Project Summary

- Team Members
    - Long Nguyen
    - Edwin Chiang
    - Danny Nguyen
- Overview
    - We will be creating a web application that at its core will allow users to budget for a road trip with more features we haven't decided on yet. With our app, we are trying to create a central location that should allow users to plan and budget for a trip at the same time in the same UI. The app should suggest multiple routes to a location, list the estimated cost for travel for each, and possibly list points of interest along each route

Project Requirements:

- Requirements and Responsibilities
    - The system should allow users to create accounts to separate trips from other users. The system should also allow a single user to create multiple trips as well as modify (update/delete) them.

Use Cases:

Use Case #1
Name – User creation
Brief Description – User creating account to tie to trips that may be created
Actors – User, database
Preconditions – User does not already exist in database
Basic Flow – Users creating an account will select a username that isn't already taken as well as a password, these two fields will be verified with RegEx
Alternate Flows – Shouldn't have any alternative flows
Exception Flow – Username chosen already exists in database, user redirected to account creation page and notification given that username already exists
Post Conditions – Username and hash for password updated in the database

Use Case #2
Name – Trip creation
Brief Description – User wants to create a trip specific to their needs, user can view and edit this later as well
Actors – Trip, user, database
Preconditions – User is signed it / user exists in database already
Basic Flow – User creates new trip and selects certain specifications they might want (visit certain points of interest during trip)
Alternate Flows – User decides to cancel the creation of this object, delete any information stored on it currently

Exception Flow – Not too sure on this, possibly trip exceeds budget or destination is in some restricted area. System will alert user of this
Post Conditions – Trip is created in the database, related to a user who created it

Use Case #3
Name – Trip modifications
Brief Description – User wants to modify an existing trip
Actors – Trip, user, database
Preconditions – Trip to be modified can be found in the database
Basic Flow – User selects and existing trip and changes some element within that object
Alternate Flows – User decides against changes and leaves object unmodified
Exception Flow – Basically identical to alternate flow
Post Conditions – Changes are updated and reflected in the database

Use Case #4
Name – Trip deletion
Brief Description – User wants to remove a certain trip from some list
Actors – Trip, user, database
Preconditions – Trip to be deleted can be found in the database
Basic Flow – User selects an existing trip to be deleted and change should be reflected in the database
Alternate Flows – User is able to delete multiple trips at once
Exception Flow – There will be dialogue boxes that pop up to make sure of the user's decision
Post Conditions – Item removed from database

Use Case #5
Name – User deletion
Brief Description – User wants to delete their account and information from the service
Actors – Trip, user, database
Preconditions – User information and trips to be deleted can be found in the database
Basic Flow – User wishes to remove their account from the website triggering the removal of any trips and personal information associated with that user.
Alternate Flows – User may remove specific information associated to their account but still retain it.
Exception Flow – A dialogue box will pop up confirming the user's actions to be sure they don't delete their account by accident.
Post Conditions – User account removed from the database

