Tourist site

An

Object-Oriented Programming Concepts through Java

Course Project Report in partial fulfillment of the degree

Bachelor of Technology

in

Computer Science &Engineering

By

2103A52055	**M.Dheeraj**
2103A52118	**P. Aashish**
2103A52128	**CH.Rohith**
2103A52136	**G. Suraj**

Under the Guidance of

Sravan sir

School of Computer Science and AI

Submitted to

School of CS & AI

SR UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the Object Oriented Programming through Java-Course Project Report entitled **“TOURIST SITE”** is a record of bonafide work carried out by the students M. Dheeraj, P.Aashish ,CH.Rohith, G.Suraj bearing RollNo(s) 203A52055, 203A52118,2103A52128,2103A52136 during the academic year 2022-2023in partial fulfillment of the award of the degree of ***Bachelor of Technology*** in **Computer Science & Engineering** by the SR University,Hasanparthy,Warangal.

Lab In-charge **Head of the Department**

ACKNOWLEDGEMENT

We owe an enormous debt of gratitude to our project guide SRAVAN SIR **,**as well as Head of the CSE Department **Dr . M. Sheshikala**, Associate Professorfor guiding us from the beginning through the end of the Capstone Project with theirintellectual advices and insightful suggestions. We truly value their consistent feedback on our progress, which always constructive and encouraging and ultimately drove us to the right direction .We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved Principal, **Dr.V.MAHESH,** for his continuous support and guidance to complete this project.

Finally, we express our thanks to all the teaching and non-teaching staff of the department for their suggestions and timely supp

TABLE OF CONTENTS

ABSTRACT

Our project is a Tourist interface. We have created a interface useful to the tourist to make their work easy and plan their schedules within their requirements.Basically, tourists plan their schedules in the weekends and in holidays.On this basis, we have created a interface consisting of weekend plans and long trip plans.Planning a trip is not easier in present situations based on the busy schedules. So, in our interface we will provide each element that is required to plan a trip based on their choice.Our interface consist of the details as days (weekend days)to be selected, number of persons to be visited and affordable budget .Based on this, tour packages and the total amount of the tour will be obtained as output based on the user's requirements of holidays .Through this interface ,tourists can schedule their tour packages in a user-friendly and in a cost-effective way.

OBJECTIVE

Our main aim is to focus on the people who want to travel to the different places. A tour package with a less number of days and budget-friendly package is not available. In our site, we suggest input to be given as budget, number of days, number of persons,location of their choice that would be affordable to the users. We will be providing the tour packages they can be visited based on the budget and the number of days using GUI(swings),if-else conditional statements, and exception handling concepts. So, we have created a user-friendly tourist interface to reach out people easily.

The primary objective of the Tours and Travel Management Java project is to develop a comprehensive and user-centric travel management system that simplifies and enhances the travel experience for both travelers and travel agents. The project seeks to achieve the following key objectives:

Efficient Booking and Reservation: Create a user-friendly platform that allows travelers to efficiently search for, book, and manage a wide range of travel services, including flights, accommodations, tours, and transportation .Personalized Itinerary Planning: Provide travelers with tools to plan and customize their travel itineraries, offering recommendations based on preferences, historical data, and real-time availability.
Secure Payment Processing: Implement secure and reliable payment processing systems to facilitate seamless financial transactions while protecting sensitive customer data.

INTRODUCTION

The Tours and Travel Management Java project aims to revolutionize the way travelers plan and organize their journeys, offering a comprehensive solution that streamlines the entire travel experience. In a world where efficiency, convenience, and personalized service are paramount, this project seeks to empower both travelers and travel agents with a robust Java-based application.

This project is driven by the recognition that modern travelers demand more than just a simple booking platform. They seek a holistic travel management system that provides end-to-end solutions, from itinerary planning and booking to communication, notifications, and customer relationship management. With a user-friendly Java Swing GUI, secure payment processing, data privacy measures, and integration with external services, the project endeavors to create a seamless and enjoyable travel experience.

By combining technical excellence with user-centric design, this project is poised to make a positive impact on the travel and tourism industry, enhancing the way travelers explore the world and travel agents manage bookings. This introduction sets the stage for a transformative journey into the future of travel management.

MODULES:

Module 1: User Interface (UI) Module

Description: This module is responsible for creating and managing the graphical user interface (GUI) of the application.
Components: It includes the creation of the main frame, panels, labels, input fields, buttons, and result display area.
Responsibilities: This module manages the layout, design, and interaction with the user, allowing them to input data, make selections, and view the results.
Module 2: Exception Handling Module

Description: This module deals with custom exception handling.
Components: It defines the custom exception class NoBudgetException.
Responsibilities: This module handles exceptions related to insufficient budget during the calculation process and provides appropriate error messages.
Module 3: Calculation Module

Description: This module is responsible for the core logic of the application. It calculates the trip details based on user input.
Components: It contains the calculate method, which processes user input, performs calculations, and displays the results.
Responsibilities: This module checks user input for validity, calculates the destination and total cost of the trip, and displays the results in the GUI.
Module 4: Main Module

Description: This module serves as the entry point for the application.
Components: It contains the main method, which initiates the application by creating an instance of the Tourist GUI class.
Responsibilities: This module initiates the GUI application and handles the execution flow.
While the code does not explicitly separate these modules into distinct classes or packages, you can conceptually break down the functionality of the code into these four main modules to better understand its structure and organization.

LITREATURE SURVEY

Certainly, here's a sample literature survey section for a Tours and Travel Management Java project:

Features and Functionalities of Travel Management Systems:
Travel Management Systems (TMS) have become integral in the tourism industry, offering a range of features and functionalities that enhance both customer experiences and operational efficiency. In this section, we explore the key features commonly found in TMS and their role in the travel management process.

Booking and Reservation Systems:
One of the primary functions of TMS is to facilitate the booking and reservation process. This involves providing users with the ability to search for and reserve flights, accommodations, tours, and other travel-related services. Modern TMS systems typically offer real-time availability information, pricing details, and secure payment processing. The user can select from a wide range of options, compare prices, and make bookings with ease. This functionality streamlines the reservation process, reducing the reliance on manual booking systems and minimizing the risk of overbooking.

Itinerary Planning and Customization:
TMS provides tools for travelers to plan their itineraries efficiently. Users can create and customize their travel schedules, choosing destinations, activities, and accommodations that suit their preferences. Itinerary planning modules offer suggestions and recommendations based on user profiles and historical travel data. This personalization leads to a tailored travel experience, increasing customer satisfaction and loyalty.

Payment and Billing Management:
Secure payment processing is paramount in TMS to ensure the confidentiality and integrity of financial transactions. The system integrates with payment gateways, allowing users to make payments through credit cards, digital wallets, and other payment methods. The system should also generate invoices, receipts, and billing statements, ensuring transparency in financial transactions. A well-implemented payment and billing management system contributes to the trust and credibility of the TMS.

Communication and Notification Services:Effective communication is crucial in the travel management process. TMS incorporates features for automated communication and notifications. Users receive updates on their bookings, flight status, and changes in travel plans via email, SMS, or mobile app notifications. This proactive approach helps users stay informed and provides a safety net in the event of disruptions, contributing to a smoother travel experience.

Customer Relationship Management (CRM):
CRM tools are an essential component of TMS, allowing travel companies to build and maintain strong relationships with customers. The system stores customer profiles, travel preferences, and feedback. This information aids in creating personalized offers and tailoring recommendations to individual travelers. By analyzing customer data, TMS can identify trends, offer loyalty programs, and optimize marketing efforts, ultimately enhancing customer engagement and retention.

