Travel Tool

Design Document

CSC 490

Jacob Johnson

Robert Minerd

Rudolph Hanzes

Kaleb Piper

Greg Bittinger

December 4, 2020

Instructor Comments/Evaluation

Table of Contents

Contents

Abstract

Travel tool will accept inputs from the client to provide travel options and pricing within the United States. Collected information will include current location, destination, trip duration, and their preferred budget. Travel tool will take the User's input information and send it to multiple websites retrieving a variety of transport and lodging options within their budget. The User will also be able to choose how their budget is divided, either favoring a higher quality transport or travel.

This software design document will specify the classes and modules we will use in the production of Travel tool. It will go in depth on individual components, design organization, programming languages, usability, implementation, schematics, team details, and functions. This document will be used as an outline through development of Travel Tool providing detailed design information, priorities, and organization and serve as a guideline to improve achievability with the development of Travel Tool.

Description of the Document

Purpose and Use

The purpose of this document will be to outline design implementation through the use of modules such as cohesion and coupling, block diagrams, architectural diagrams as well as class layouts and descriptions. We will describe the organization behind our design, explaining team member objectives and strategies. The program languages we will be using to develop travel tool will be declared in this document as well as describing our implementation process via a timeline. Lastly, we will describe our testing procedures for each stage of development.

Another major purpose of this document is to describe the techniques we will be using to efficiently organize and implement code later in the developmental process. This includes classes and objects used to perform specific functions within Travel Tool. A detailed description of each individual class and object will be expressed in this document along with any input, output, return parameters, and types used in each class. The modules used in each class will outlined and defined as well.

Links to Specification Document

Our specification document provided a basic outline of how we would like Travel Tool to perform, this document will further explain each of these objectives as well as implementation techniques. The specification document also served as a contracted agreement between the Travel Tool development team and the client, this design document will provide the client with an understanding of our implementation methods for the design, development, testing, and deployment of our project.

Intended Audience

The intended audience for this document is the development team, as well as any stakeholders behind Travel Tool. This document further explains details about the design of our project. The diagrams used may only be understandable by those with experience in computer science or a similar field. Third party members may use this document to confirm the client's requests are met by Travel Tool. Software developers should use this design specification document to further their understanding of our developmental process and coding techniques.

Project Block System Diagram

The Travel Tool consists of a frontend and a backend. The frontend is where the interface will be displayed, and the user will interact with the tool. Data that the user inputs will be sent to a backend where that uses that information to scrape the internet and find appropriate travel and hotel options. Once that process is completed the accommodation / transportation data will be sent back to the frontend where combinations will be displayed to the user. The user has the option to change the budget which will then send the new data back to the backend and repeat the process.

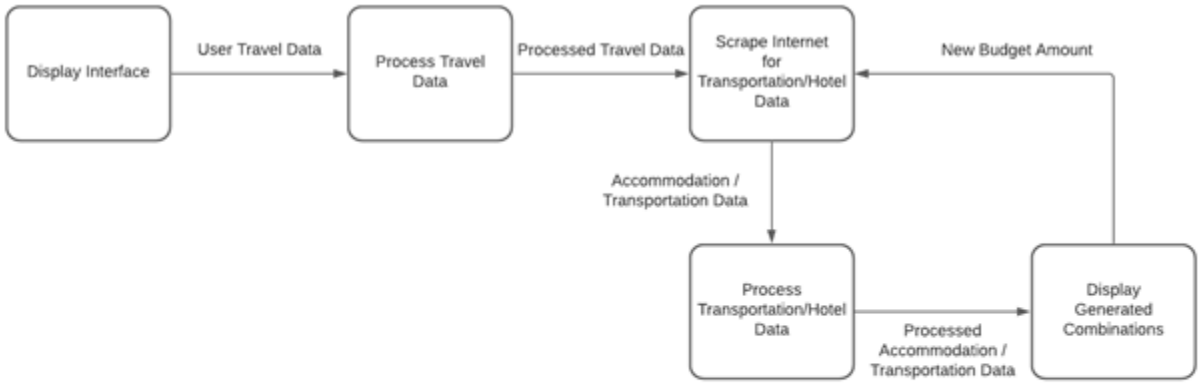


Figure 1: Block Diagram

Design Details

System Modules and Responsibilities

System Overview

Travel Tool will provide an application that will give users an option to make their travel plans in one place. It relies on the user inputting the data that is needed to find the desired information the user is looking for. It will ask the user to input the current location, travel destination, trip duration, and the budget in which they would like to spend. The user interface will be separated into different sections for each section of information that is needed from the user. This will help to keep everything organized and easy for the user to input their information.