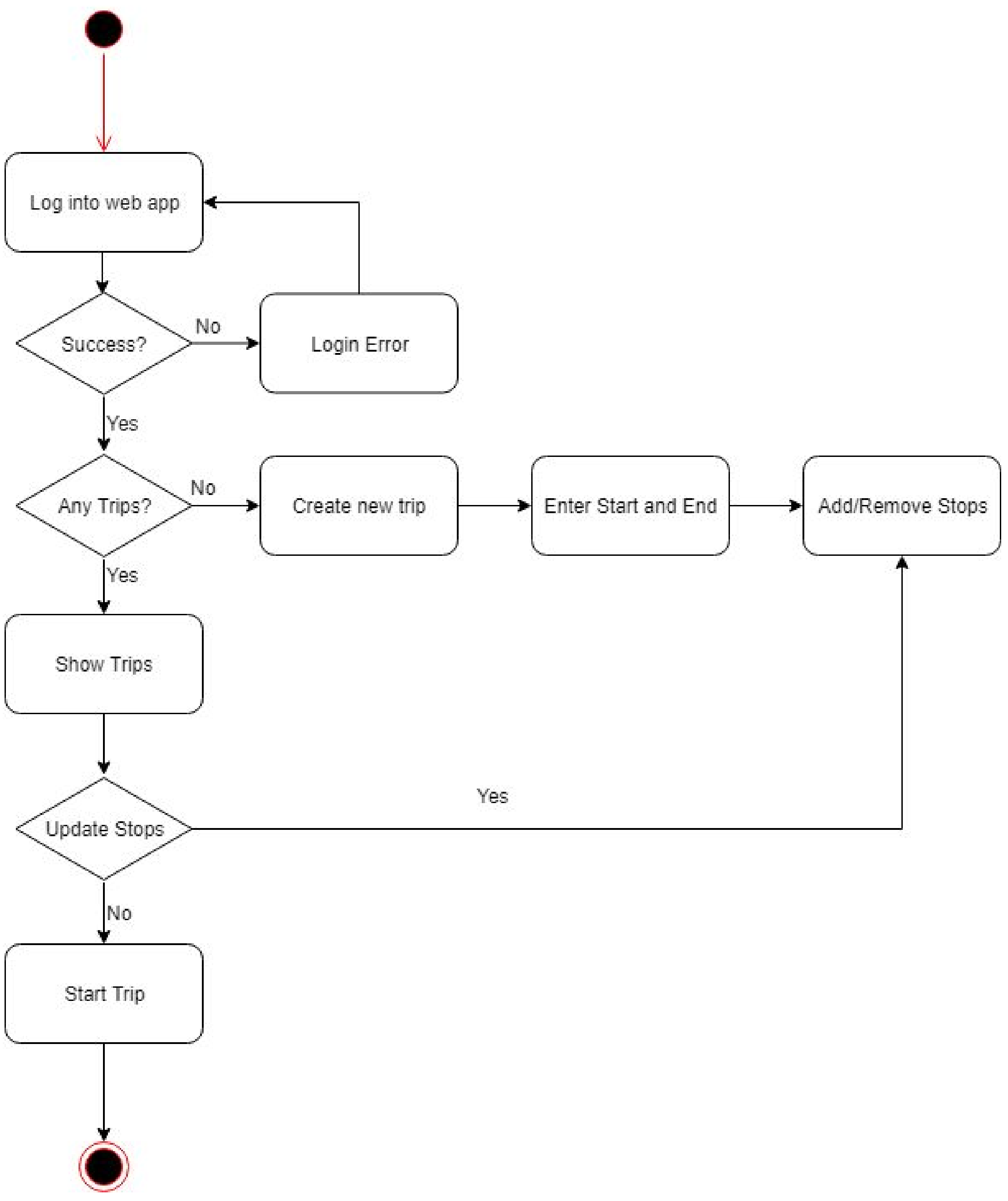
Use Case #6
Name – Budget options
Brief Description – User wishes to add or remove stops along the way which will increase the cost or lower it.
Actors – Trip, user, database

Preconditions – Trip endpoints have been chosen
Basic Flow – User adds a stop if they wish to increase their budget and spend money at a specific location otherwise remove stops and lower the budget.
Alternate Flows – User has a specific budget in mind and caters their stops based on that value.
Exception Flow – Some stops are mandatory such as refueling or resting.
Post Conditions – Trips become finalized and are associated with the user.