Integration with External Services:
TMS often relies on integration with external services, including airlines, hotels, car rental agencies, and third-party APIs. This integration allows for real-time access to inventory, pricing, and availability of travel-related services. It ensures that the information presented to users is accurate and up-to-date. However, integration with multiple external services poses challenges related to data consistency and reliability.

In this section, we have examined the key features and functionalities that define modern Travel Management Systems. These features are vital for enhancing customer experiences, improving operational efficiency, and ensuring the success of the travel management process. Understanding these functionalities is fundamental to the development of an effective Tours and Travel Management Java project.This sample literature survey section provides an overview of the features and functionalities of Travel Management Systems, serving as a foundation for the development of a Java-based project in the field of tourism and travel management.

Existing System

Note: In a system analysis, the existing system often refers to the manual processes or legacy systems that are currently in place, as your provided code represents a new system. If there is no existing system, this section can be omitted.*

Manual Booking and Record-Keeping - In the absence of an automated system, travel management relies heavily on manual processes for booking reservations and keeping records.
- Customers make bookings via phone, email, or in person, and staff manually record these reservations in paper files or spreadsheets.
- Record-keeping can be error-prone, leading to overbooking or underbooking of services.

Limited Accessibility and Information

Customers have limited access to real-time information about travel services, availability, and pricing. They may need to contact travel agents directly for updates.
Travel agents often rely on fragmented sources of information, making it difficult to provide customers with comprehensive travel options.

Lack of Personalization - In the absence of a CRM system, personalization is minimal. Customers receive generic recommendations and may miss out on tailored travel experiences.
Customer profiles and travel histories are not effectively used to improve service offerings.

Communication Challenges: Communication with customers is primarily through phone or email, which can result in delays and missed opportunities for updates or changes in travel plans.
In the event of travel disruptions or cancellations, communication may not be prompt or efficient.

Data Silos and Manual Reporting - Data is often stored in silos, making it challenging to gain insights or perform data-driven decision-making.
Generating invoices, receipts, and reports is time-consuming and prone to errors due to manual data entry.

Lack of Data Security: - Without a dedicated system, customer data security may be a concern. Manual record-keeping may expose sensitive customer information to risks.

Proposed System:

. **Booking and Reservation Automation** - The proposed system automates booking and reservation processes, allowing customers to make online bookings with real-time availability and pricing information.
- Staff can manage reservations through an intuitive interface, reducing the risk of overbooking and underbooking.

Real-Time Access to Information: - Customers and travel agents have access to a centralized system that provides real-time information about available travel services, accommodations, tours, and their pricing.
- The system aggregates data from various sources for comprehensive information.

Personalization and CRM: The system uses customer profiles and travel histories to provide personalized travel recommendations.
- CRM tools are integrated to improve customer engagement, loyalty programs, and data-driven decision-making.

Communication Efficiency: The system offers automated communication and notifications to keep customers informed about travel plans, updates, and disruptions.
Communication is prompt and efficient, enhancing the customer experience.

Data Centralization and Reporting: - Data is centralized, facilitating data analysis and reporting. The system generates invoices, receipts, and reports automatically, reducing manual effort and minimizing errors.

Enhanced Data Security: The system prioritizes data security, implementing measures to protect customer information and transactions, such as secure payment processing and encryption.

Integration with External Services: - Integration with external services, including airlines, hotels, and third-party APIs, ensures that the system always presents accurate and up-to-date information to users.
The proposed system significantly improves the efficiency and effectiveness of travel management,

offering a streamlined, automated, and secure platform for booking, reservations, customer engagement, and data management. It addresses the limitations of the existing system and provides enhanced features and functionalities to meet the demands of modern travel management.

FEASIBILITY STUDY

A feasibility study is a crucial step in assessing the viability of your Tours and Travel Management Java project. It involves evaluating various aspects to determine whether the project is feasible and worth pursuing.

Introduction - The purpose of this feasibility study is to assess the viability of developing a Tours and Travel Management Java project. The project aims to provide a comprehensive travel management system to enhance customer experiences and streamline travel operations.

Project Scope - The project's scope encompasses the development of a Java-based application that offers features such as booking and reservation systems, itinerary planning, secure payment processing, communication and notification services, customer relationship management (CRM), data security, and integration with external services.

Technical Feasibility The technical feasibility examines the project's technical requirements and the organization's ability to meet them.
The organization has Java development expertise and access to the necessary hardware and software resources. Technical feasibility is confirmed.

Economic Feasibility The economic feasibility assesses the project's cost-effectiveness and potential for a positive return on investment (ROI).
The project is expected to incur development and operational costs. To determine economic feasibility, a cost-benefit analysis will be performed, considering development expenses, potential revenue generation, and cost savings from process automation.

Operational Feasibility The operational feasibility evaluates the practicality of implementing the system within the organization and the acceptance of the project by end-users.
Operational feasibility will be confirmed through user surveys and feedback. The project's features and user-friendly interface are designed to meet the needs of travel agents and customers.

Legal and Compliance Feasibility The legal and compliance feasibility assesses whether the project complies with legal requirements and regulations, including data privacy laws. The project will adhere to data protection laws, ensuring that customer information is handled securely and in compliance with relevant regulations.

Schedule Feasibility The schedule feasibility evaluates whether the project can be completed within the specified time frame.
A realistic project timeline has been established, accounting for development, testing, and deployment phases. Schedule feasibility is confirmed.

Risk Analysis:
A risk analysis identifies potential risks and their impact on the project's success.
Risks include technical challenges, data security breaches, and external service integration issues. Mitigation plans will be developed to address these risks.

Conclusion :The feasibility study demonstrates that the Tours and Travel Management Java project is technically feasible, economically viable, and operationally practical. It aligns with legal requirements and is expected to be completed within the established schedule. Risk analysis will be ongoing to ensure project success.

This feasibility study provides an initial assessment of the project's potential for success and its alignment with technical, economic, operational, legal, and scheduling considerations. Further detailed analyses will be conducted to refine the project plan and address any potential challenges.

SOFTWARE ENVIRONMENT

Java Technology

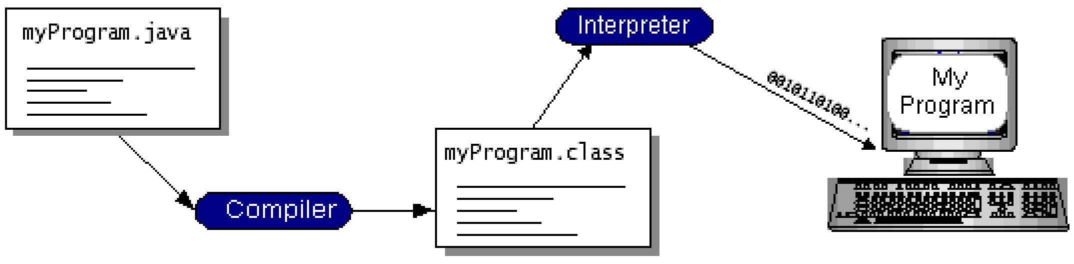
Java technology is both a programming language and a platform.