The user will only have to enter their information using the user interface that will be given to them. They will do this by entering the information on whatever device they are using at the time. The information will then be sent to multiple different websites which will retrieve a variety of transport and lodging options within their budget. The user will then choose how their budget will be divided, either favoring a high or low budget quality travel option.

The main components of Travel Tool are the web application, the user, and information gathered after the user inputs the desired data. The web application and user work together to retrieve the data that is desired by the user. The general overview will give a scenario of this application being used in the real world.

Overview

Travel Tool will be implemented as a free to use web software accessible through any standard search engine. Development will be conducted by the project team and provided for anyone looking for travel rates and information on their trip in the United States. Testing will also be conducted by our team as well as some external sources such as volunteers, friends, and family members for outside opinions and feedback.

Travel Tool will first be accessed from the users' favorite search engine such as google, they will then find the correct link to the Travel Tool website to gain access if they have an active internet connection. Users will then be brough to the Travel Tool website home page with instructions on how to use our software. The user will be provided with a series of questions to fill in such as their current location, destination, travel date, preferred method of transportation, lodging preferences, and budget.

Travel Tool will then provide users with the ability to divide their budget either spending more on transportation or stay. Our software will then take the provided data from the client and search the internet for the best rates and a variety of travel and lodging options to choose from. Travel Tool will provide options both under and around their preferred budget to provide flexibility in case the client wishes to spend a bit more on their trip. The client will be able to select one of each for transportation and stay, then Travel Tool will calculate the total cost and provide links to the websites used to book their trip.

Architectural Diagram

Figure 2 shows an architecture diagram that represents the Travel Tool. The diagram shows the layout of the system and its processes. Along with that it shows what data will be sent from which process to the next. The data being sent from each process to the next is vital to conduct certain calculations and generate our combinations for the user during the last process.

