Christopher Brown & Angie Well

Automatic Form Generation Wizard

Associates Capstone – Summer 2018

Table of Contents

[Version History 2](#_Toc522633098)

[Brief Overview - Strategy 3](#_Toc522633099)

[Key Features 3](#_Toc522633100)

[Limitations 4](#_Toc522633101)

[Technologies 5](#_Toc522633102)

[User Concepts 5](#_Toc522633103)

[Pre-User Testing 7](#_Toc522633104)

[Version 0.1 Mock-Up Designs 8](#_Toc522633105)

# Version History

|  |  |  |
| --- | --- | --- |
| **Date** | **Author** | **Description** |
| **7/18/18** | **Chris** | **Document Setup** |
| **7/21/18** | **Chris** | **Added key features, limitations, constructed website mockup in marvel,** |
| **7/23/18** | **Chris** | **Added Technologies, Users Concept** |
| **7/23/18** | **Angie** | **Added flowchart and mockup pictures.** |
| **7/27/18** | **Chris & Angie** | **Added application interactivity breakdown** |
| **8/15/18** | **Angie** | **Worked on changes in the document** |
| **8/20/18** | **Angie** | **Changes to the application roadmap** |
| **8/20/18** | **Chris & Angie** | **Changes to interactivity breakdown** |

# Brief Overview - Strategy

#### What is it?

The Automatic Form Generation Wizard is a web application that allows users to create, manage, and track custom printable web forms.

#### Who is it for?

This application is built for casual to intermediate computer users who need a quick and clean way to generate and track forms. The design will be tailored as simplistic, minimalistic, and straightforward so casual computer users won’t get lost within the application.

#### Why is it being developed?

We are developing this application because rehashing previously written web form code can be tedious and we wanted to provide the users with a tool they can cleanly and quickly use to track information without the need of consulting a software developer.

# Key Features

* User login and registration system
* Clean and robust custom form creation system
  + Ask multiple questions with several different types of user inputs:
    - Single Line
    - Paragraph (long text in SQL)
    - Multiple Choice
    - Checkboxes
    - Date/Time
  + Allow the user to dictate what fields are mandatory for completion
* Generation of a sharable URL for guest users to fill out the users forms
* A tracking system to pool and maintain accurate records of the form’s results
  + The ability to alter completed forms
  + The ability to delete form instances
  + The ability to delete a generated form and all of its instances
* The ability to print out the generated forms in a coherent layout

# Limitations

#### Time

This project has a tiny development cycle at ten weeks, so coding time is limited and some initially intended functionality may be lost.

#### Heavy Back-End

This application requires robust SQL table generation and management in order to be scalable. Due to the necessity of this for the basic implementation of this application, most of the time spent in the development phase will be focused on making sure the database logic is clean and efficient.

#### Tracking System Flaws

As of 7/21/2018, the envisioned tracking system can potentially have a visible flaw in its ability to provide the user with enough information to distinguish instances of completed forms from one another beyond the usage of date/time. Using an “anchor” set of questions or forcing every form to begin with a full name field are possible solutions to this flaw.