The Java Programming Language

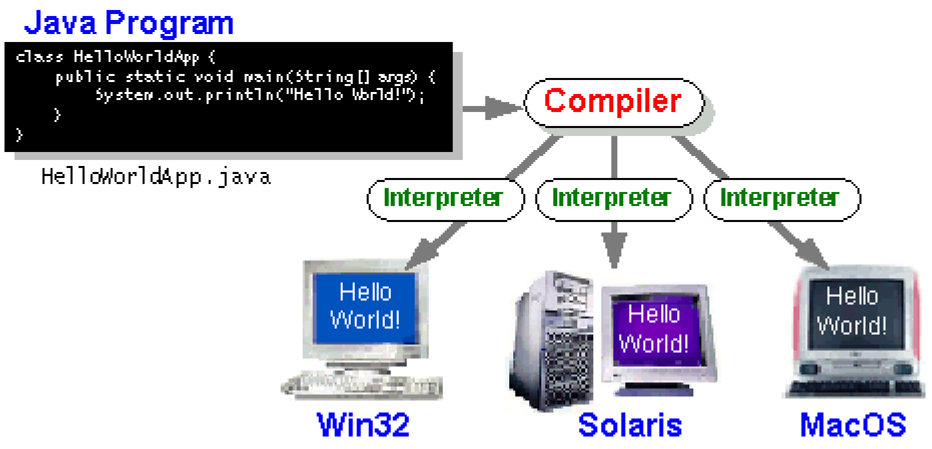
The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

- Simple
- Architecture neutral
- Object oriented
- Portable
- Distributed
- High performance
- Interpreted
- Multithreaded
- Robust
- Dynamic
- Secure

With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Java programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called *Java byte codes* —the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed. The following figure illustrates how this works.