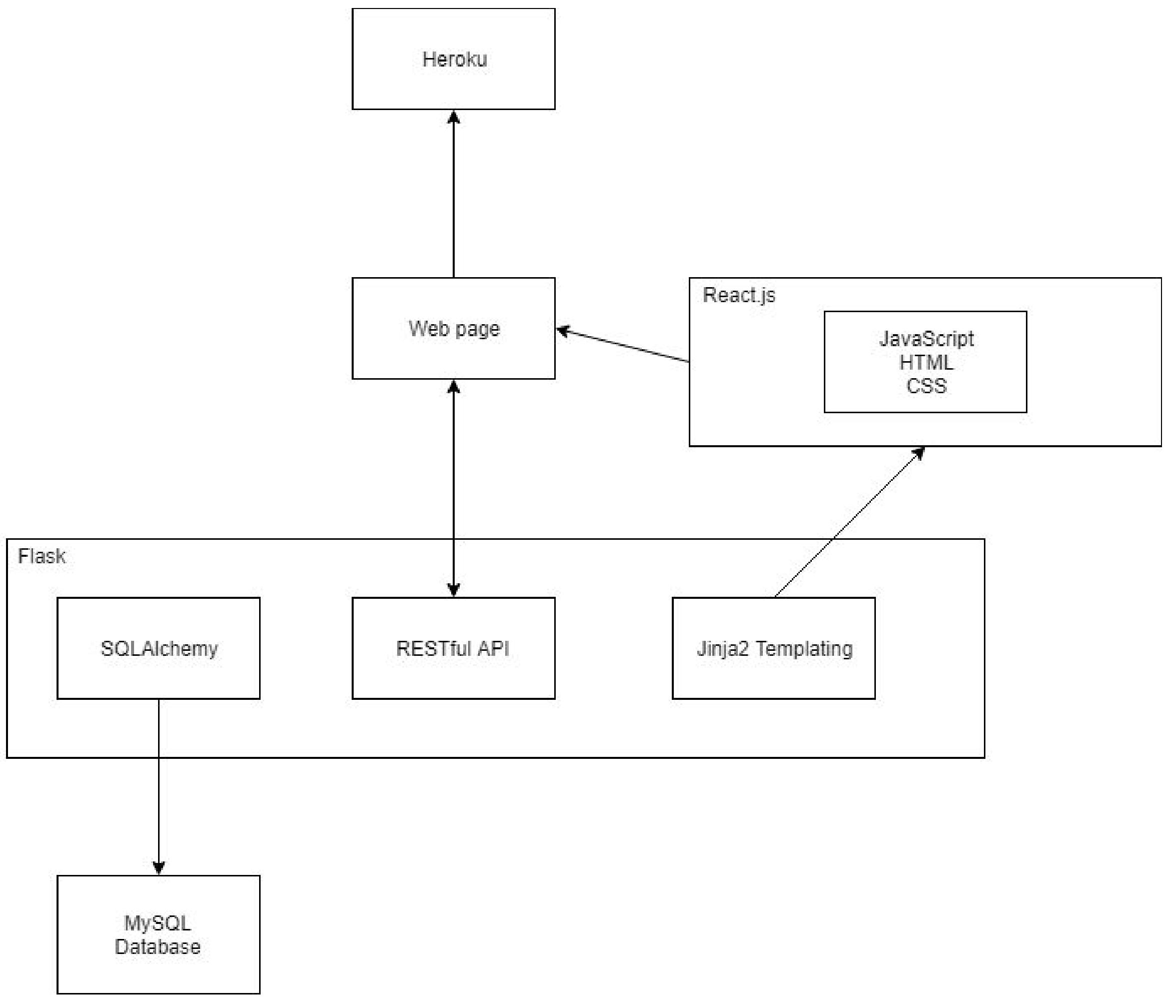
Use Case #7
Name – Budget Calculator
Brief Description – The user may calculate their trip costs based on what stops they have planned.
Actors – Trip, user, database
Preconditions – Trip endpoints have been chosen and stops have been added
Basic Flow – User adds stops which have costs associated with them which are added in a calculator.
Alternate Flows – The calculator has the option showing the cost of the route up to a certain point
Exception Flow – Some stops are fixed and cannot be removed
Post Conditions – User is updated with the cost of their trip