#### Limited Platforms

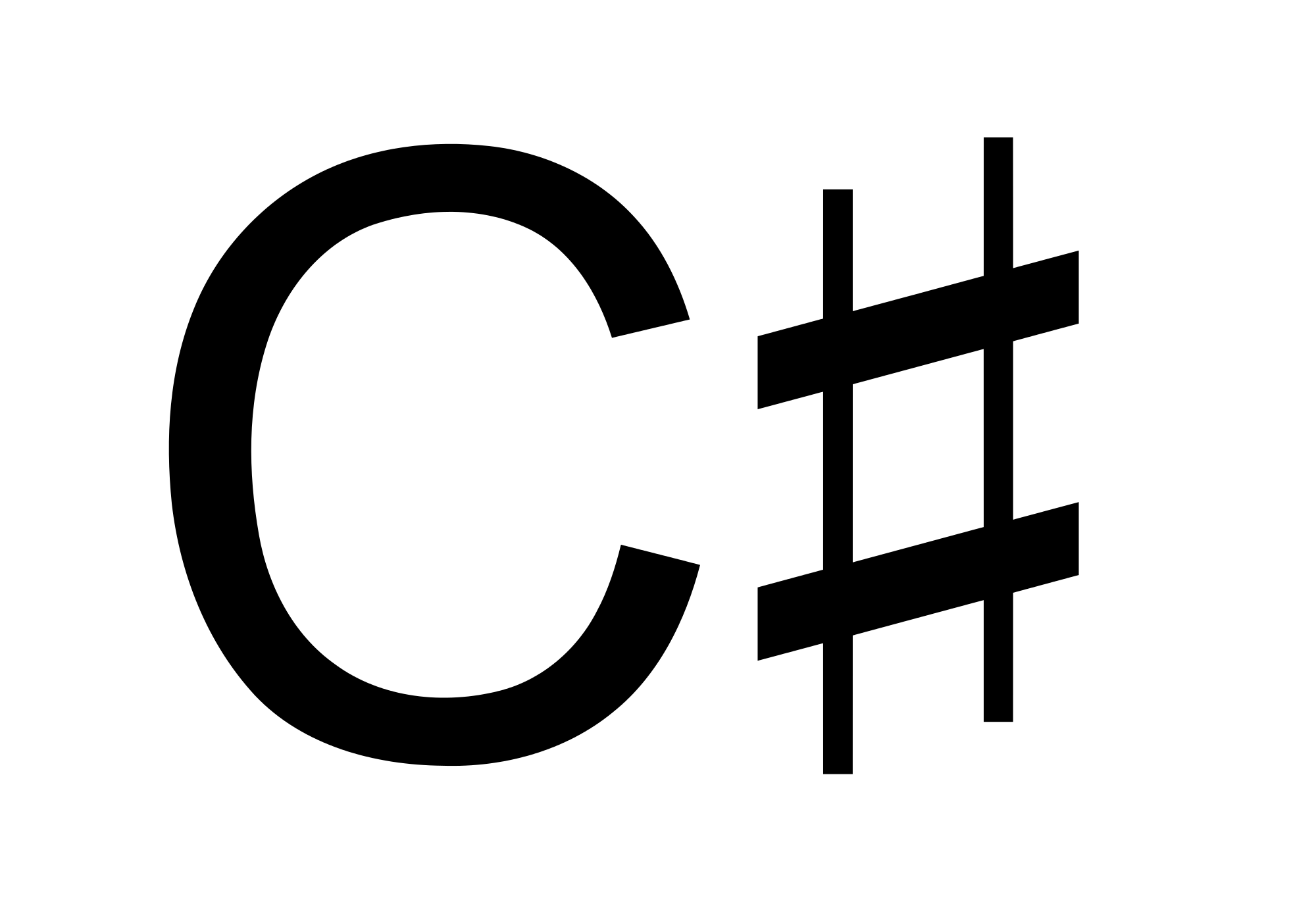
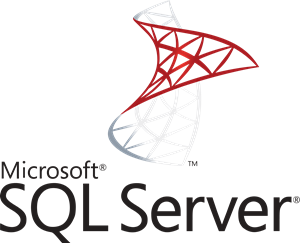
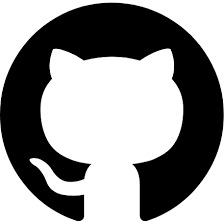
As of 7/21/2018, the intended platform for the usage of this application will be a desktop/laptop computer. Bootstrap or a similar CSS library is intended to be used as the main styling agent for this application, so if coded correctly, hypothetically it will be responsive enough for use on a tablet/phone device without having to specifically code it for that purpose. However, due to the small amount of time for development, multi-platform usage is not a main goal of the project.

#### Custom form validation is complicated

Coding form validation can sometimes be an easy endeavor, but due to the nature and usage of this application, coding custom form validation for users to implement is a very heavy feature that could easily take ten weeks of development on its own. The end goal is to allow the user to force a field to contain some kind of input (no empty response). Anything implemented beyond this scope will be an after-thought of this original design.