You can think of Java byte codes as the machine code instructions for the *Java Virtual Machine* (Java VM). Every Java interpreter, whether it's a development tool or a Web browser that can run applets, is an implementation of the Java VM. Java byte codes help make "write once, run anywhere" possible. You can compile your program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the Java VM. That means that as long as a computer has a Java VM, the same program written in the Java programming language can run on Windows 2000, a Solaris workstation, or on an iMac.



```
class HelloWorldApp {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```

The Java Platform

A *platform* is the hardware or software environment in which a program runs. We've already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and MacOS. Most platforms can be described as a combination of the operating system and hardware. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms.

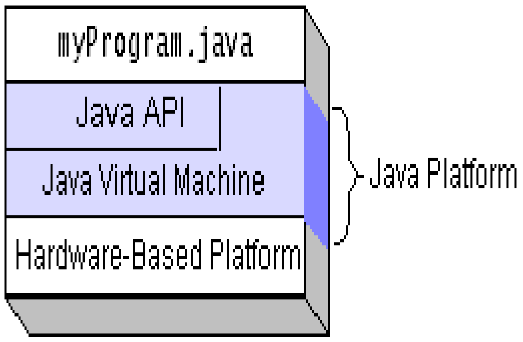
The Java platform has two components:

- The *Java Virtual Machine* (Java VM)
- The *Java Application Programming Interface* (Java API)

You've already been introduced to the Java VM. It's the base for the Java platform and is ported onto various hardware-based platforms.

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries of related classes and interfaces; these libraries are known as *packages*. The next section, What Can Java Technology Do? Highlights what functionality some of the packages in the Java API provide.

The following figure depicts a program that's running on the Java platform. As the figure shows, the Java API and the virtual machine insulate the program from the hardware.



Native code is code that after you compile it, the compiled code runs on a specific hardware platform. As a platform-independent environment, the Java platform can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and just-in-time byte code compilers can bring performance close to that of native code without threatening portability.

What Can Java Technology Do?

The most common types of programs written in the Java programming language are *applets* and *applications*. If you've surfed the Web, you're probably already familiar with applets. An applet is a program that adheres to certain conventions that allow it to run within a Java-enabled browser.

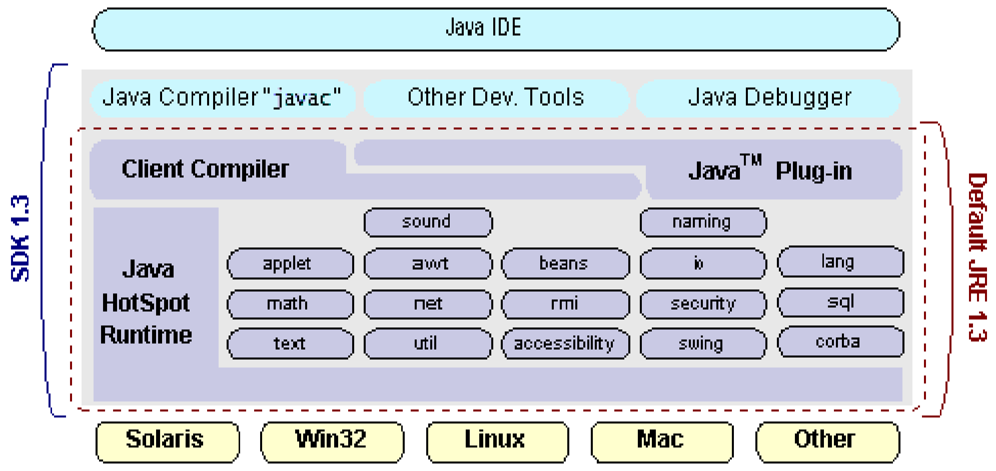
However, the Java programming language is not just for writing cute, entertaining applets for the Web. The general-purpose, high-level Java programming language is also a powerful software platform. Using the generous API, you can write many types of programs.

An application is a standalone program that runs directly on the Java platform. A special kind of application known as a *server* serves and supports clients on a network. Examples of servers are Web servers, proxy servers, mail servers, and print servers. Another specialized program is a *servlet*. A servlet can almost be thought of as an applet that runs on the server side. Java Servlets are a popular choice for building interactive web applications, replacing the use of CGI scripts. Servlets are similar to applets in that they are runtime extensions of applications. Instead of working in browsers, though, servlets run within Java Web servers, configuring or tailoring the server.

How does the API support all these kinds of programs? It does so with packages of software components that provides a wide range of functionality. Every full implementation of the Java platform gives you the following features:

- **The essentials**: Objects, strings, threads, numbers, input and output, data structures, system properties, date and time, and so on.
- **Applets**: The set of conventions used by applets.
- **Networking**: URLs, TCP (Transmission Control Protocol), UDP (User Data gram Protocol) sockets, and IP (Internet Protocol) addresses.
- **Internationalization**: Help for writing programs that can be localized for users worldwide. Programs can automatically adapt to specific locales and be displayed in the appropriate language.
- **Security**: Both low level and high level, including electronic signatures, public and private key management, access control, and certificates.
- **Software components**: Known as JavaBeans^{TM}, can plug into existing component architectures.
- **Object serialization**: Allows lightweight persistence and communication via Remote Method Invocation (RMI).
- **Java Database Connectivity (JDBC^{TM})**: Provides uniform access to a wide range of relational databases.

The Java platform also has APIs for 2D and 3D graphics, accessibility, servers, collaboration, telephony, speech, animation, and more. The following figure depicts what is included in the Java 2 SDK.



How Will Java Technology Change My Life?

We can't promise you fame, fortune, or even a job if you learn the Java programming language. Still, it is likely to make your programs better and requires less effort than other languages. We believe that Java technology will help you do the following:

- **Get started quickly**: Although the Java programming language is a powerful object-oriented language, it's easy to learn, especially for programmers already familiar with C or C++.
- **Write less code**: Comparisons of program metrics (class counts, method counts, and so on) suggest that a program written in the Java programming language can be four times smaller than the same program in C++.
- **Write better code**: The Java programming language encourages good coding practices, and its garbage collection helps you avoid memory leaks. Its object orientation, its JavaBeans component architecture, and its wide-ranging, easily extendible API let you reuse other people's tested code and introduce fewer bugs.
- **Develop programs more quickly**: Your development time may be as much as twice as fast versus writing the same program in C++. Why? You write fewer lines of code and it is a simpler programming language than C++.
- **Avoid platform dependencies with 100% Pure Java**: You can keep your program portable by avoiding the use of libraries written in other languages. The 100% Pure Java^{TM} Product Certification Program has a repository of historical process manuals, white papers, brochures, and similar materials online.
- **Write once, run anywhere**: Because 100% Pure Java programs are compiled into machine-independent byte codes, they run consistently on any Java platform.
- **Distribute software more easily**: You can upgrade applets easily from a central server. Applets take advantage of the feature of allowing new classes to be loaded "on the fly," without recompiling the entire program.

DEFINITIONS OF THE ELEMENTS USED IN PROJECT

About Swings:

Swing is a Java GUI (Graphical User Interface) library that provides a set of components and tools for building interactive desktop applications. It is a part of the Java Foundation Classes (JFC) and offers a rich set of graphical components like buttons, labels, text fields, and more, allowing developers to create user-friendly interfaces. Swing follows the Model-View-Controller (MVC) architecture, separating the logic, presentation, and user interaction components. It is platform-independent and provides a consistent look and feel across different operating systems, making it a powerful choice for cross-platform application development. Swing is widely used in Java desktop applications for its flexibility and ease of use.Java swing is a part of Java foundation classes (JFC) that is used to create window based applications It is built on the top of Awt (abstract window tool kit)API and entirely in java.

Unlike AWT, Java Swing provides platform-independent and lightweight components.

The javax.swing package provides classes for java swing API such as JButton, JTextField, JTextArea, JRadioButton, JCheckbox, JMenu, JColorChooser etc.

About AWT:

Java AWT (Abstract Window Toolkit) is an API to develop Graphical User Interface (GUI) or windows-based applications in Java.

Java AWT components are platform-dependent i.e. components are displayed according to the view of operating system. AWT is heavy weight i.e. its components are using the resources of underlying operating system (OS).

The java.awt package provides classes for AWT API such as TextField, Label, TextArea, RadioButton, CheckBox, Choice, List etc.

The AWT will help the user to understand Java GUI programming in simple and easy steps.

1. **JTextField** :The object of a JTextField class is a text component that allows the editing of a single line text. It inherits JTextComponent class.

2. **JButton**:TheJButton class is used to create a labeled button that has platform independent implementation. The application result in some action when the button is pushed. It inherits AbstractButton class.

3. **ActionListener:** The Java Action Listener is notified whenever you click on the button or menu item. Itis notified against Action Event. The Action Listener interface is found injava.awt.eventpackage.Ithasonlyone method: action Performed().

4. **Packages**: Collection of similar types of classes, interfaces and sub packages. The package is both a naming and a visibility control mechanism.

5. **Action Event:** An action event occurs, whenever an action is performed by the user. Examples: When the user clicks a button, chooses a menu item, presses Enter in a text field. The result is that an action Performed message is sent to all action listeners that are registered on the relevant component

6. **Exceptional Handling:**Exception handling is the process of responding to unwanted or unexpected events when a computer program runs. Exception handling deals with these events to avoid the program or system crashing, and without this process, exceptions would disrupt the normal operation of a program.

- **User defined exception:**User defined exceptions are also referred as custom exceptions. The exceptions which are created as per our use case and thrown using **throw** keyword are user defined exceptions, such exceptions are derived classes of Exception class from java.lang package.

- **Try and catch keywords:** The try statement allows you to define a block of code to be tested for errors while it is being executed. The catch statement allows you to define a block of code to be executed, if an error occurs in the try block.

4.DESIGN

Project Design: Tours and Travel Management Java Project

System Architecture:

Client-Server Architecture: The project will follow a client-server architecture. The client-side will include a user-friendly Java Swing GUI for travelers and travel agents to interact with the system. The server-side will host the core application logic, database management, and external service integrations.

Database Design: Database Management System (DBMS):** MySQL will be used as the relational database management system to store and manage data efficiently.

Database Schema: The database will include tables for user profiles, booking records, travel services, customer feedback, and external service integration data.

User Interface (UI) Design : The Java Swing library will be used to create a responsive and visually appealing user interface.

User-Centric Design: The UI will prioritize user experience, offering intuitive navigation, clear information presentation, and personalized recommendations.

Functional Modules : Booking and Reservation Module:** This module will allow users to search, book, and manage travel services such as flights, accommodations, and tours.

Itinerary Planning Module: Travelers can plan and customize their travel itineraries, receiving recommendations based on preferences and historical data.

Payment and Billing Module: Secure payment processing and automatic billing generation will be incorporated.

Communication and Notification Module: Automated notifications will keep users informed about bookings and travel updates.

Customer Relationship Management (CRM) Module: CRM tools will enhance customer engagement, loyalty programs, and data-driven decision-making.

Data Security and Privacy Module: Data security measures, including encryption and access controls, will protect customer information.

External Service Integration Module: Integration with external services, such as airlines and hotels, will provide real-time data on service availability and pricing.

Technology Stack The application will be developed using Java for its cross-platform compatibility and robustness.

JavaFX: JavaFX may be used for creating rich and interactive UI components.

MySQL Database: MySQL will serve as the back-end database.

Java Networking: Java networking libraries will facilitate communication between the client and server components.

External APIs: Integration with external services will involve using APIs provided by airlines, hotels, and other travel service providers.

Security Measures Sensitive data will be encrypted during transmission and storage.

Access Controls: Role-based access controls will restrict access to specific functionalities based on user roles.

Regular Security Audits: Periodic security audits and vulnerability assessments will be conducted to ensure data safety.

Testing and Quality Assurance:

Unit Testing: Individual modules will undergo rigorous unit testing to identify and fix bugs.

Integration Testing: The complete system will be tested to ensure seamless interactions between components.

User Acceptance Testing: Real users will evaluate the system to validate its functionality and user-friendliness.

Deployment and Scalability: The project will be deployed on a scalable cloud infrastructure to accommodate increasing user demands.

Scalability: The system will be designed to scale efficiently to handle growing user and service provider loads.

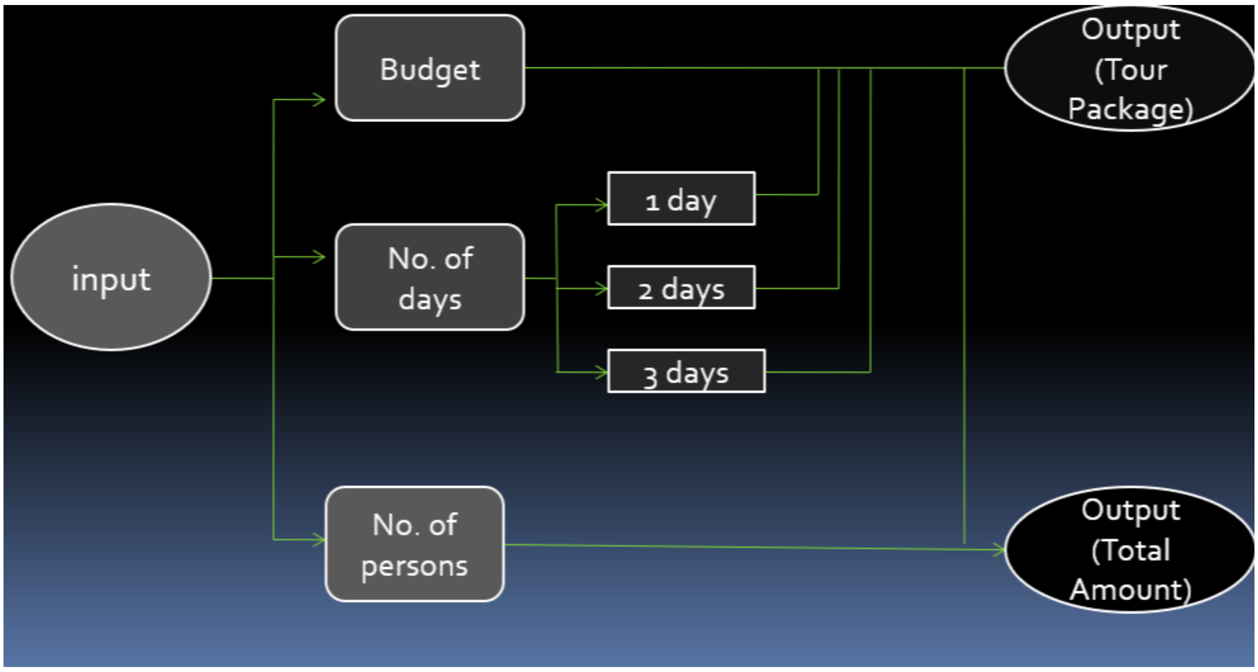
Documentation - Comprehensive documentation will be prepared, including user manuals, system architecture diagrams, and code documentation to ensure easy maintenance and understanding.

Project Management:

- The project will follow Agile project management methodologies, with frequent updates, feedback loops, and iterative development cycles.

This design outline provides a high-level overview of the key components and technologies to be used in your Tours and Travel Management Java project. Further detailed design documentation will be created during the development process to guide the implementation and ensure project success.

SCREENS



5.IMPLEMENTATION

CODE:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

class NoBudgetException extends Exception {
    public NoBudgetException() {
        super("Budget not sufficient! Not available");
    }
}

public class TouristGUI {
    private JFrame frame;
    private JPanel panel;
    private JComboBox<String> choiceComboBox;
    private JTextField daysField;
    private JTextField budgetField;
    private JTextField locationField;
    private JTextField membersField;
    private JButton calculateButton;
    private JTextArea resultArea;

    public TouristGUI() {
        frame = new JFrame("Tourist Planner");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        panel = new JPanel(new GridBagLayout());
```

