

# CS 530: Developing User Interfaces

## Assignment 2

### Goals

This assignment asks you to continue developing your web site for “Drexel Bikes”, specifically in adding several features to make it a more realistic site—receiving data from a live database and allowing the user to view and change this information.

### Assignment

The implementation of our “Drexel Bikes” web site for Assignment 1 built the basic styling and navigation bar for the site, and added a page “Our Bikes” where the user could see the list of available bikes. In this assignment, we will enhance this site to include a live SQLite database.

To begin, please take your site from Assignment 1 and copy it into a new directory, **a2**. Please note that, for this assignment, we will only be grading the parts relevant to the current assignment and not those from Assignment 1; thus, if you didn’t quite get everything working correctly in the first assignment, you don’t need to worry about that functionality—please just focus on the functionality for this assignment.

### Implementation Setup

To start this assignment, first please download the **bikes.csv** file that comes with the assignment. This file contains CSV data for the bike database, with the following fields (in this order):

- id – text (cannot be null)
- name – text (cannot be null)
- wheels – integer
- size -- integer
- motor – integer (0 or 1)
- folding – integer (0 or 1)
- image – text
- available – integer

The meaning of each field is the same as in Assignment 1.

Given this file, create a SQLite database **bikes.db** that contains the above fields as columns in the database, and import the data in **bikes.csv**. To do this, please create a script **create\_db.sh** with the command-line commands needed to create and populate the database, and put this script and the **bikes.csv** file in a folder **/a2/dev**. (This folder will not do anything during runtime; it simply allows us to see how you constructed your database.) At the end of this process, you should have a SQLite database **bikes.db** that will serve as the back-end data storage for your system.

## “Rent a Bike” Page

We will now create a page “Rent a Bike” that generates dynamic content, using JavaScript to populate the page table and handle events on the page. The end result should look as follows (only the top part of the page is shown):

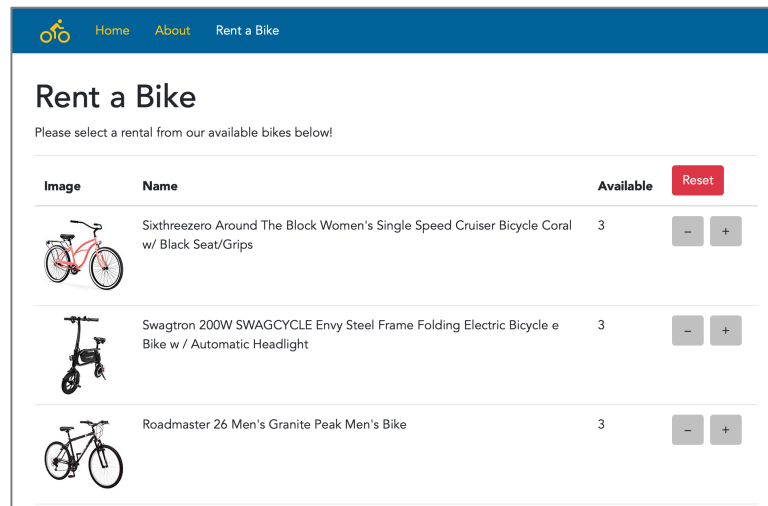





Image	Name	Available	Reset
	Sixthreezero Around The Block Women's Single Speed Cruiser Bicycle Coral w/ Black Seat/Grips	3	<input type="button" value="-"/> <input data-bbox="1070 593 1142 622" type="button" value="+"/>
	Swagtron 200W SWAGCYCLE Envy Steel Frame Folding Electric Bicycle e Bike w / Automatic Headlight	3	<input type="button" value="-"/> <input data-bbox="1070 689 1142 719" type="button" value="+"/>
	Roadmaster 26 Men's Granite Peak Men's Bike	3	<input type="button" value="-"/> <input data-bbox="1070 786 1142 815" type="button" value="+"/>

In this page, the user sees the number of bikes available for each bike. The user can click on ‘–’ to decrease this number and ‘+’ to increase this number. If the number of bikes reaches 0, the row should be grayed out, and the ‘–’ button should no longer work (so that the number cannot go below 0). A sample grayed-out row is shown below. If the user clicks the “Reset” button at the top of the table, the available number for *\*all\** bikes should be set to 3.