# Technologies



#### Application Navigation Roadmap



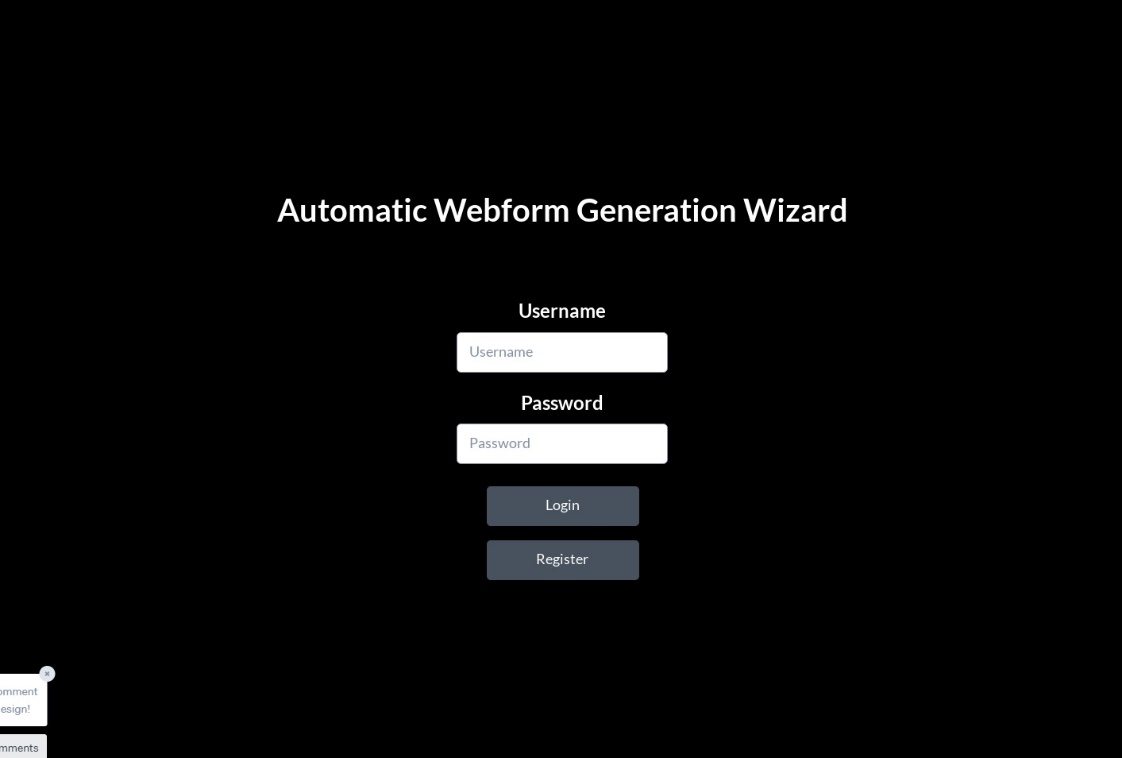
# Version 0.1 Mock-Up Designs

#### Registration