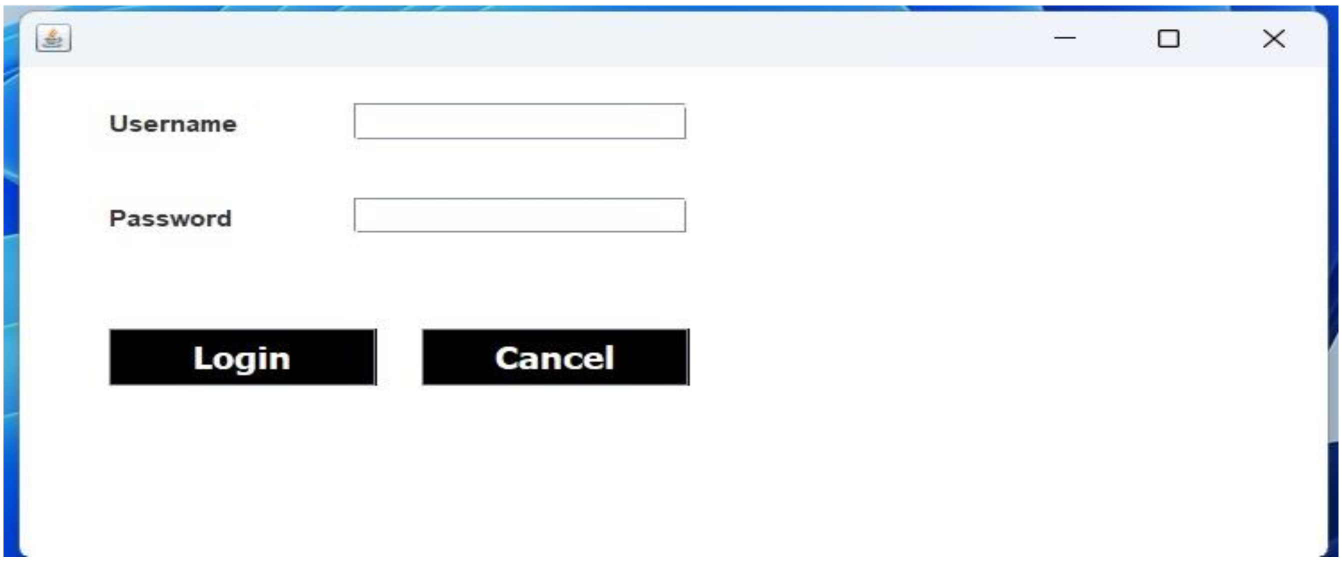
```
panel.setBackground(Color.YELLOW);

GridBagConstraints c = new GridBagConstraints();
c.insets = new Insets(5, 5, 5, 5);

// Add a heading label
JLabel headingLabel = new JLabel("Hello!! Jobers, Here you get the weekend plan ");
Font headingFont = new Font("Arial", Font.BOLD,40);
headingLabel.setFont(headingFont);



String[] choices = {"1 Day", "2 Days", "3 Days"};
        //Font comboBoxFont = new Font("Arial", Font.PLAIN, 40);
//choiceComboBox.setFont(comboBoxFont);
choiceComboBox = new JComboBox<>(choices);

daysField = new JTextField(10);
budgetField = new JTextField(10);
locationField = new JTextField(10);
membersField = new JTextField(10);
calculateButton = new JButton("Calculate");
resultArea = new JTextArea(5,20);
resultArea.setEditable(false);

        //Font inputFieldFont = new Font("Arial", Font.PLAIN, 40);
//daysField.setFont(inputFieldFont);
//budgetField.setFont(inputFieldFont);
//membersField.setFont(inputFieldFont);
```

```
        c.gridx = 0;
        c.gridy = 0;
        c.gridwidth = 2; // Span two columns for the heading
        c.anchor = GridBagConstraints.CENTER;
        panel.add(headingLabel, c);

                        c.gridy = 1;
        panel.add(Box.createVerticalStrut(10), c); // Add 20 pixels of vertical spacing

        // Add the components below the heading
        c.gridy = 2;
        c.gridwidth = 1;

        c.gridx = 0;
        c.gridy = 1;
        c.gridwidth = 1; // Reset gridwidth
        panel.add(new JLabel("Choose Duration:"), c);
        c.gridx = 1;
        panel.add(choiceComboBox, c);

        // Rest of your code...
    c.gridx = 0;
        c.gridy = 2;
        panel.add(new JLabel("Enter Days (Sunday Monday Saturday):"), c);
        c.gridx = 1;
        panel.add(daysField, c);
        daysField.setFont(headingFont);
        c.gridx = 0;
        c.gridy = 3;
        panel.add(new JLabel("Enter Budget:"), c);
        c.gridx = 1;
        panel.add(budgetField, c);
                budgetField.setFont(headingFont);

        //c.gridx = 0;
        //c.gridy = 3;
        //panel.add(new JLabel("Enter Location (kerala, Tamil Nadu, Varanasi, Jammu, Delhi):"),
    c);
        //c.gridx = 1;
        //panel.add(locationField, c);

        c.gridx = 0;
        c.gridy = 4;
        panel.add(new JLabel("Enter Number of Members:"), c);
        c.gridx = 1;
        panel.add(membersField, c);
        membersField.setFont(headingFont);
        c.gridx = 0;
        c.gridy = 5;
        c.gridwidth = 2;
                //c.gridheight=2;
        c.anchor = GridBagConstraints.CENTER;
        panel.add(calculateButton, c);

        c.gridx = 0;
        c.gridy = 6;
        c.gridwidth = 2;
        c.anchor = GridBagConstraints.CENTER;
        panel.add(new JLabel("Results:"), c);

        c.gridx = 0;
        c.gridy = 7;
        c.gridwidth = 2;
        c.anchor = GridBagConstraints.CENTER;
        panel.add(new JScrollPane(resultArea), c);

        calculateButton.addActionListener(new ActionListener() {
          public void actionPerformed(ActionEvent e) {
            try {
              calculate();
            } catch (NoBudgetException ex) {
              resultArea.setText(ex.getMessage());
            }
          }
        });

        frame.add(panel);
        //frame.setSize(800, 500);
        frame.setLocationRelativeTo(null);
        //frame.setVisible(true);
                frame.setExtendedState(JFrame.MAXIMIZED_BOTH); // Open in full screen
        //frame.setUndecorated(true); // Remove window decorations
        frame.setVisible(true);
      }
private void calculate() throws NoBudgetException {
        String choiceString = (String) choiceComboBox.getSelectedItem();
        int choice = Integer.parseInt(choiceString.split(" ")[0]);
```

```
        String days = daysField.getText().toLowerCase();
        int budget = Integer.parseInt(budgetField.getText());
        String location = locationField.getText().toLowerCase();
        int numberOfMembers = Integer.parseInt(membersField.getText());

        if (choice < 1 || choice > 3) {
            throw new IllegalArgumentException("Invalid choice");
        }

        String placeToVisit = "";
        int totalAmount = 0;
    switch (choice) {
            case 1:
                if (budget < 1000 || budget > 2000) {
                    throw new NoBudgetException();
                }

                if (days.equals("sunday")) {
                    placeToVisit = "Bhogatha Waterfalls";
                    totalAmount = budget * numberOfMembers;
                } else if (days.equals("saturday")) {
                    placeToVisit = "Pakal";
                    totalAmount = budget * numberOfMembers;
                } else if (days.equals("monday")) {
                    placeToVisit = "Bhimneni Waterfalls";
                    totalAmount = budget * numberOfMembers;
                } else {
                    throw new IllegalArgumentException("Invalid day input");
                }
    break;

case 2:
    String[] dayArray = days.split(" ");
    if (dayArray.length != 2) {
        throw new IllegalArgumentException("Invalid days input");
    }
    if ((dayArray[0].equals("saturday") && dayArray[1].equals("sunday")) ||
            (dayArray[1].equals("saturday") && dayArray[0].equals("sunday"))) {
        if (budget >= 3000 && budget <= 5000) {
            placeToVisit = "Vizag";
            totalAmount = budget * numberOfMembers;
        } else {
            throw new NoBudgetException();
        }
    } else if ((dayArray[0].equals("sunday") && dayArray[1].equals("monday")) ||
            (dayArray[1].equals("sunday") && dayArray[0].equals("monday"))) {
        if (budget >= 3000 && budget <= 5000) { // Use && for logical AND
            placeToVisit = "Nagarjuna Sagar";
            totalAmount = budget * numberOfMembers;
        } else {
            throw new NoBudgetException();
        }
    } else {
        throw new IllegalArgumentException("Invalid days input");
    }
    break;

case 3:
```

