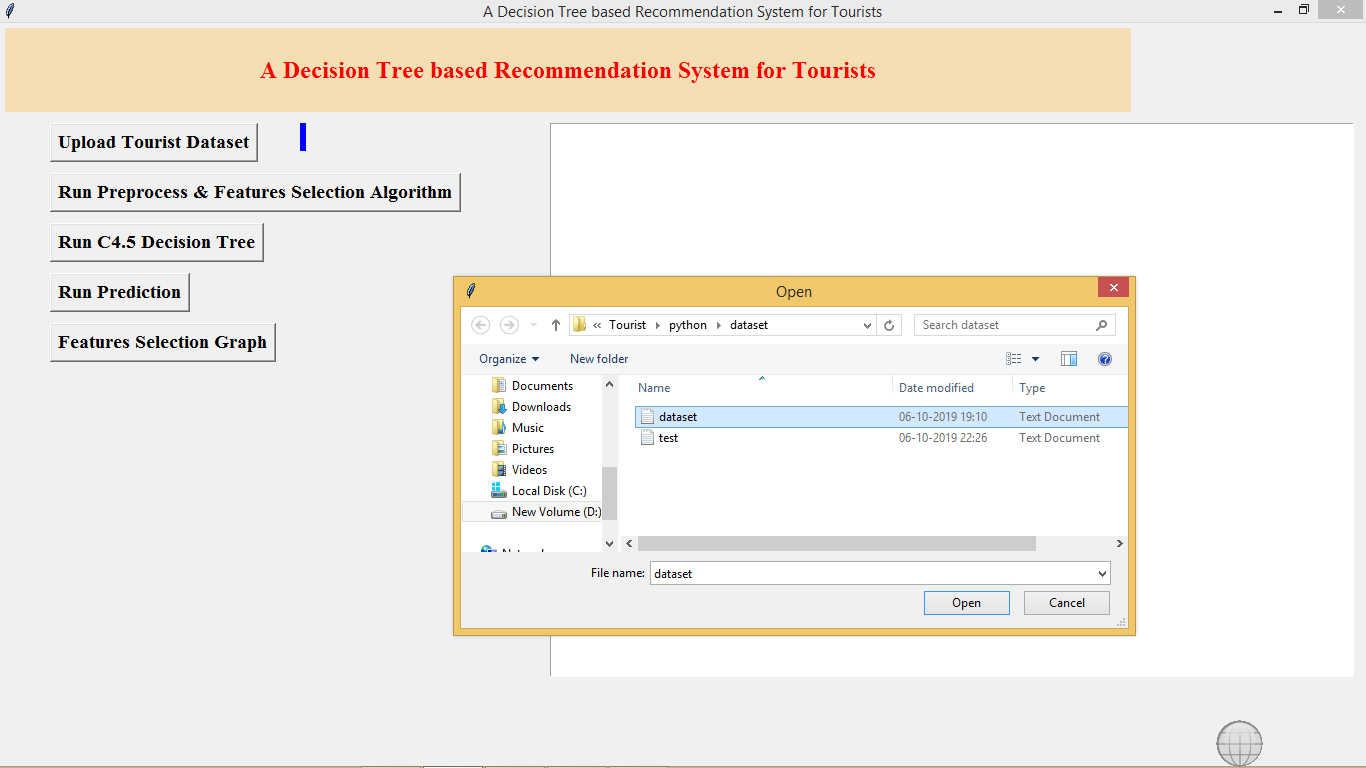
In above test values new user has given values to look for location which has above service rating but new user don’t know which location provides such services so he will put question mark and when we upload above test values to decision tree then it will take decision and predict best location and inform to user.