Activity Diagram:

Use Case #6 Budget Options



Architecture Diagram:

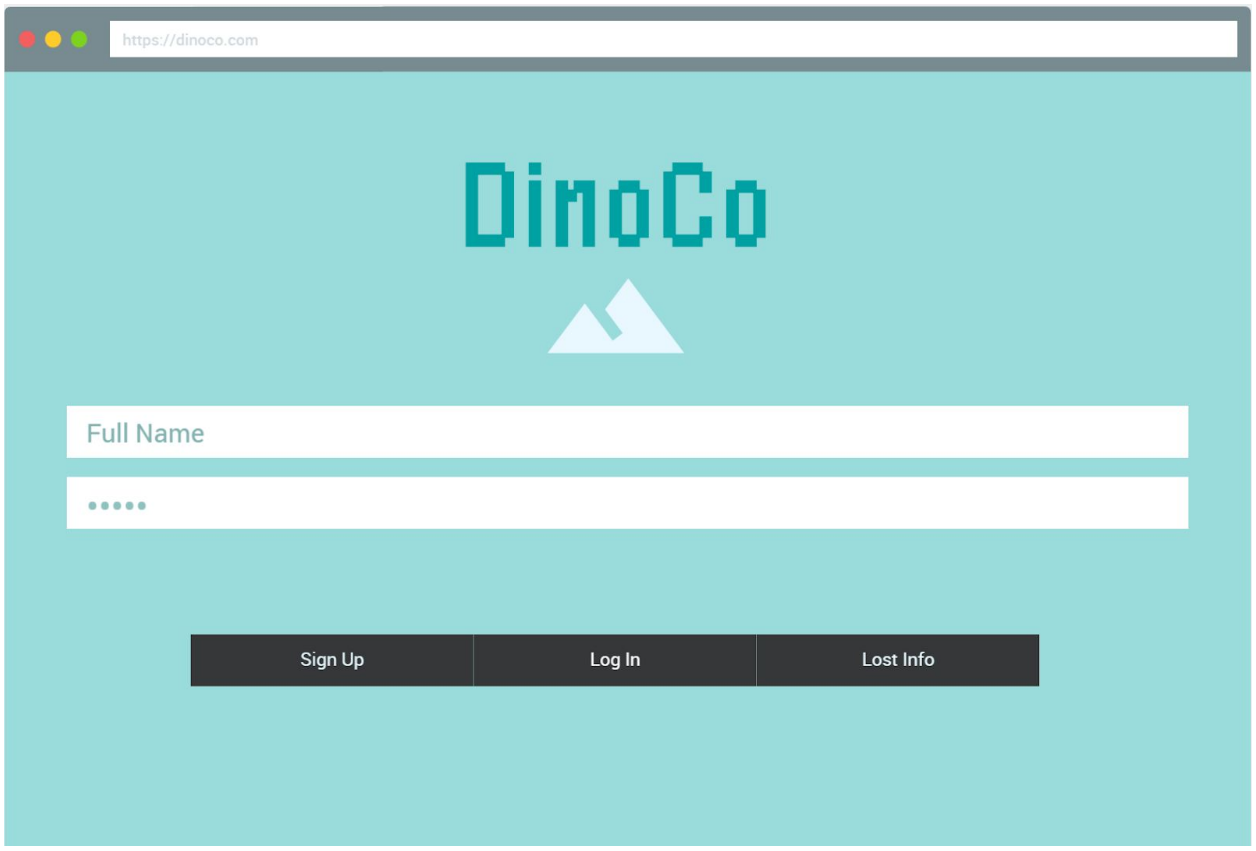


Our project is a web app that uses flask as its main backbone. The tools/frameworks will vary depending on the task. As our data storage we have a MySQL database, which we will be able to send and receive data from using SQLAlchemy ORM. We will use flask's RESTful API to send data to a web page and or to get user inputted data into our database. We will use Jinja2 templating within flask to render the UI, along with it we'll use react.js to make the components more modular and reusable within our different page designs. All of this is then hosted on Heroku.

Data Storage:

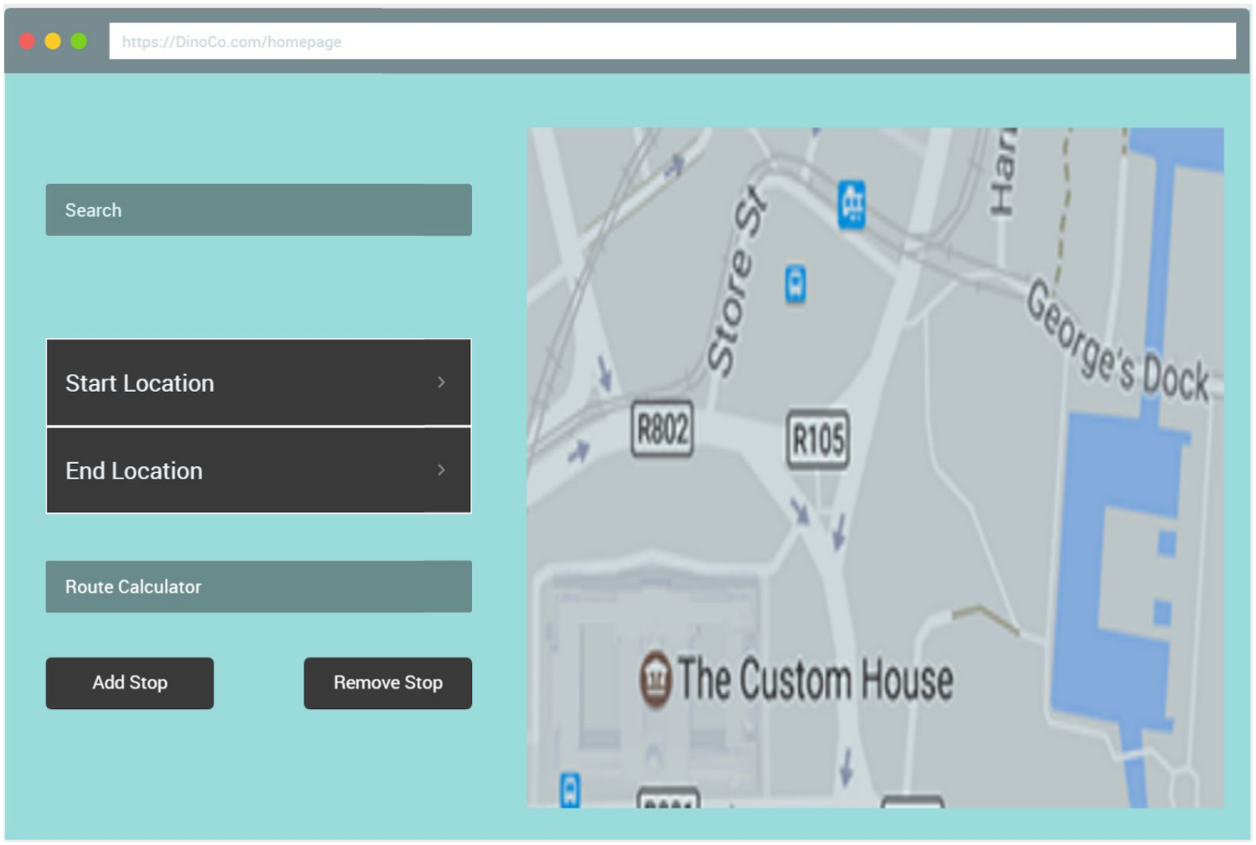
- We will be using MySQL as the data storage, some of the data we have will need to be transferred from a csv into a database. We will probably use a data controller class with SQLAlchemy as the ORM to manage data logic at run-time

UI Mockups:

