Cyclomaniacs Design Documentation

Critical Personnel

Name	Main Role
Alexander Webber	Team Advisor
Colin Adams	Back End Developer
Xavier Adams-Stewart	Spokesperson and Editor
Norland Batista	Back End Developer
Nicholas Miller	Front End Developer
Austin Sullivan	Front End Developer

Project Overview

1. Description

Our project is a web application that will compile all the prices of a given bicycle or bicycle accessory offered by major retailers in a user's area. Prices will come from the website of online and in-person vendors. Users should be redirected to the address on the website upon clicking a link.

Ultimate Application Goal: Compile and display user requested bicycle products and accessories from a set number of online retailers.

2. User Stories

In order to determine some of the features of our application, we created User Stories to determine what a user would want from our application. The stories were created by our team members and corroborated by our advisor.

- As a user, I will be able to easily navigate to the application.
- As a user, I will be able to click on a link or button that will take me to a product's webpage.
- As a user, I will be able to sort lists based on list categories.

- As a user, I will be able to search for an item.
- As a user, I will receive results that are available online or near my location.
- As a user, I will be able to select which retailers show up in my search results.
- As a user, I will be able to select which brands show up in my search results.
- As a user, I will be able to see the price of products.
- As a user, I will be able to set a price range for a search.
- As a user, I will be able to choose the search location.

3. Functional Requirements

With the help of our User Stories, we were able to determine the features that are absolutely required for the application to function. Failure to consider and implement any of these requirements means that the application will fail to function in the intended way.

- Search function for products
- Search function for categories
- Clear display of links to product pages
- Clear display of the price of products from product pages
- Search Database of major retailers

4. Non-Functional Requirements

With the help of our User Stories, we were able to determine the features that are not required for the application to function, but would improve user experience. Failure to consider and implement any of these requirements will not affect the intended function of the application but may negatively affect user experience. User experience is critical to the success of the application.

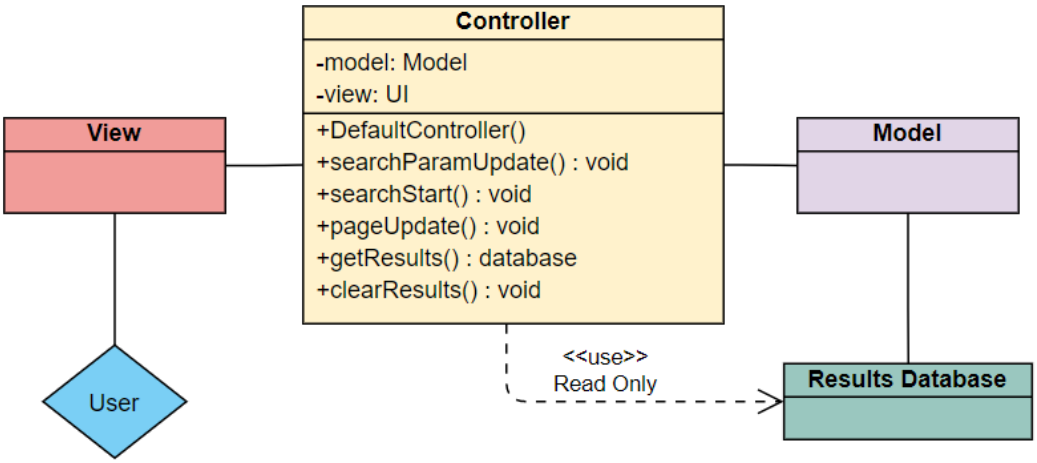
- Category sorting capabilities
- Retailer specification abilities
- Price specification abilities
- Location specification abilities
- Informative UI
- Simple UI
- Access to location data
- Search function based on user location

Design

---

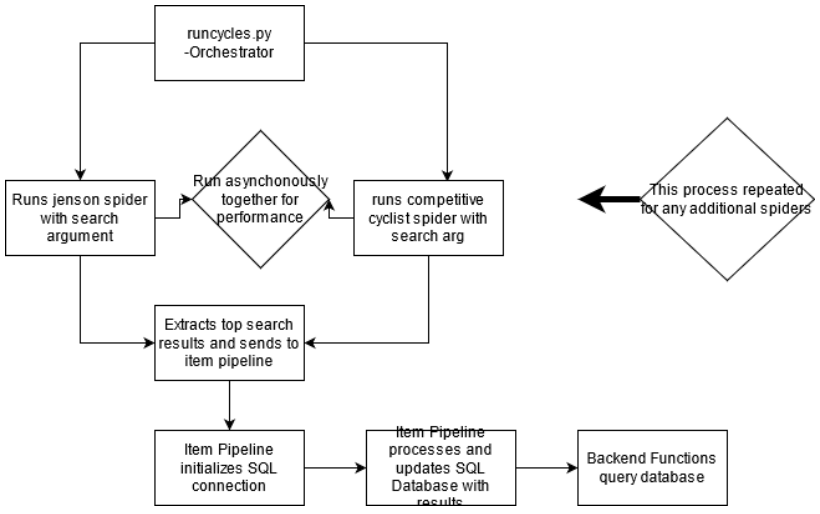
https://online.visual-paradigm.com/drive/#diagramlist:proj=0&new for diagram designs