Screen shots

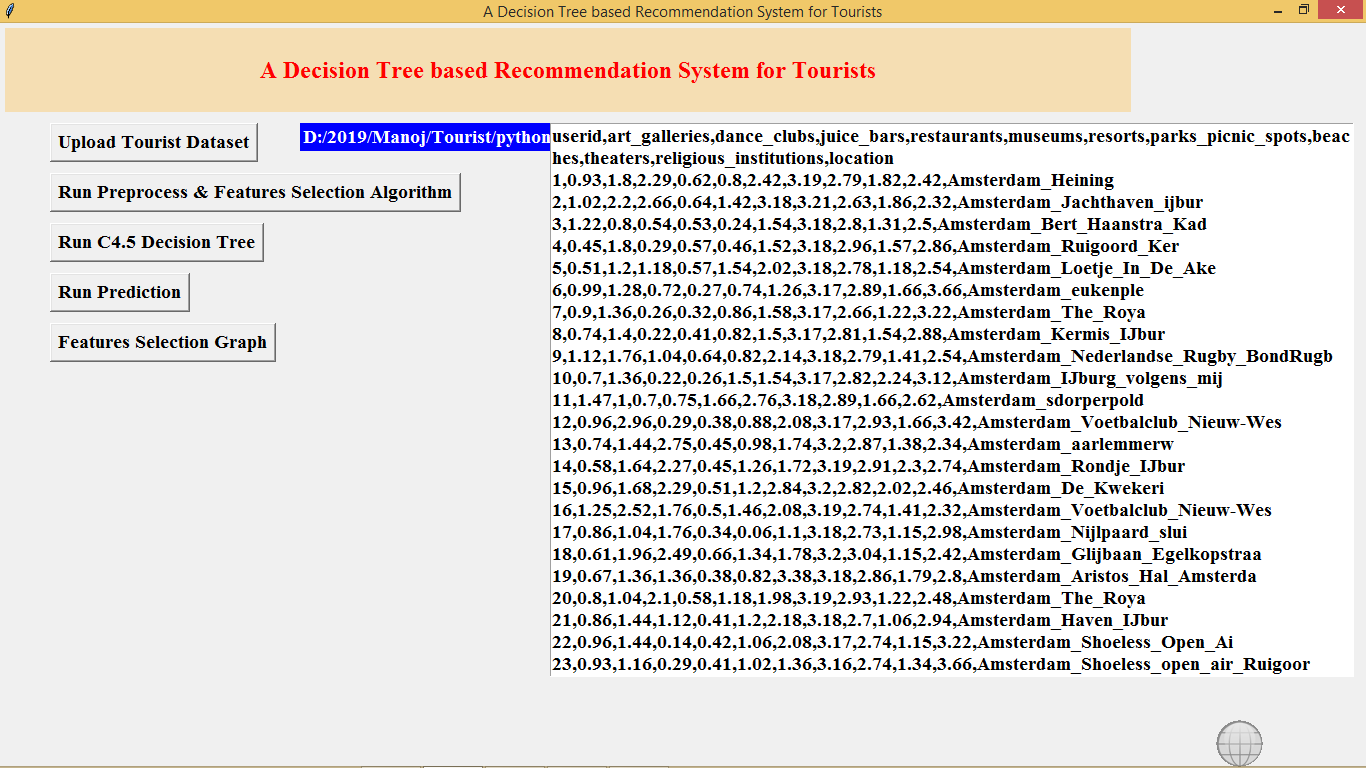
Double click on ‘run.bat’ file to get below screen