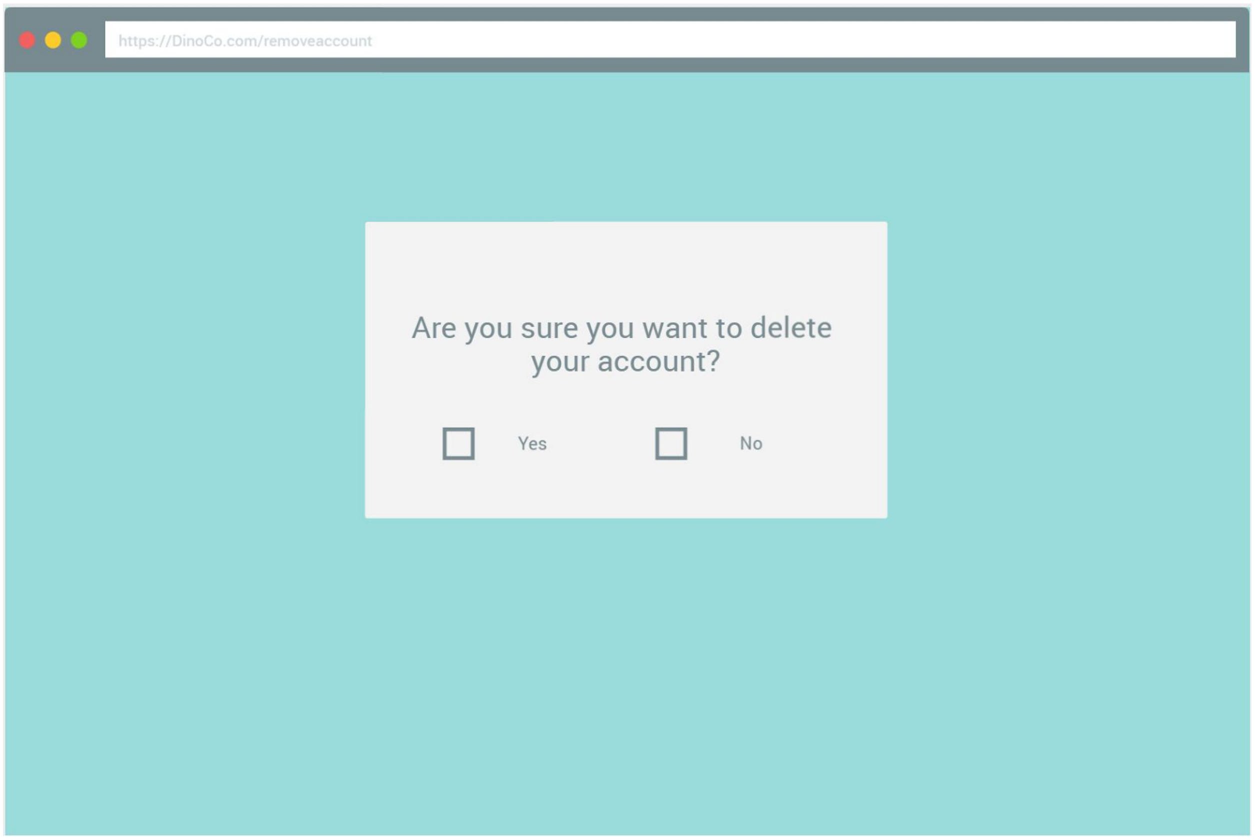


<u>User login page</u>

The user begins at the user login page where they are able to sign up for a new account, login to the web app, or retrieve their lost information. Each button takes the user to a different page corresponding to what the button says.



<u>Homepage for the web application</u>
The home page for this web app consists of a geographical map and various other functions. A user is able to search for different locations throughout the United States to scout out their trips. This information will reflect back on the map along with any other relevant information. The user may also add a start location and an end location for their trip and then have the budget display on the page. Users may also add stops and remove stops which will update the map as well as the budget.