Page

#### registration

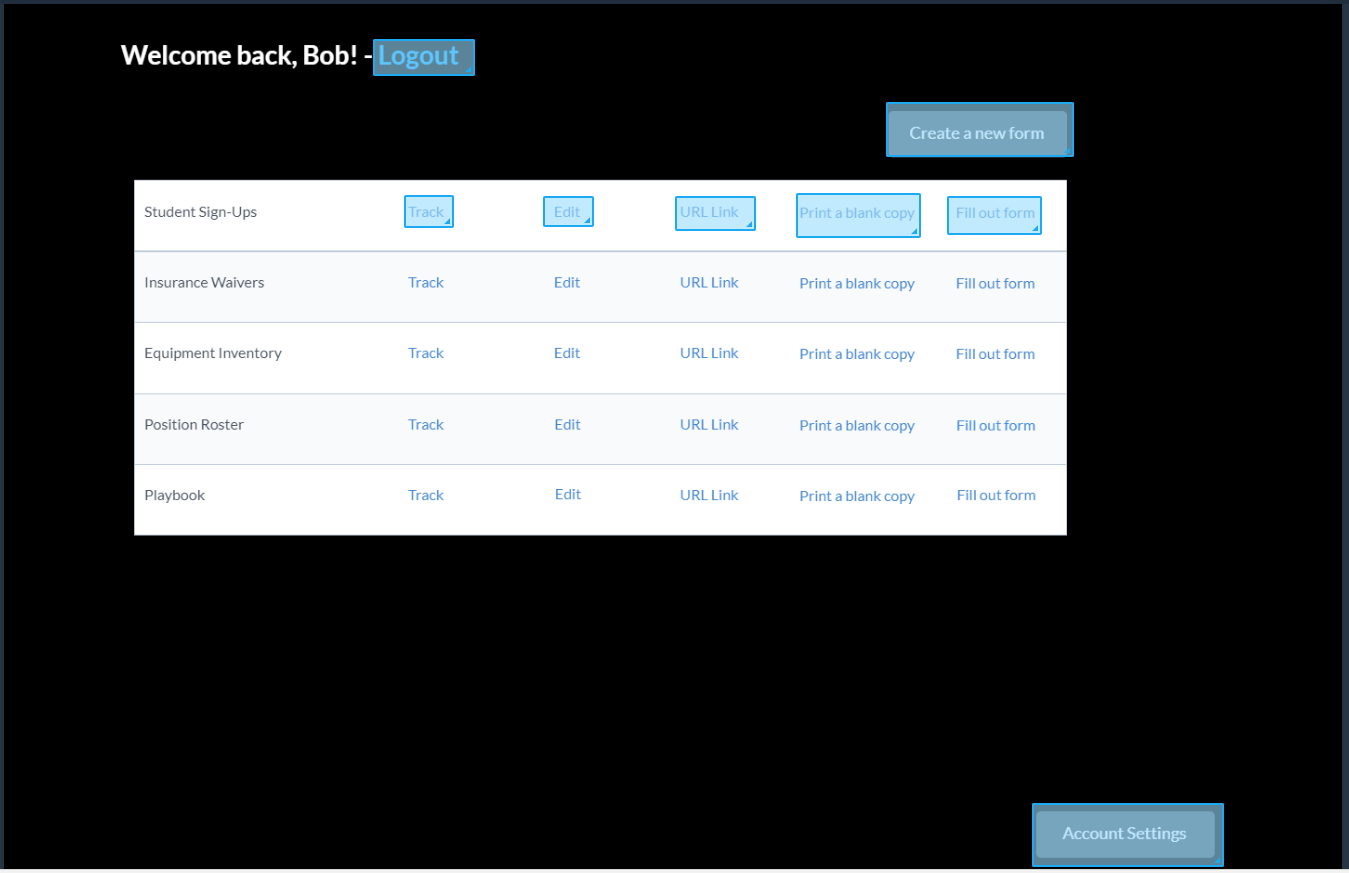
* Users will be able to fill out a simple registration form to register as a user