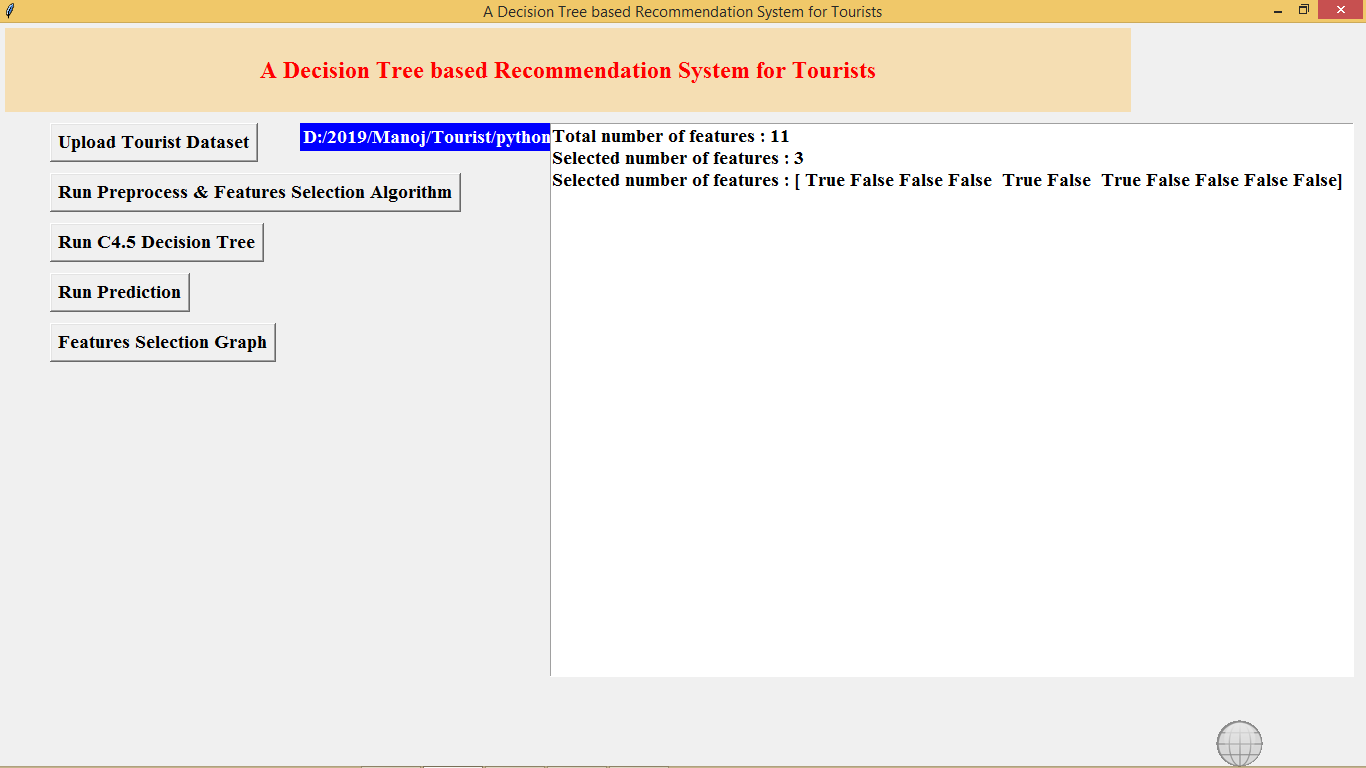
In above screen click on ‘Upload Tourist Dataset’ button and upload dataset file