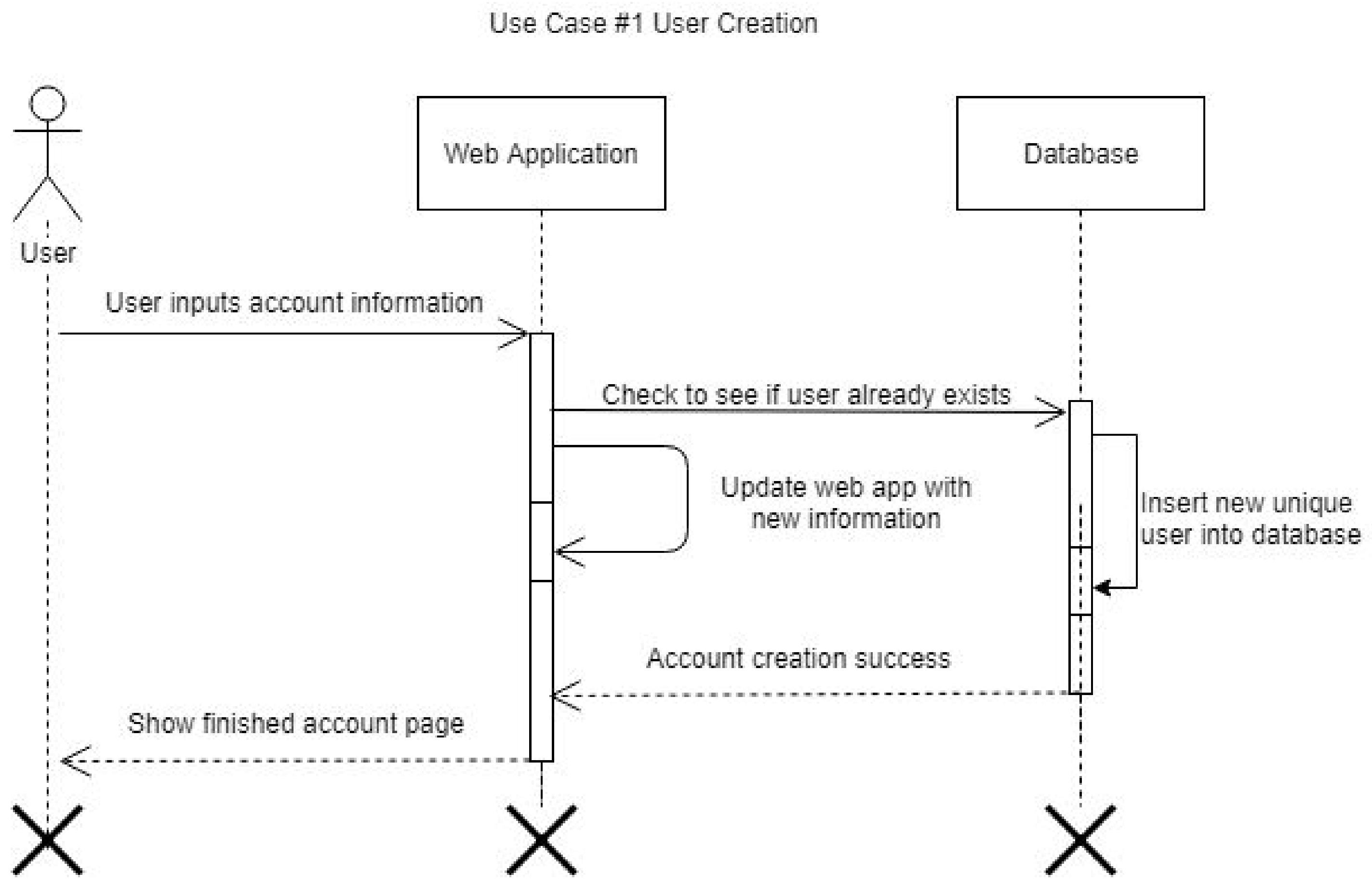

https://DinoCo.com/removeaccount
Are you sure you want to delete your account?
Yes
No

Screen for removing a user account
The web app provides a fail safe in case a user accidently deletes their account which comes in the form of a dialogue box. If the user is certain of deleting their account then they may complete the box and press yes.

User Interactions:

User interaction 1: Making an account

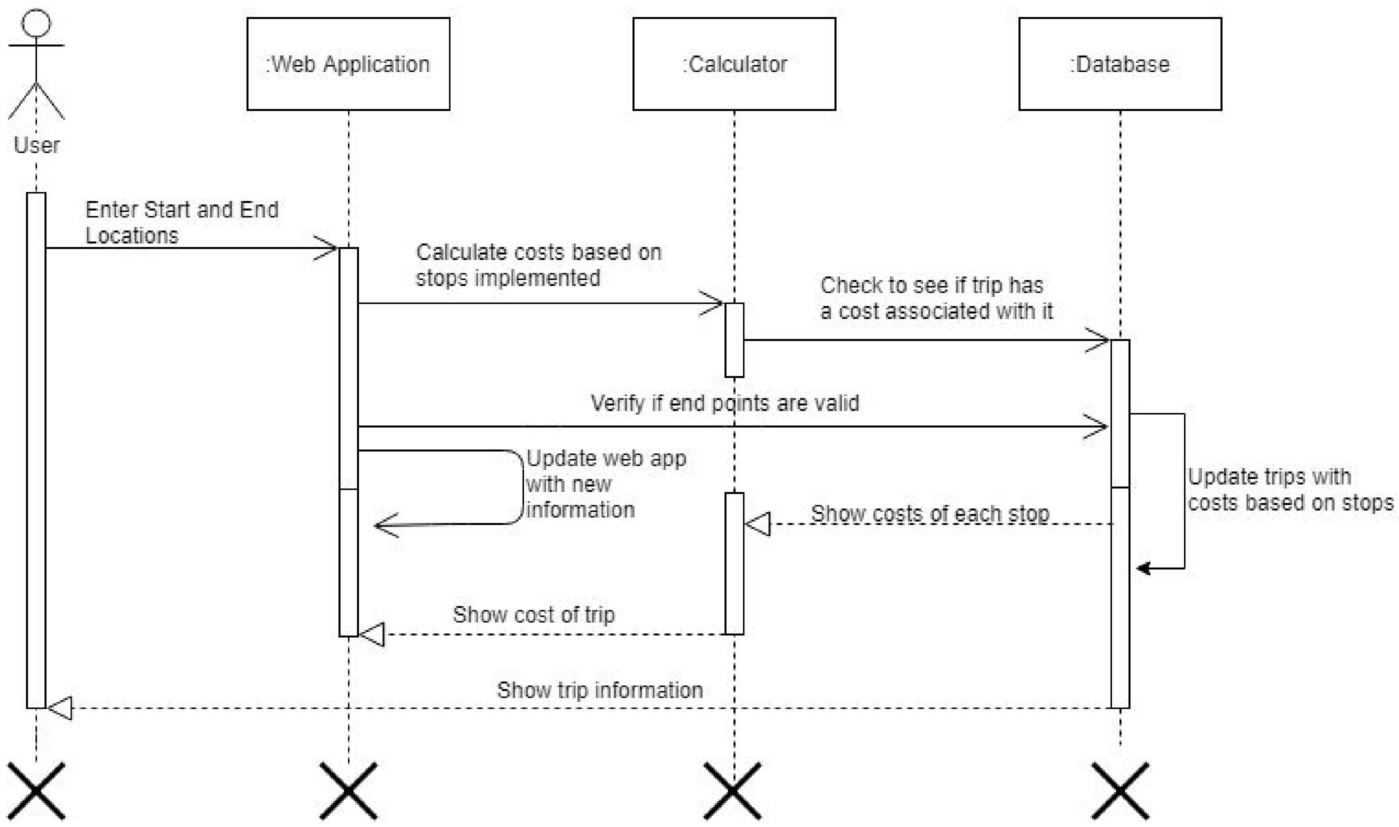
Description - In order to make an account the user must enter their information into a webpage which then gets stored into the webpage's database.