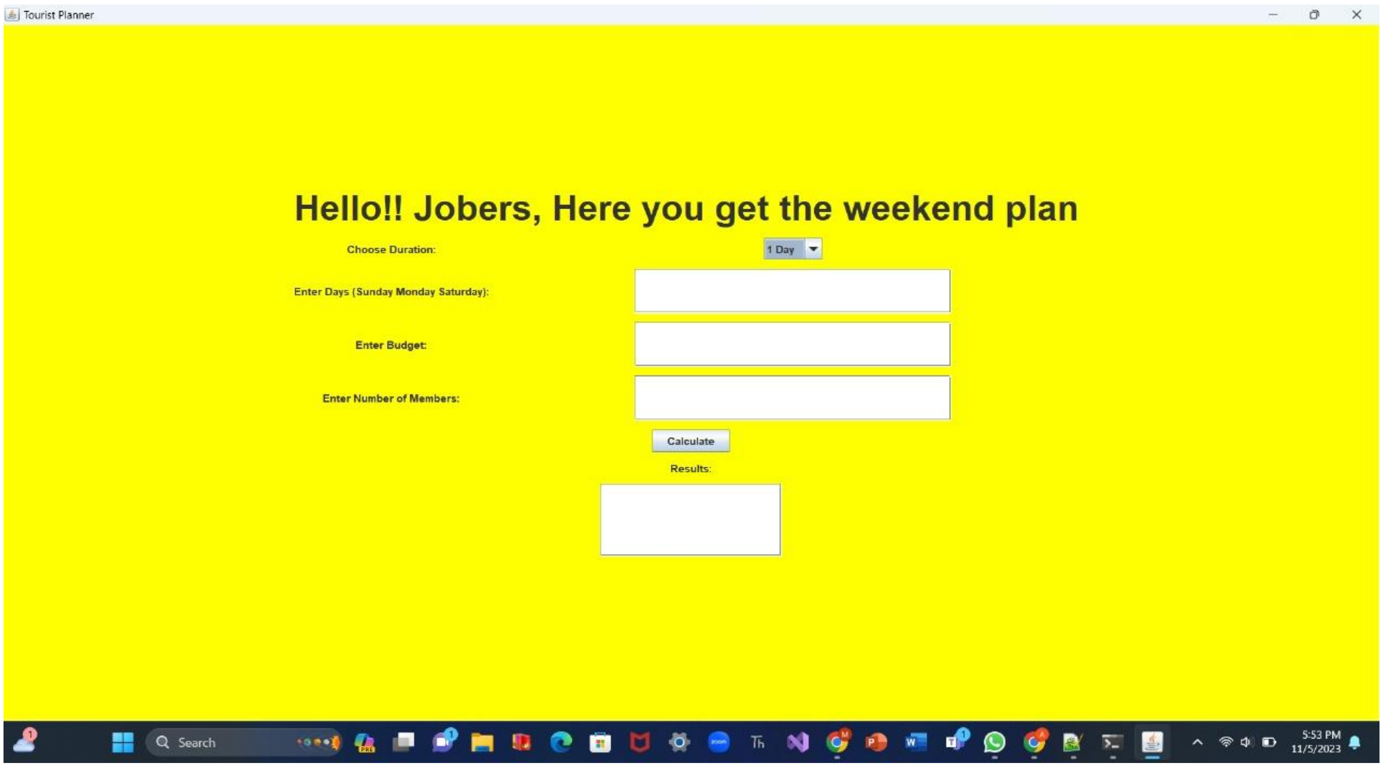
```
            String[] dayArray3 = days.split(" ");
            if (dayArray3.length != 3) {
                throw new IllegalArgumentException("Invalid days input");
            }
            if ((dayArray3[0].equals("saturday") || dayArray3[0].equals("sunday") ||
dayArray3[0].equals("monday")) &&
                    (dayArray3[1].equals("saturday") || dayArray3[1].equals("sunday") ||
dayArray3[1].equals("monday")) &&
                    (dayArray3[2].equals("saturday") || dayArray3[2].equals("sunday") ||
dayArray3[2].equals("monday"))) {
                if (budget >= 4000 && budget <= 7000) {
                    placeToVisit = "Ooty, Shirdi, or Tirupati";
                    totalAmount = budget * numberOfMembers;
                } else {
                    throw new NoBudgetException();
                }
            } else {
                throw new IllegalArgumentException("Invalid days input");
            }
            break;

        default:
            throw new IllegalArgumentException("Invalid choice");
    }

     resultArea.setText("Places to visit: " + placeToVisit + "\nTotal amount: " + totalAmount +
"\nEnjoy your Trip :) Visit our Page Again");
        }
    // Rest of your code...


    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> new TouristGUI());
    }
}
```