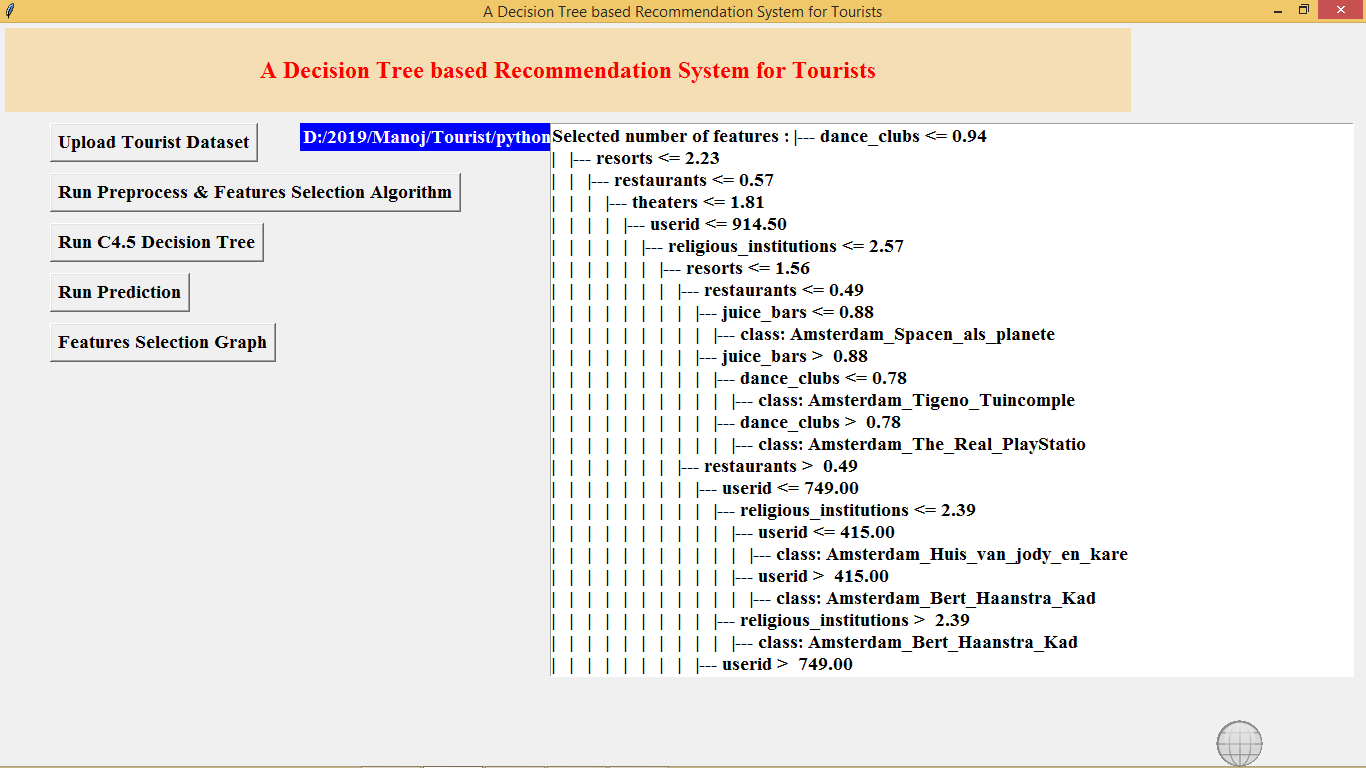
After file upload will get below screen with all dataset details



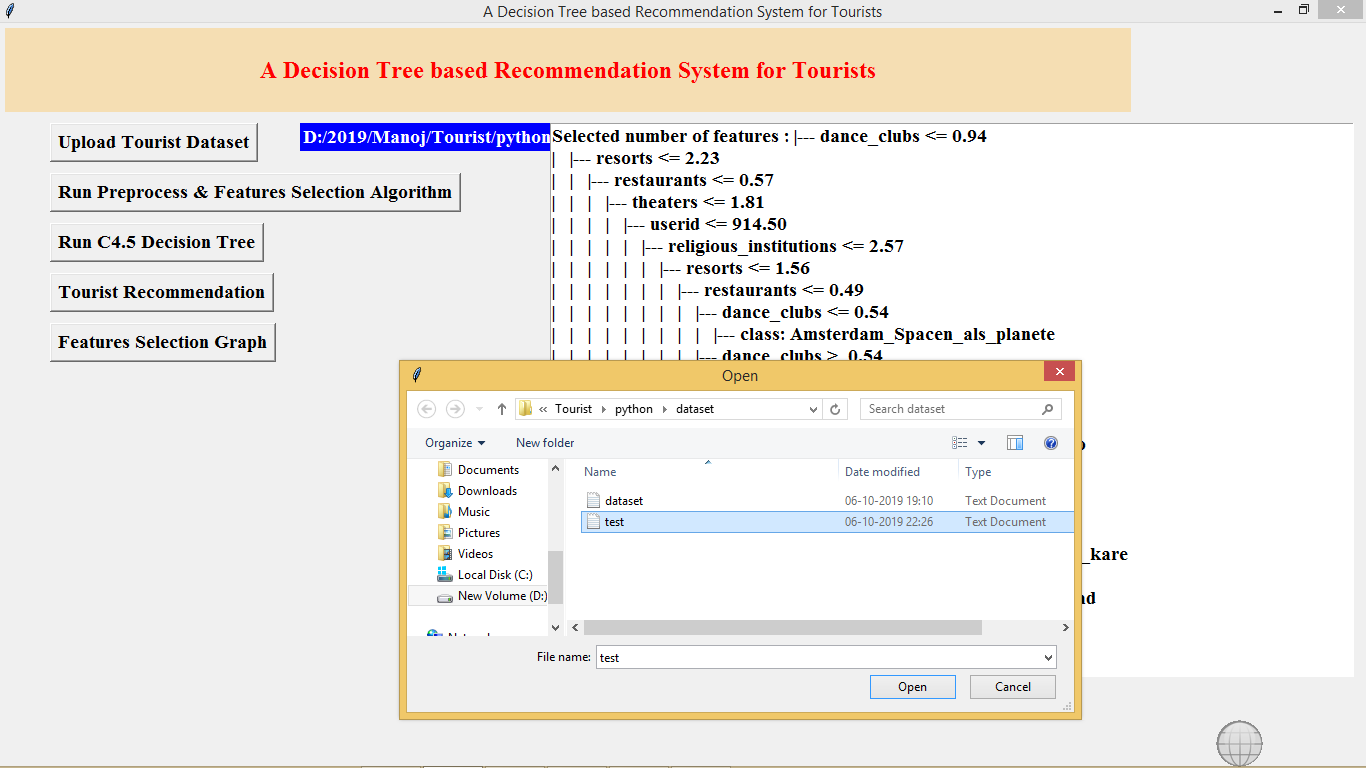
In above screen all users past experience dataset loaded and total 12 attributes are there in the dataset. Now click on ‘Run Preprocess & Feature Selection Algorithm’ button to remove empty values and reduce attributes size.