Use Case #1 User Creation
Web Application
Database
User
User inputs account information
Check to see if user already exists
Update web app with new information
Insert new unique user into database
Account creation success
Show finished account page

User Interaction 2: Creating a trip

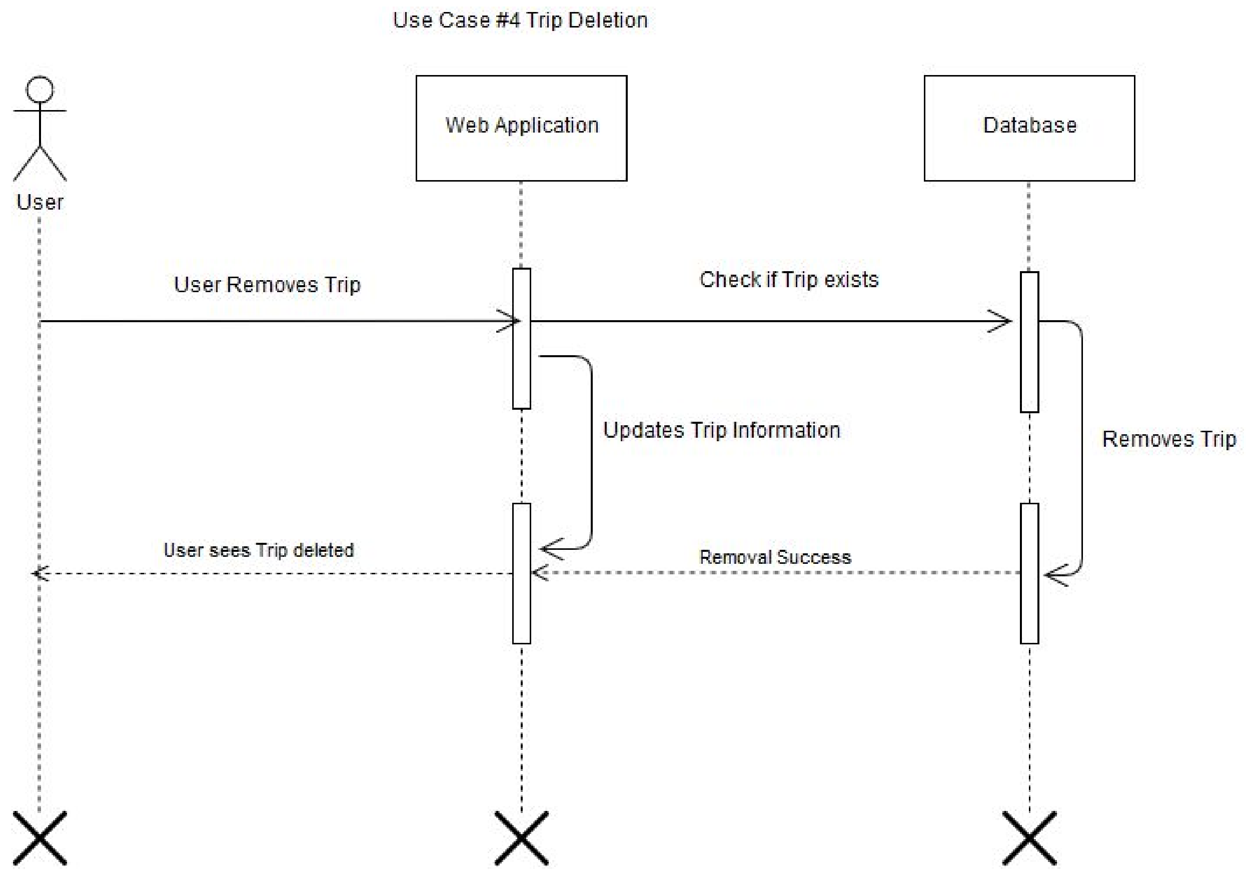
Description - A user enters a start and end location into the webpage which is verified through the database and then returns the cost of the trip after it goes through a calculator.