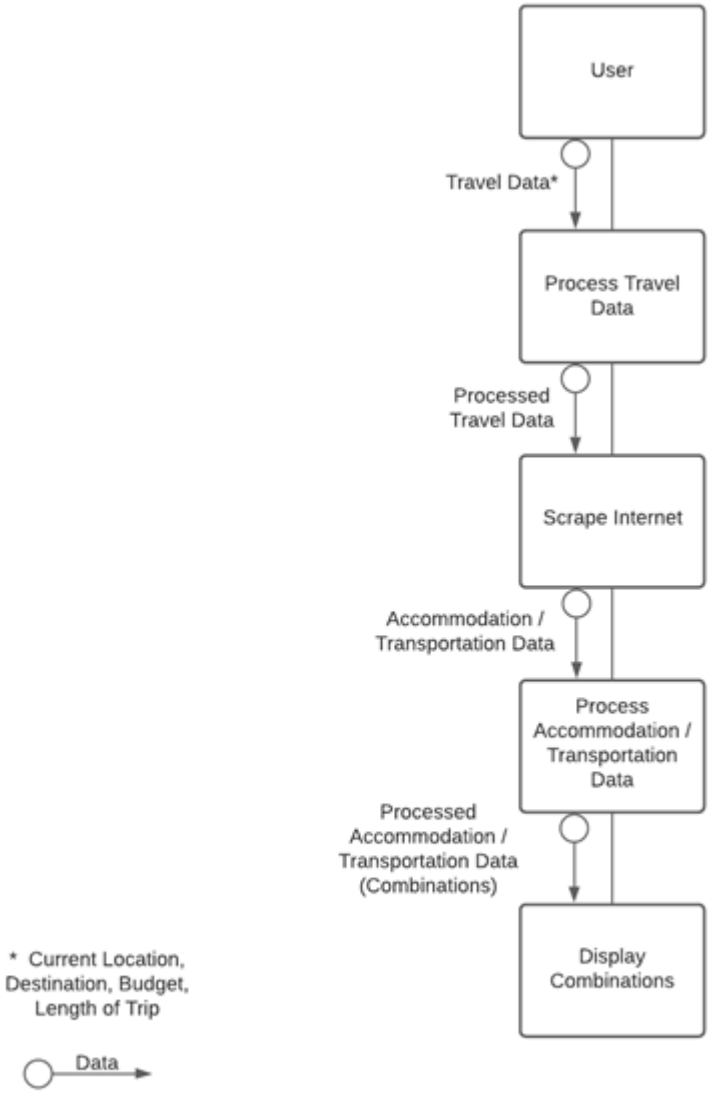


Figure 2: System Architecture

Module Cohesion

Cohesion is considered the degree that elements inside modules fit together [1]. Cohesion is commonly defined in seven categories with some having high cohesion and some having low. Because there is a series of operations that are followed in a sequence of steps, the type of cohesion in our project would be considered communicational. While not the high degree of cohesion, it is the third highest of the seven, giving the project sufficient cohesion.

Module Coupling

Coupling is the degree of interaction between modules [1]. This is ranked similarly to cohesion but with five levels instead of seven. The Travel Tool's modules are primarily data based, so the type of coupling would be considered data coupling. This is the best degree of coupling when ranking the five different types.

Design Analysis

Data Flow

The data flow of the Travel Tool can be seen in Figure 3. The user inputs the travel data that is then sent to be processed. Once processed the data is used to help scrape the internet and find different accommodation and transportation options. This data is then processed, and combinations are generated using the data. These combinations are then displayed to the user.