#### Login Page



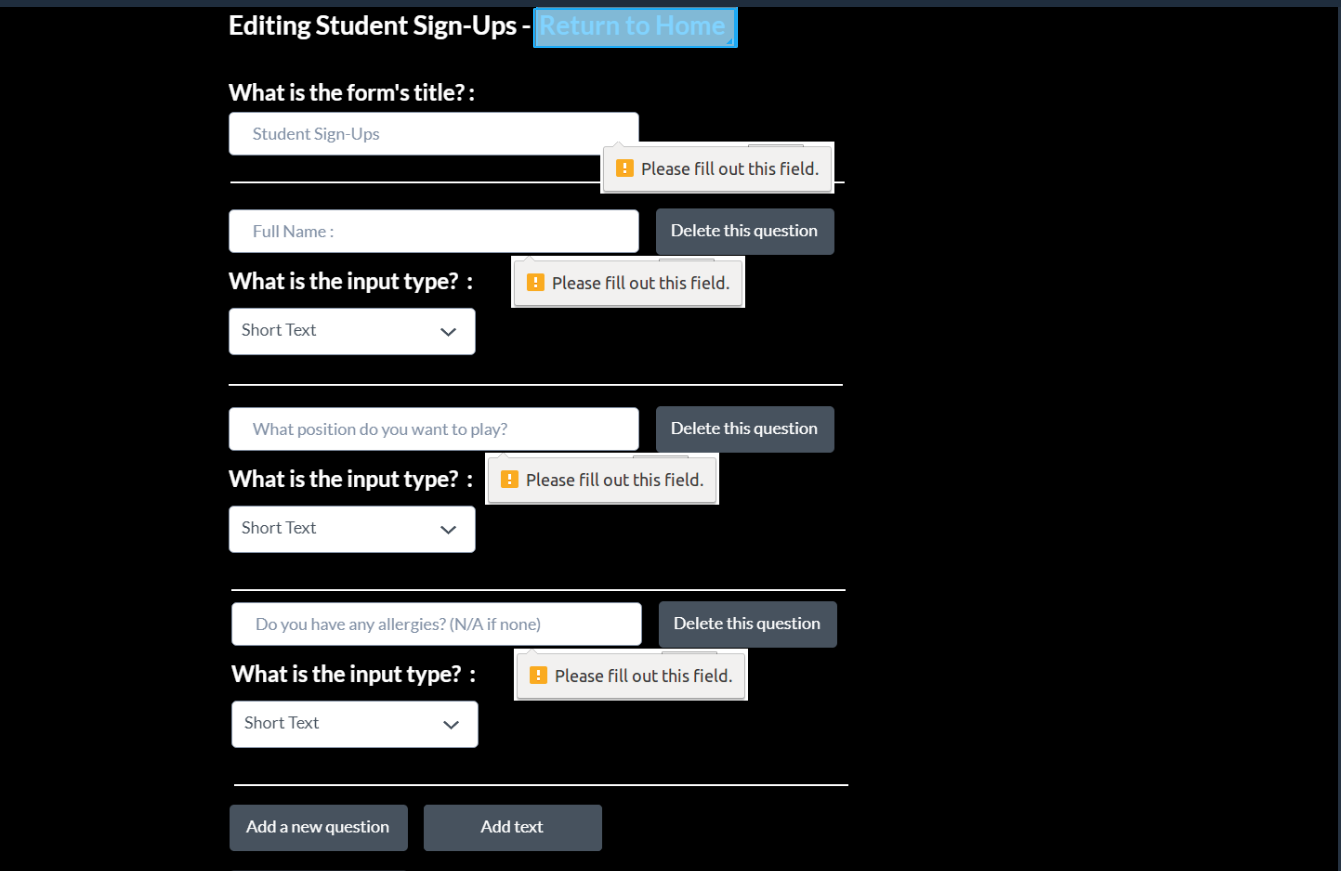
* Once registered, users will be able to login and proceed to their own homepage

#### Homepage



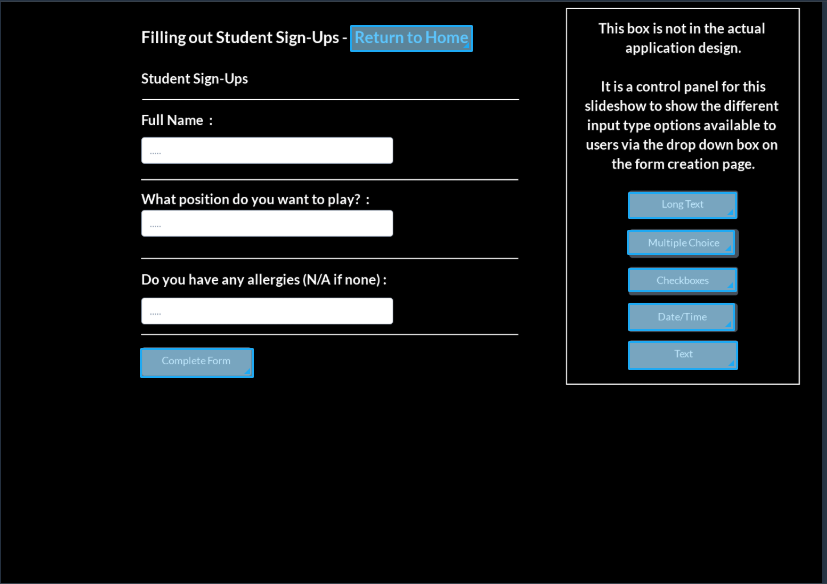
* At the top, the user’s username will appear and beside it a link to log out and return to the login page
* Underneath that will be a table whose contents will be generated based on the amount of forms the user has created. If nothing is created, the table will not be drawn and instead a message asking the user to create a new form.
* Inside the table, each generated form will have its own row. Within its row will have five working links:
  + Tracking homepage
  + Edit the form structure
  + Generate a URL link so another person can fill the form out
  + Print a blank copy of the form – this will generate a printable version of the generated form with writable inputs for the different input types.
  + Fill out the form natively
* Underneath the table will be a button that leads to the form creation page.