NOTE: All design aspects were subject to change throughout the course of our application.
We applied the MVC framework to our application idea. Our initial framework was as follows:



MODEL:

- General scraping algorithm for all chosen websites
- Consideration of IP API for location (Non-functional requirement)
- The algorithms will scrape website data into a data frame
    - Export the frame to database
- Model is responsible for database operations



Keyword search:
We decided that we will use the search urls for our chosen set of websites, and add the search term that the user inputs in our website.
Search urls for major websites (Search term = "mountain bike"):

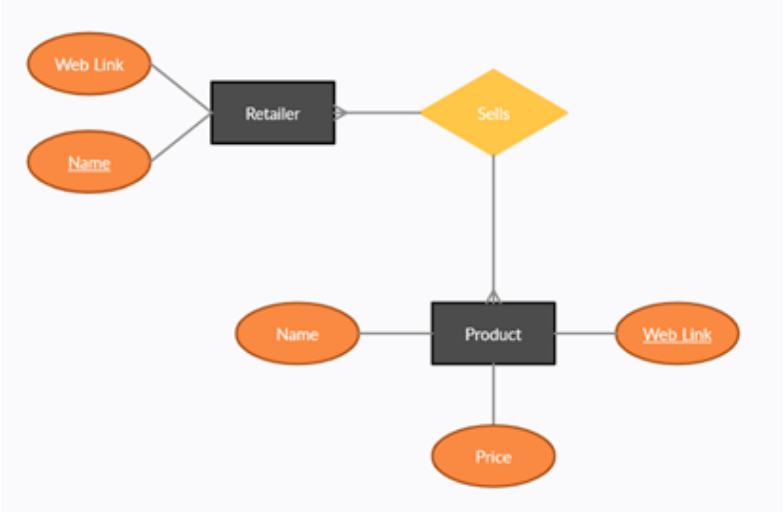
https://www.chainreactioncycles.com/us/en/s?q=mountain+bike
https://www.jensonusa.com/search?q=black+helmet
https://www.bikebling.com/SearchResults.asp?Search=mountain+bike
https://www.nashbar.com/search?s=black+helmet
https://www.performancebike.com/search?s=mountain+bike
https://www.competitivecyclist.com/Store/catalog/search.jsp?s=u&q=black+helmet
https://www.backcountry.com/Store/catalog/search.jsp?s=u&q=mountain+bike
https://www.rei.com/search?q=black+helmet

CONTROLLER:

- The buffer between the View, Model, and Database
- Search terms sent from the view are relayed to the model
- Parameter/attribute adjustments made in the view are relayed to the model (price/name/website based search)
- Controller has read-only access to database
    - That access allows it to ensure that a database is connected
- Controller language = JavaScript or Python

RESULTS DATABASE:

E/R diagram



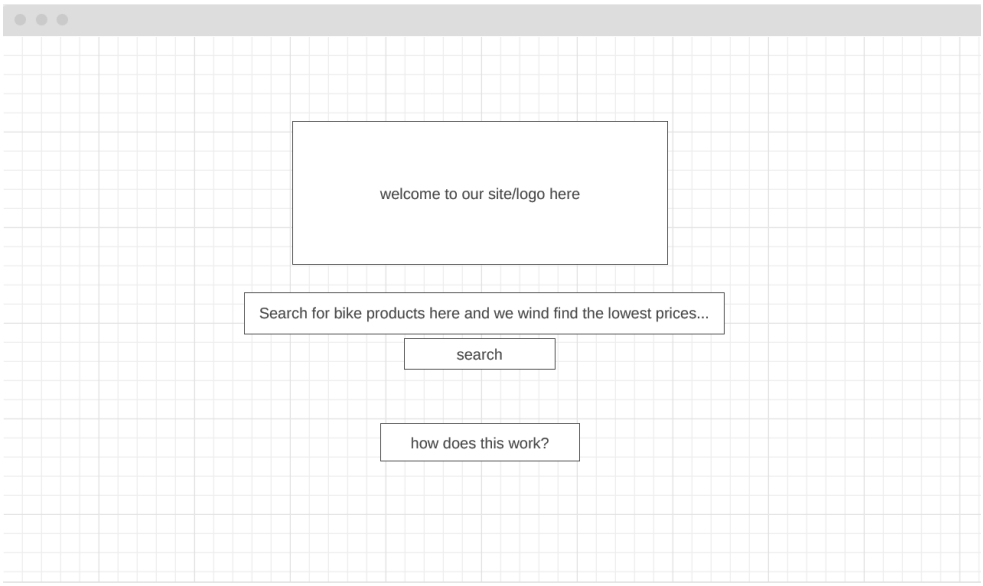
Two entities are contained within our database: the Retailer entity and Product entity. The Retailer entity has two relevant attributes, which is its name and website link. The name of a retailer is unique, which led us to choosing that attribute as the key attribute. The Product entity has three attributes: price, name, and website link. There can be products with the same name