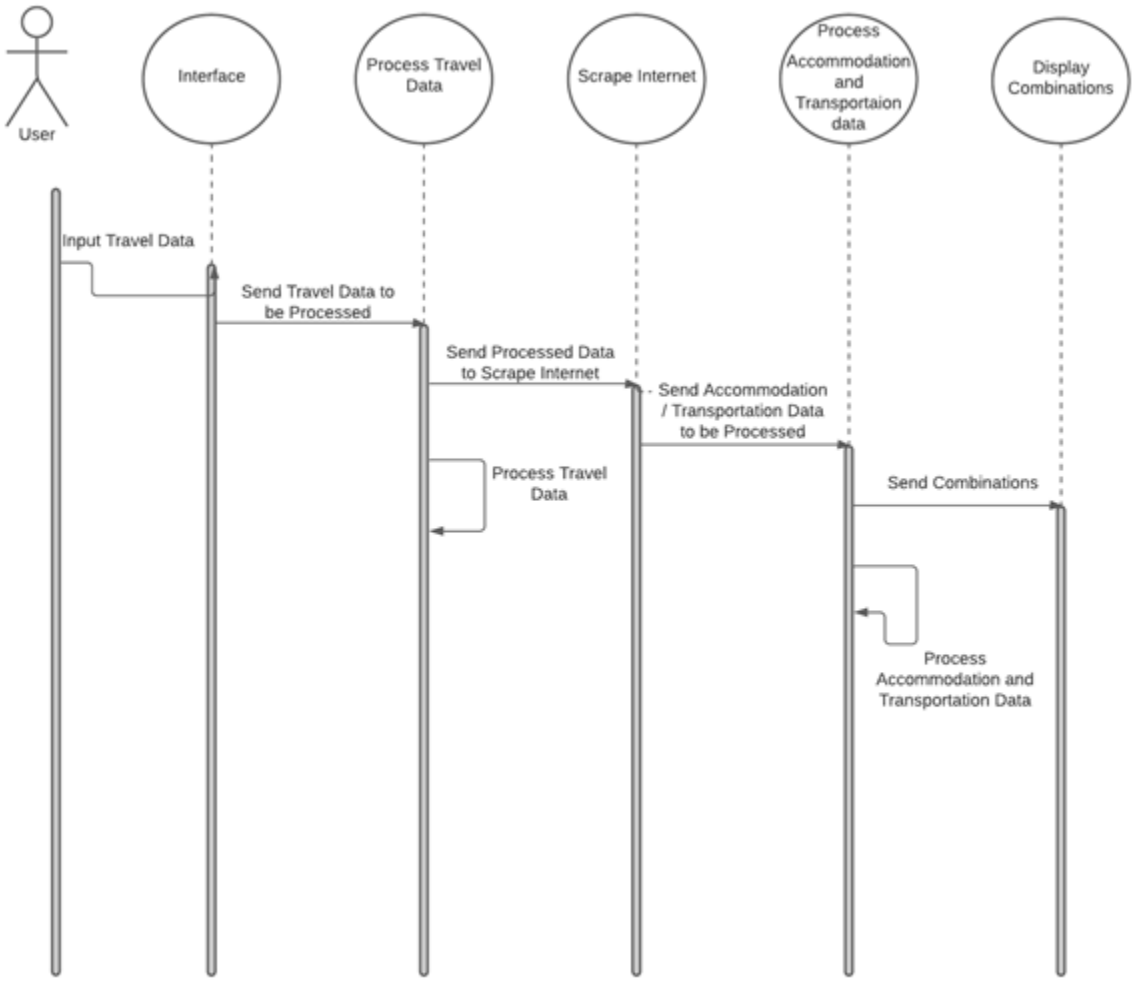


Figure 3: Dataflow Diagram

Design Organization

Detailed Tabular Description of Classes and Objects

This section will provide an overview of and descriptions of all classes and objects. Figure 4 below gives a general overview of the classes that will be used in the Travel Tool.

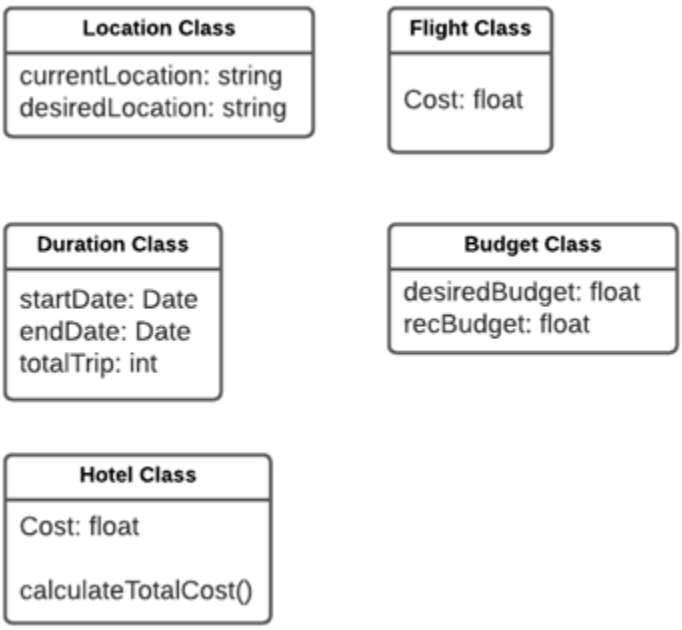
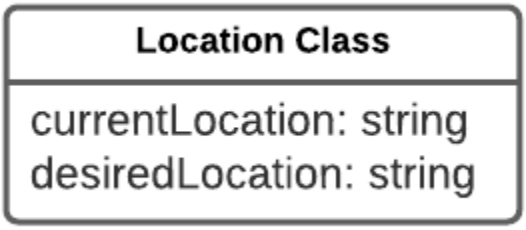


Figure 4: Class Overview

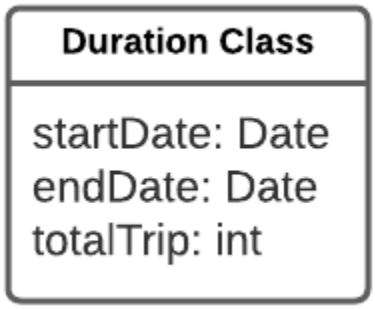
Location Class



Tabular Description of Location Class:

Module Name	Location Class
Module Type	Class / Object
Data Members	currentLocation: String desiredLocation: String
Constraints	Class must contain necessary constructors and functions to access the data.
Description	This class will hold both of the locations set by the user when inputting their travel information.