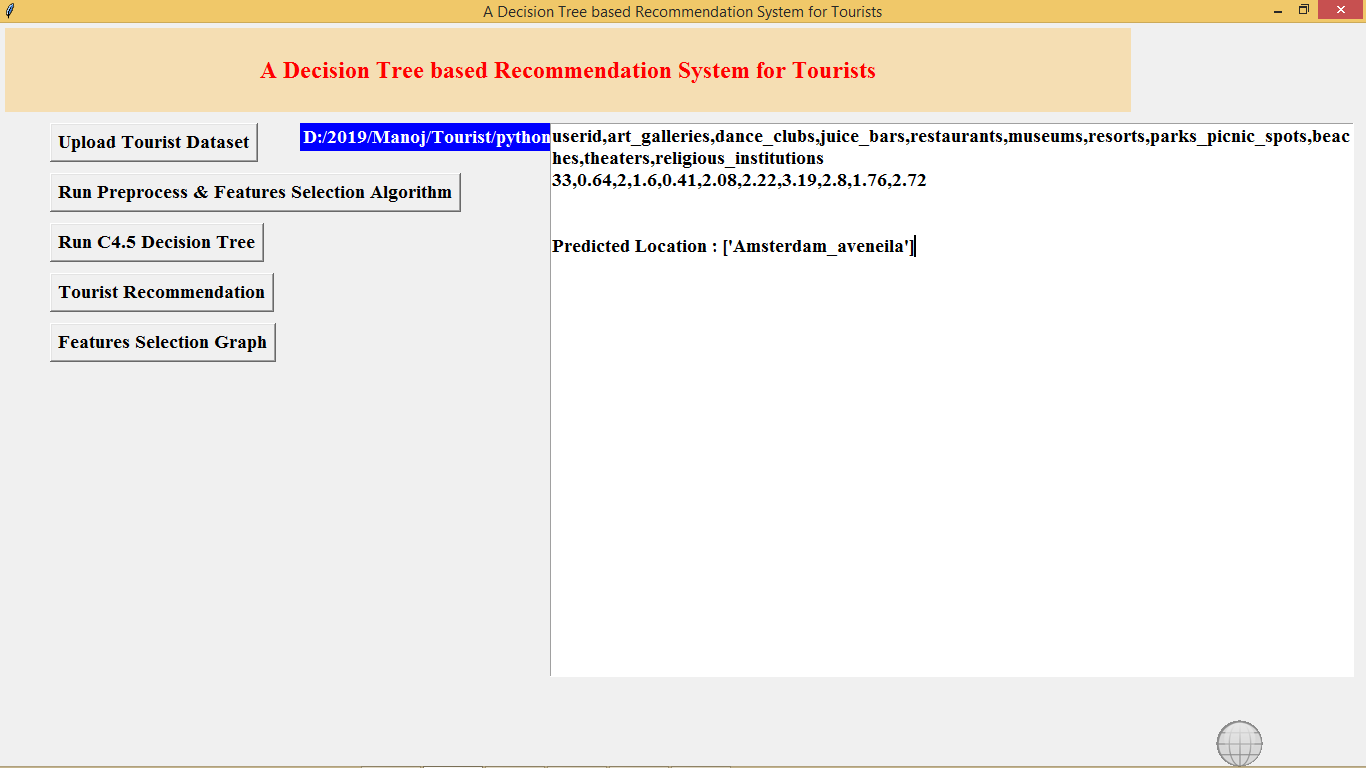
In above screen after applying MRMR features size reduces to 3 and only those attributes will be used whose column is TRUE and FALSE column will be ignore. Now click on ‘Generate C4.5 Decision Tree Model’ to build model