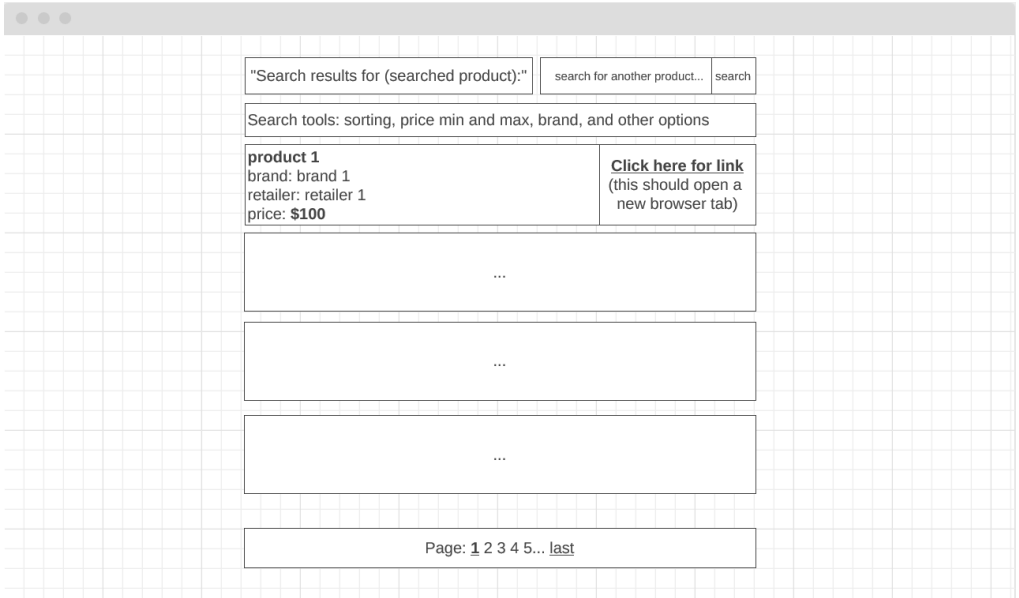
that can be sold by retailers, but each product's url is unique, which is perfect as an attribute. This database will be filled with data from the Model algorithms and its language will be PostgreSQL.

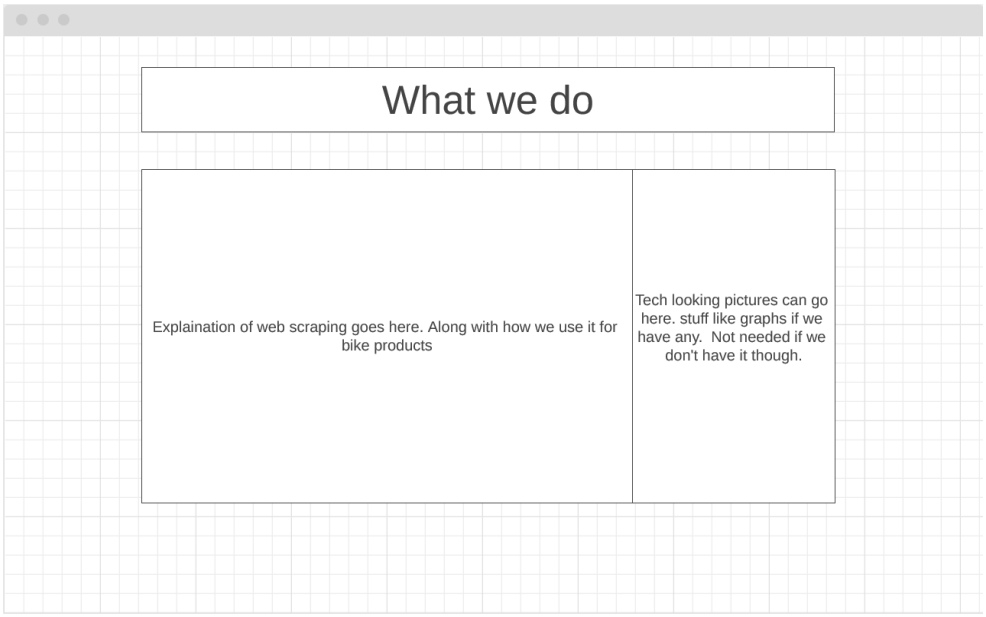
VIEW:

- React Javascript library
- Contains search bar and website description
- Potential search parameter buttons/checkboxes
- Results are listed with name/price/url link (picture if possible but not necessary)
    - Sorting capabilities

Wireframe:







MISCELLANEOUS:

Our tech stack also includes the Flask microframework to establish local servers to test and demo our application. Plans for general deployment are being discussed.