#### Edit Existing Form



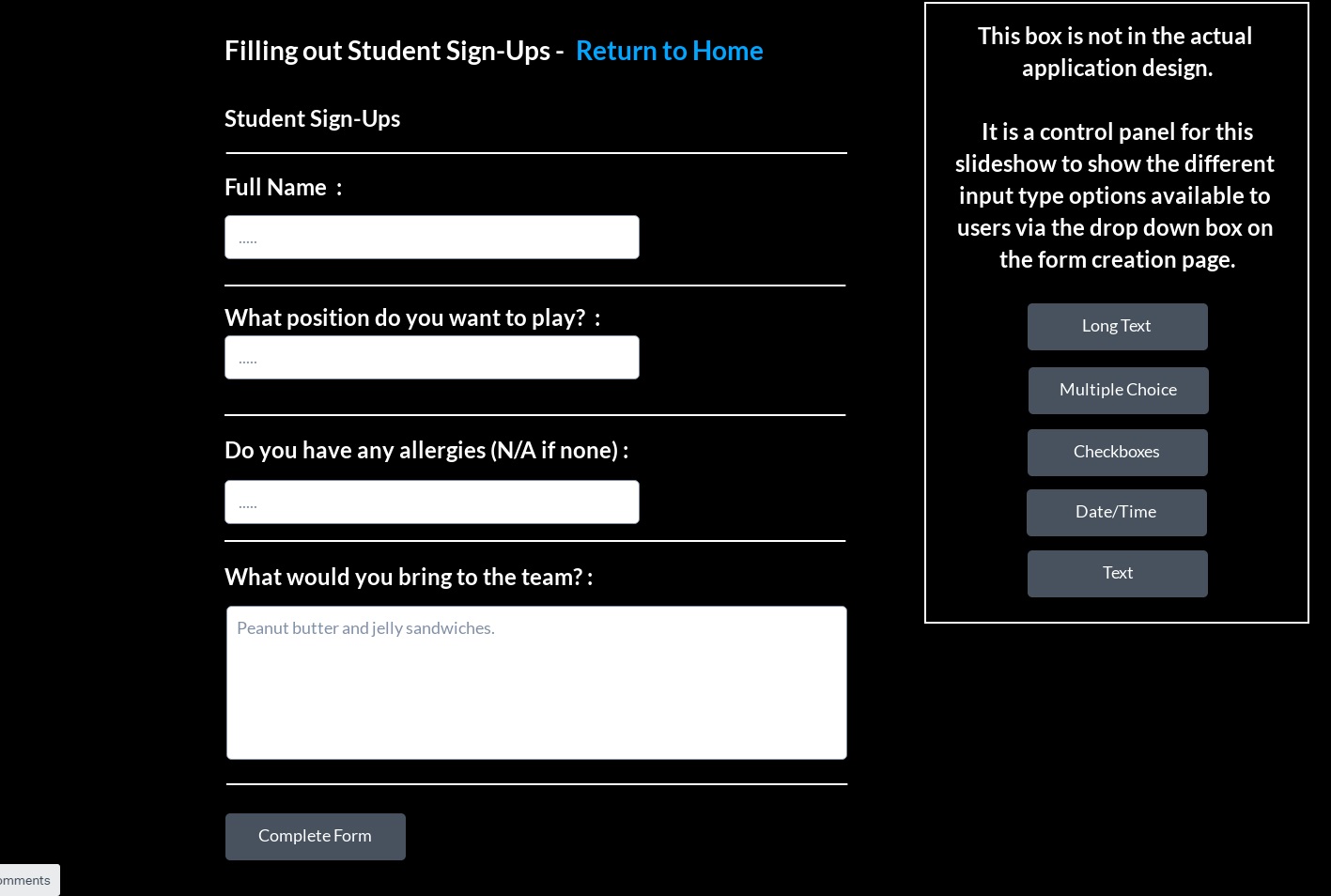
* The edit existing form will operate very similarly to the form creation pages, the only difference will be that it be automatically generated with the form questions previously built and instead of generating new tables, and it will alter the tables and scrub any fields not being tracked anymore. Possibly will add warning text so the user understands the impact of deleting form questions.