In above screen we can see using IF and ELSE statement decision tree has generated model. If > it will choose some decision if < it will choose some other decision. Now click on ‘Tourist Recommendation’ button to upload test file with no location name and application will predict it

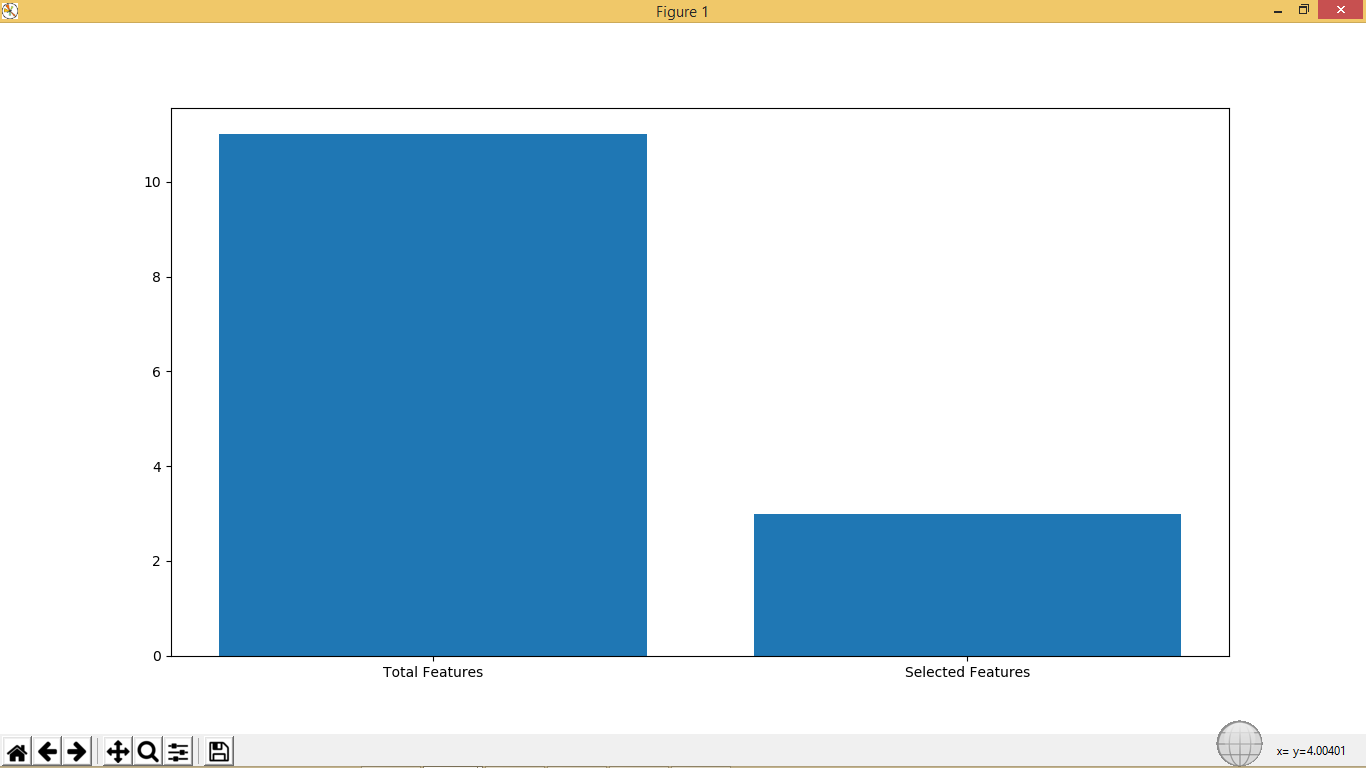


In above screen i am uploading test file now click open to get predicted or recommended location. In test file location name is not there application will give



In above screen after uploading test data we can see all values are there in test data but it not has location name and base on test values application predicted or recommend location name.

Now click on Features Selection Graph button to get below graph



In above graph x-axis contains total features and MRMR selected features and y-axis represents count of features and in above graph we can see after applying MRMR technique features size reduces to 3.

**CHAPTER-11**

**CONCLUSION**

A decision tree based tourist www.jespublication.com recommendation system has been presented in attempt of solving the current challenge of the destination TRS. The data set has been decomposed into two sub data sets using relevant tourism domain knowledge. This was done to increase classification accuracy rate and to reduce the complexity of the decision tree. The optimal decision trees from NMIFS with the highest accuracy rate and simplicity (i.e. less number of leaf and tree size) have been constructed for destination choice. The decision rules from decision trees were extracted. It can be seen that NMIFS is the optimum method because it uses fewer number of feature than MRMR for both of the data sets. Finally, the experimental results confirm applicable of the proposed a TRS. The proposed TRS satisfies the tourists’ requirements who plan to visit or during their visit the city of Chiang Mai.

**CHAPTER-12**

**REFERENCES**

**[1].**J.Chiverton, “Helmet Presence Classification with Motorcycle Detection And Tracking”,IET Intelligent Transport Systems,Vol. 6, Issue 3, pp. 259–269, March 2012.

**[2].** Rattapoom Waranusast, Nannaphat Bundon, Vasan Timtong and Chainarong Tangnoi, “Machine Vision techniques for Motorcycle Safety Helmet Detection”, 28th International Conference on Image and Vision Computing New Zealand, pp 35-40, IVCNZ 2013.

**[3].** Romuere Silva, elson Aires, Thiago Santos, al f Abdala, Rodrigo Veras, Andr e Soares, “Automatic Detection Of Motorc clists without Helmet”, 2013 XXXIX Latin America Computing Conference (CLEI).IEEE,2013.

**[4].** Romuere Silva, “Helmet Detection on Motorcyclists Using Image Descriptors and Classifiers”, 27th SIBGRAPI Conference on Graphics, Patterns and Images.IEEE, 2014.

**[5].** Thepnimit Marayatr, Pinit Kumhom, “Motorc clist‟s Helmet Wearing Detection Using Image Processing”, Advanced Materials Research Vol 931- 932,pp. 588-592,May-2014.

**[6].** Amir Mukhtar, Tong Boon Tang, “Vision Based Motorc cle Detection using HOG features”, IEEE International Conference on Signal and Image Processing Applications (ICSIPA).IEEE, 2015.

**[7].** Abu H. M. Rubaiyat, Tanjin T. Toma, Masoumeh Kalantari-Khandani, “Automatic Detection of Helmet Uses www.jespublication.com for Construction Safet ”, IEEE/WIC/ACM International Conference on Web Intelligence Workshops(WIW).IEEE, 2016.

**[8].** .XINHUA JIANG “A Stud of Lowresolution Safety Helmet Image Recognition Combining Statistical Features with Artificial Neural Network”.ISSN: 1473-804x .

**[9].** .Kunal Dahiya, Dinesh Singh, C. rishna Mohan, “Automatic Detection of Bike-riders without Helmet using Surveillance Videos in Real-time”, International joint conference on neural network(IJCNN). IEEE, 2016.