Duration Class

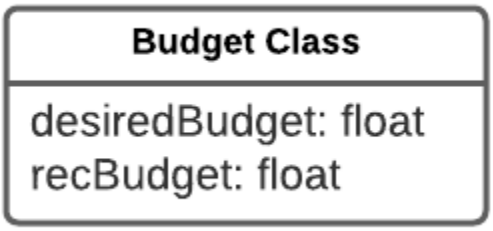


Tabular Description of Duration Class:

Module Name	Duration Class
Module Type	Class / Object
Data Members	startDate: Date endDate: Date

	totalTrip: Int
Constraints	Class must contain necessary constructors and functions to access the data. The proper library must also be imported for the data type Date.
Description	This class will hold the start and end dates of the trip. The class will also hold the total length of the trip.

Budget Class

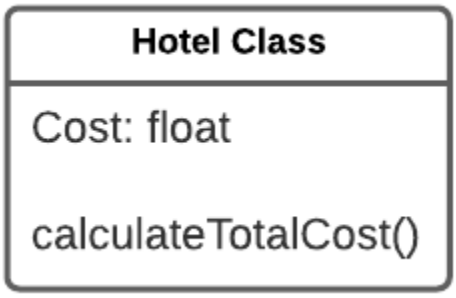


Tabular Description of Budget Class:

Module Name	Budget Class
Module Type	Class / Object
Data Members	desiredBudget: Float recBudget: Float
Constraints	Class must contain necessary constructors and functions to access the data.

Description	This class will hold the budget desired from the user as well as the budget recommended by the Travel Tool.

Hotel Class



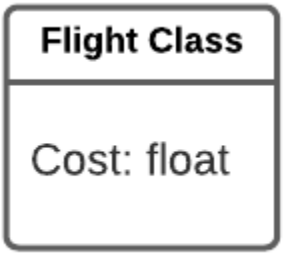
Tabular Description of Budget Class:

Module Name	Hotel Class
Module Type	Class / Object
Data Members	Cost: Float
Member Functions	calculateTotalCost()
Constraints	Class must contain necessary constructors and functions to access the data. Must inherit duration class and location class.
Description	This class calculate the total cost of a hotel stay and hold that number.

Hotel Class Member Functions:

calculateTotalCost() – This takes the cost of one night in a hotel and multiplies it by the total duration of the trip.

Flight Class



Tabular Description of Budget Class:

Module Name	Flight Class
Module Type	Class / Object
Data Members	Cost: Float
Constraints	Class must contain necessary constructors and functions to access the data. Must inherit duration class and location class.
Description	This class holds the cost of a flight from the current location to the desired location.

Functional Descriptions

Input / Output / Return Parameters and Types

location():

Input:

The location class will require input from the user for both their current location and the desired destination.

Output:

The website will display the information they are typing in as well as conformation the data was successfully received.

Return Parameters:

This function will return either the success or failure of the input received.

Types:

A string will be used for the location data.

duration():

Input:

The duration class will require user input to retrieve the length of their trip.

Output:

The website will display the information they are typing in as well as conformation the data was successfully received.

Return Parameters:

This function will return either the success or failure of the input received.

Types:

The data type for the duration will be an int for the total trip and then dates will need to be entered for the startDate and endDate.

budget():

Input:

The budget class will require user input to retrieve the user's desired budget.

Output:

The website will display the information they are typing in as well as conformation the data was successfully received.

Return Parameters:

This function will return either the success or failure of the input received.

Types:

A float will be used for the budget data.

hotel():

Input:

Output:

List of possible lodging options within and around the user's desired budget.

Return Parameters:

Types:

A float will be used for the hotel data.

Flight():

Input:

Output:

List of possible flight options within and around the user's desired budget

Return Parameters:

Types:

A float will be used for the flight data.

Modules Used

Figure 4 provides a chart that shows the class modules that will be made for this project. Along with the class modules there is a many entry of data that will be needed from the user to be entered in order for the application to find what the user is looking for.

Files Accessed

Currently the Travel Tool primarily has data being moved back and forth directly and will not need extra storage for those files as they will be discarded when the user exits the webpage. This is subject to change if it is found that storing more data benefits the user experience. However, due to the changing of flight and hotel prices this is unlikely.

Real-Time Requirements

A stable internet connection is required to use Travel Tool. This can come in the form of home Wi-Fi or a mobile device with a cellular connection such as LTE. There will not be many requirements that you will need to access our application besides the stable internet connection and the device that you will be using to connect to it.