#### Fill-out Existing Form Natively / Via Generated URL Link / Single Line Example



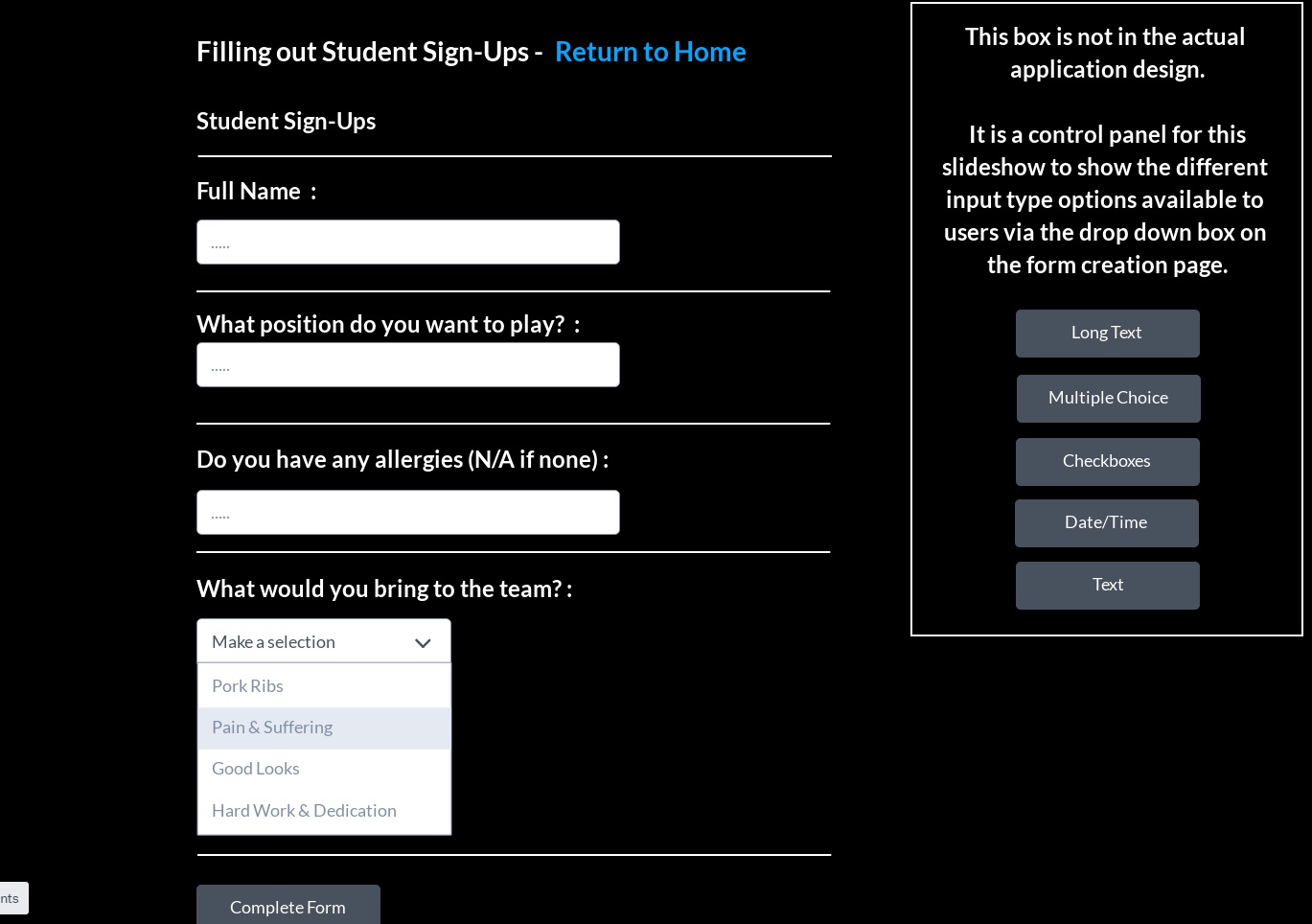
* At the top, the form title will be displayed. Every field the user has created in their form is generated as a question. Any static text the user generates will be displayed in their slot of the question pool. At the bottom of all the entries will be a complete form button that will insert the data in the inputs into their appropriate tables as another entry. There are several different input types:
  + Single line will generate a single line text box