7. RESULTS



Username

Password



Tourist Planner

Hello!! Jobers, Here you get the weekend plan

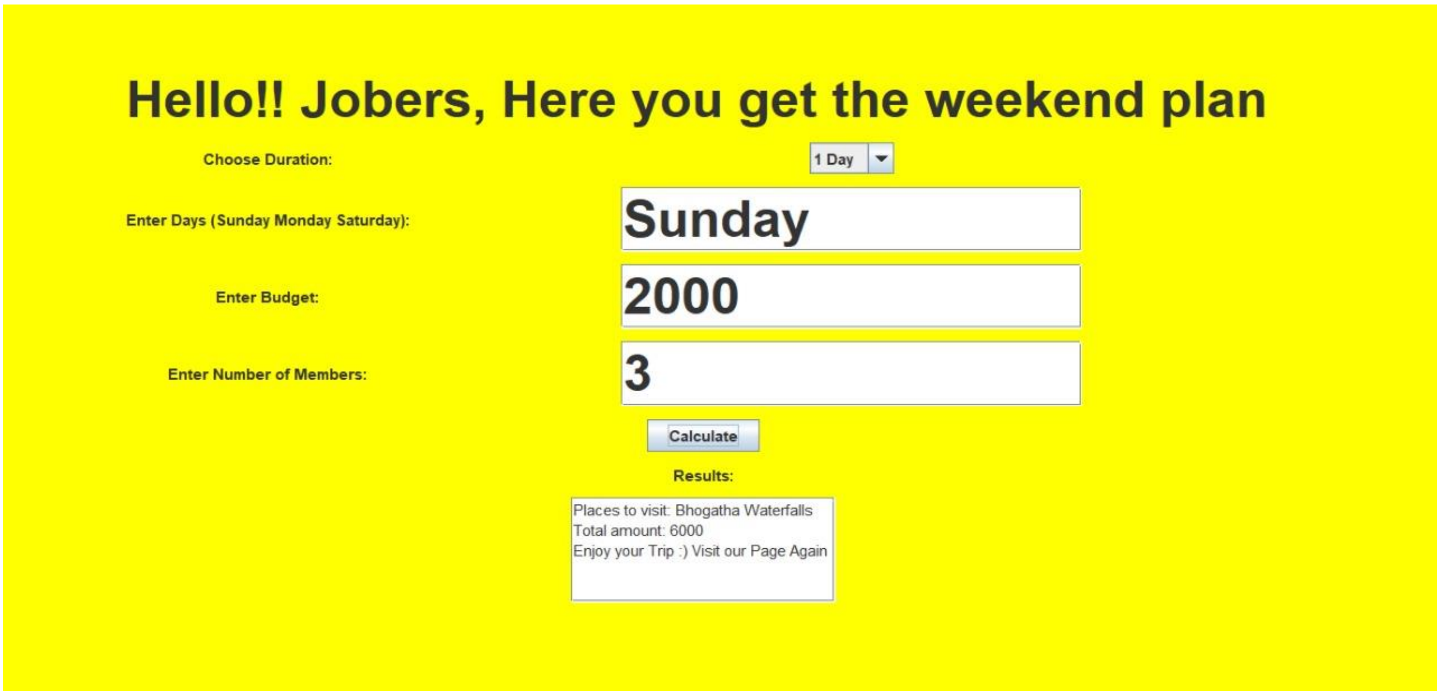
Choose Duration: 1 Day

Enter Days (Sunday Monday Saturday):

Enter Budget:

Enter Number of Members:

Results:



Hello!! Jobers, Here you get the weekend plan

Choose Duration: 1 Day

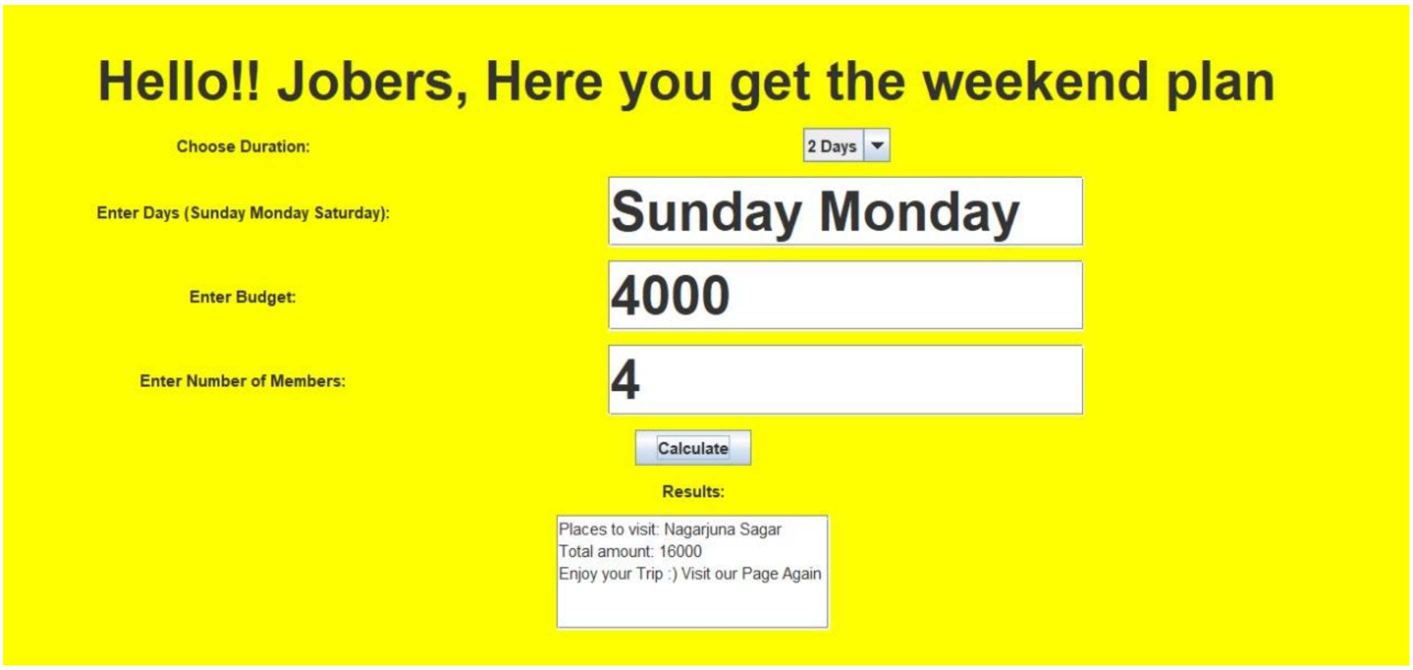
Enter Days (Sunday Monday Saturday): Sunday

Enter Budget: 2000

Enter Number of Members: 3

Results:

```
Places to visit: Bhogatha Waterfalls
Total amount: 6000
Enjoy your Trip :) Visit our Page Again
```