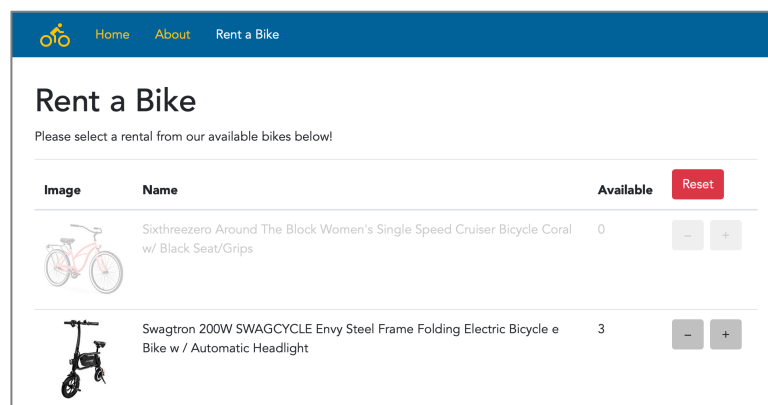




Image	Name	Available	Reset
	Sixthreezero Around The Block Women's Single Speed Cruiser Bicycle Coral w/ Black Seat/Grips	0	<input type="button" value="-"/> <input data-bbox="1070 1373 1142 1402" type="button" value="+"/>
	Swagtron 200W SWAGCYCLE Envy Steel Frame Folding Electric Bicycle e Bike w / Automatic Headlight	3	<input type="button" value="-"/> <input data-bbox="1070 1469 1142 1498" type="button" value="+"/>

Create a template **rent.html** that contains the basic header and content above the table, and modify your main navigation bar at the top of the page to include a link to the new “Rent a Bike” page.

Next, create a file **static/js/rentView.js** that builds the “view” for this page, namely the main table that includes all the bikes with their availability. We recommend creating a JavaScript object **RentView**, using **function RentView(..) { .. }** and then instantiating the object with **new RentView(..)**. In this object, you can implement the necessary functionality to: (1) load data from the server, (2) change the availability of a given bike, and (3) reset all bikes to have the same availability (namely 3).

For each of these functions, you will also need to implement the corresponding functionality on the server side in the **run.py** file. Specifically, you should define the following API URLs with associated AJAX functionality:

- **/api/get\_bikes** : a GET request that returns all bikes in the database;
- **/api/update\_bike** : a POST request with two arguments — a bike ID and a number available — that updates that particular bike to have that given number available;
- **/api/reset\_bikes** : a POST request with one argument — a number available — that resets all bikes that have that given number available.

(Recall that the functions that modify the database should be done using POST requests instead of GET requests.)

Also, please note that for the grayed-out rows, you should add a class **unavailable** to the table row, and then in your **main.css** file, give this class an opacity of 0.25. You will also need to separately disable the ‘–’ button, since graying out the row does not affect the functionality of the button.

Your final page should provide all the functionality of the buttons described above, and since you have a database back end, all data should remain the same even after reloading the page.

## Documentation

It is expected that all code written for this assignment is properly commented where needed, especially in places where you have made particular choices about data structures and/or algorithms to employ. Also, please add the following identification header to every file you create:

```
# Your Name, Your Email  
# CS530: DUI, Assignment [#]
```

(or whatever commenting syntax is appropriate for the file at hand).

## Submission

Please submit your files as an attachment for the assignment on Blackboard Learn. Please use a compression utility to compress your files into a single ZIP file (not RAR or any other compression format). The final ZIP file must be submitted electronically using Blackboard—do not email your assignment to a TA or instructor! If you are having difficulty with your Blackboard account, you are responsible for resolving these problems with a TA or someone from IRT before the assignment is due. If you have any doubts, complete your work early so that someone can help you.