Messages

Error Code	**Error Description**
0001	Invalid data entered for the location, must enter a location in the United States
0002	Invalid data entered for the destination, must enter location in the United States
0003	Invalid data entered for the budget

0004	Invalid data entered for duration
0005	A connection could not be made to the application.

Narrative / PDL

Main Menu:

1. Enter Current Location
2. Enter Travel Location
3. Trip Duration / Amount of People
4. Travel Budget

Enter Current Location:

1. Address
2. City
3. State
4. Zip code

Enter Travel Location:

1. Address
2. City
3. State

4. Zip code

3. Trip Duration / Amount of People:

1. Date of Arrival – Date of Departure

2. Number of People Traveling

4. Travel Budget

1. Desired budget you would like to spend

Decision: Programming language / Reuse / Portability

We will be utilizing the Python programming language for web scraping and other applications during development to take advantage of the many libraries and functions that the language provides. We will also be using Java for its robust and scalable developer tools, as well as its many useful plugins.

Implementation TimeLine

The following timeline contains deadlines for certain aspects of the project to maintain consistent progress.

Design Testing

Design testing will occur consistently throughout the project's development. Most testing will occur prior to the completion of the development, making our testing primarily non-execution based. This testing will consist of each group member reviewing the code to ensure there are no syntax or logical errors.

Once sections become complete the group may test certain aspects of the project through execution. This will be to ensure that certain sections work on their own before they are connected to the other processes.

After the full completion of the project there will be more execution-based testing. This is to ensure that all aspects of the project work when combined. Each group member will use the Travel Tool as a normal user and attempt to cause errors or other issues. If these issues arise the group will go back into the code to make the necessary changes.

Appendix A: Technical Glossary

Backend – the data access layer that user does not interact with

Classes - a defined structure to create an object in an object-oriented programming language

Cohesion - how related the functions within a single module are

Coupling - refers to the interdependencies between modules

Frontend – everything the user interacts with

Non execution-based testing – the module being tested is always reviewed by the team, leading to rapid and thorough fault detection

Appendix B: Team Details

Workflow Leader: Jacob Johnson

Greg Bittinger – Reviewed the document and made additions to various sections. Added a technical glossary and other sections to the document.

Robert Minerd – Worked on different sections throughout the document being made, while helping with reviewing and editing the document throughout the process of it being developed.

Kaleb Piper - Reviewed the document as a whole, making adjustments to grammar and typos when any were found, as well as reviewing the figures in the document for any errors.

Rudolph Hanzes – Worked on beginning sections such as Abstract, Description, Links To Specification, and intended Audience as well as looked over and reviewed various sections within the document.

Jacob Johnson – Jacob worked on various sections of this document and created many of the figures, while also reviewing a majority of the document during its creation and after completion.

Appendix C: Workflow Authentication

I, Greg Bittinger, hereby attest that I have performed the work as documented herein.

Signature: Gregory Bittinger Date: 12/2/20

I, Kaleb Piper, hereby attest that I have performed the work as documented herein.

Signature: Kaleb Piper Date: 12/2/20

I, Robert Minerd, hereby attest that I have performed the work as documented herein.

Signature: Robert Minerd Date: 12/2/20

I, Rudolph Hanzes, hereby attest that I have performed the work as documented herein.

Signature: Rudolph Hanzes Date: 12/2/20

I, Jacob Johnson, hereby attest that I have performed the work as documented herein.

Signature: Jacob Johnson Date: 12/2/20

Appendix D: Writing Center Report

Cal U Vulcan Learning Commons Report

Client: Jacob Johnson

Staff or Resource: Nathan Z.

Date: December 4, 2020, 4:00pm - 5:00pm

What course was serviced by this visit?: CSC 490

Did the student request that the instructor receive a visit report?: Yes

Please provide any additional comments relevant to this session.:

I didn't finish reviewing the document. I believe I had only one page left (your design testing page). If you have any questions, comments, or concerns, feel free to email me at any time at Zis0062@calu.edu.

How did the process of this consulting session address the established goals?:

I reviewed the client's document and made suggestions regarding grammar and syntax.

Appendix E: References

[1] Software Engineering: Coupling and Cohesion. (2020, January 08). Retrieved December 05, 2020, from https://www.geeksforgeeks.org/software-engineering-coupling-and-cohesion/