Use Case #7 Budget Calculator

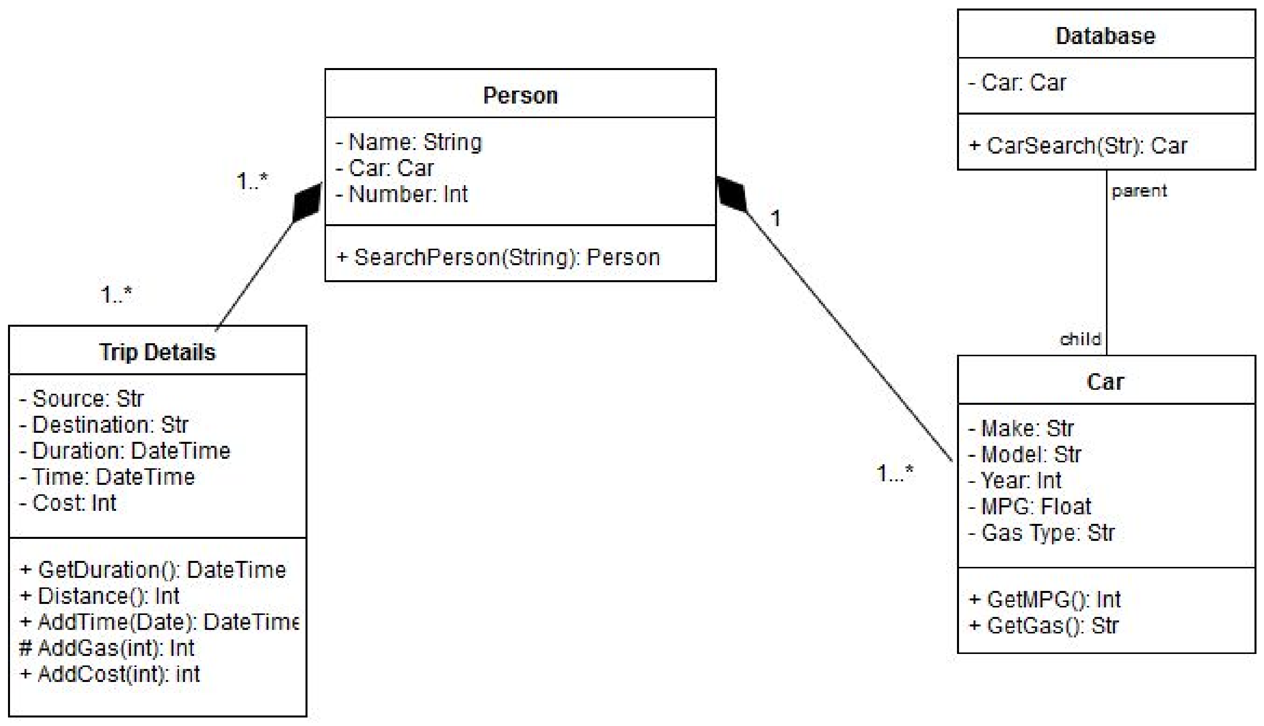

:Web Application
:Calculator
:Database
User
Enter Start and End Locations
Calculate costs based on stops implemented
Check to see if trip has a cost associated with it
Verify if end points are valid
Update web app with new information
Update trips with costs based on stops
Show costs of each stop
Show cost of trip
Show trip information

User Interaction 3: Trip Deletion

Description - When deleting a trip the user must input that they want to delete a trip. Information is then transferred from the web app to the database where the task is executed. Web app will then update with new information later visible for the user.



Class Diagram:


Person
- Name: String
- Car: Car
- Number: Int
+ SearchPerson(String): Person
Database
- Car: Car
+ CarSearch(Str): Car
parent
child
Trip Details
- Source: Str
- Destination: Str
- Duration: DateTime
- Time: DateTime
- Cost: Int
+ GetDuration(): DateTime
+ Distance(): Int
+ AddTime(Date): DateTime
# AddGas(int): Int
+ AddCost(int): int
Car
- Make: Str
- Model: Str
- Year: Int
- MPG: Float
- Gas Type: Str
+ GetMPG(): Int
+ GetGas(): Str
1..*
1..*
1
1...*