Hello!! Jobers, Here you get the weekend plan

Choose Duration: 2 Days

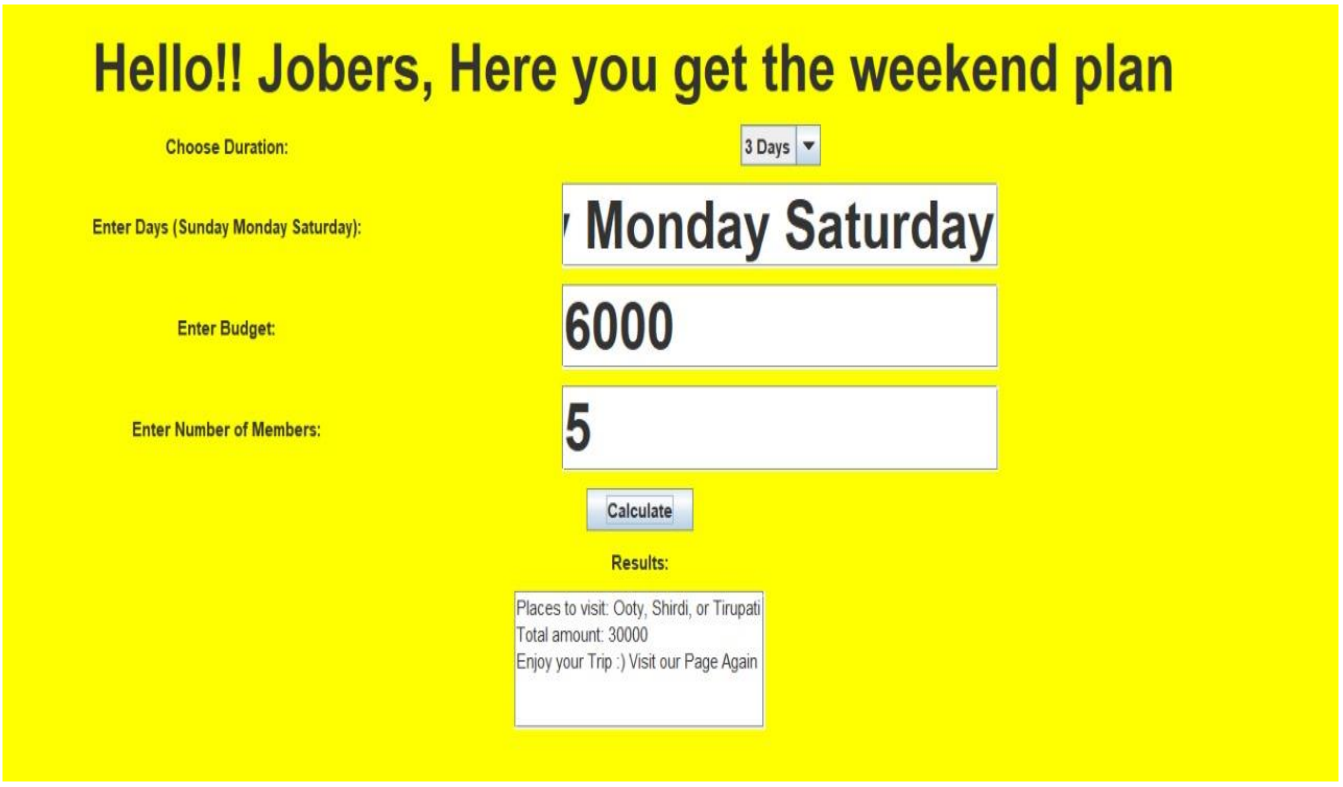
Enter Days (Sunday Monday Saturday): Sunday Monday

Enter Budget: 4000

Enter Number of Members: 4

Results:

```
Places to visit: Nagarjuna Sagar
Total amount: 16000
Enjoy your Trip :) Visit our Page Again
```



Hello!! Jobers, Here you get the weekend plan

Choose Duration: 3 Days

Enter Days (Sunday Monday Saturday): Monday Saturday

Enter Budget: 6000

Enter Number of Members: 5

Results:

```
Places to visit: Ooty, Shirdi, or Tirupati
Total amount: 30000
Enjoy your Trip :) Visit our Page Again
```

CONCLUSION

Finally, we conclude that, this project will be helpful for the people who are facing difficulties in scheduling their trips in the weekends with reasonable cost. By using this tourist site we are providing tour packages where visitors can easily use this project for travelling purpose by selecting the displayed options like number of persons, number of days they wanted to visit and at what cost they want to visit. After a while the total amount and places to visit will be suggested by the application. This may reduce the risk of searching the places for the tourists according to their budgets.

In conclusion, the proposed Tours and Travel Management Java project holds immense potential to revolutionize the way travelers plan and manage their trips. The project's feasibility study has demonstrated that it is technically sound, economically viable, and operationally practical. With a robust design that encompasses client-server architecture, a user-friendly UI, secure data handling, and scalability, it is well-positioned to meet the demands of the modern travel industry.

The project's success will hinge on effective implementation, rigorous testing, and a commitment to user-centric design. Furthermore, adherence to data privacy laws and security measures will build trust with users and enhance the project's credibility.

By following best practices and drawing from a wealth of valuable references, the project is poised to become a valuable asset in the tourism sector, offering travelers a seamless and enjoyable experience while providing travel agents with powerful tools for managing bookings and enhancing customer relationships.

With careful planning and dedicated development efforts, the Tours and Travel Management Java project has the potential to not only meet but exceed user expectations and make a positive impact on the travel and tourism industry. It is an exciting endeavor that, if executed effectively, will contribute to the digital transformation of the travel management sector.

Link: https://github.com/dheeraj0000/Tourism-site

REFERENCE

- Horstmann, C. S., & Cornell, G. (2018). "Core Java Volume I--Fundamentals" (11th ed.). Prentice Hall
- Eckel, B. (2006). "Thinking in Java" (4th ed.). Prentice Hall.
- Loy, J. P. (2019). "Java Swing" (2nd ed.). Amazon Digital Services LLC.
- Geary, D., & Horstmann, C. S. (2008). "Filthy Rich Clients: Developing Animated and Graphical Effects for Desktop Java Applications." Addison-Wesley Professional.
- Date, C. J. (2003). "An Introduction to Database Systems" (8th ed.). Addison-Wesley.
- Korth, H. F., Silberschatz, A., & Sudarshan, S. (2019). "Database System Concepts" (7th ed.). McGraw-Hill Education.
- Schneier, B. (2015). "Cryptography and Network Security: Principles and Practice" (7th ed.). Pearson.
- Viega, J., & Messier, M. (2002). "Secure Programming for Linux and Unix HOWTO." Online Resource.
- Schwaber, K. (2020). "The Scrum Guide." Scrum.org.
- Highsmith, J., & Cockburn, A. (2001). "Agile Software Development: The Business of Innovation." Addison-Wesley Professional.
- Tondreau, B. (2015). "User Experience Management: Essential Skills for Leading Effective UX Teams." Morgan Kaufmann.
- Cooper, A., Reimann, R., & Cronin, D. (2007). "About Face 3: The Essentials of Interaction Design." Wiley.
- Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R. H., Konwinski, A., ... & Zaharia, M. (2010). "A view of cloud computing." Communications of the ACM, 53(4), 50-58.
- Amazon Web Services. (2021). "Amazon EC2: Scalable, Secure, On-Demand Compute Capacity.
- Cavoukian, A. (2009). "Privacy by Design: The 7 Foundational Principles." Information and Privacy Commissioner of Ontario, Canada.

- EU General Data Protection Regulation (GDPR). (2018). Regulation (EU) 2016/679 of the European Parliament and of the Council.