#### Fill-Out Long Text Example



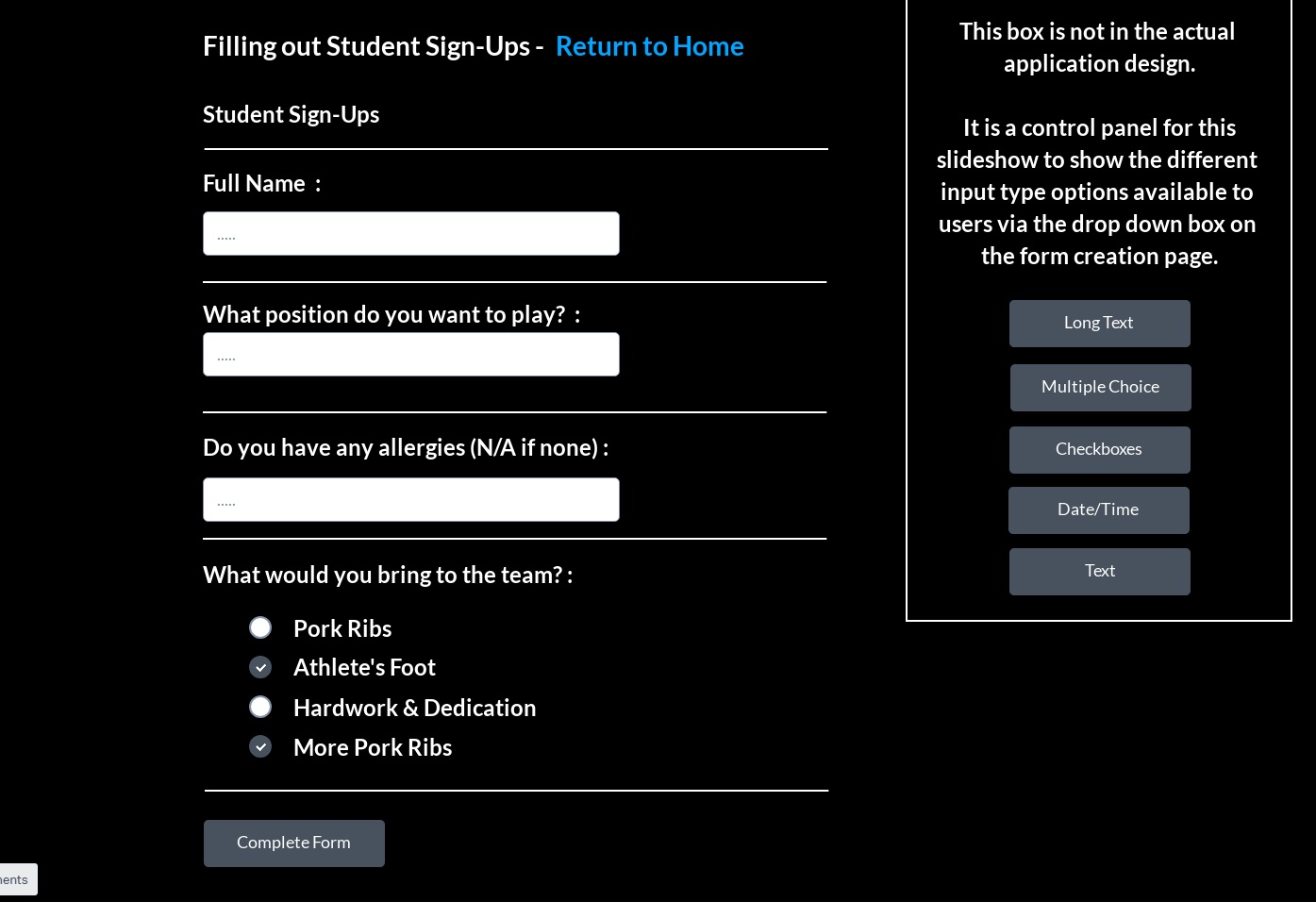
* + Long text will generate a multiple line text box

#### Fill-out Multiple Choice Example



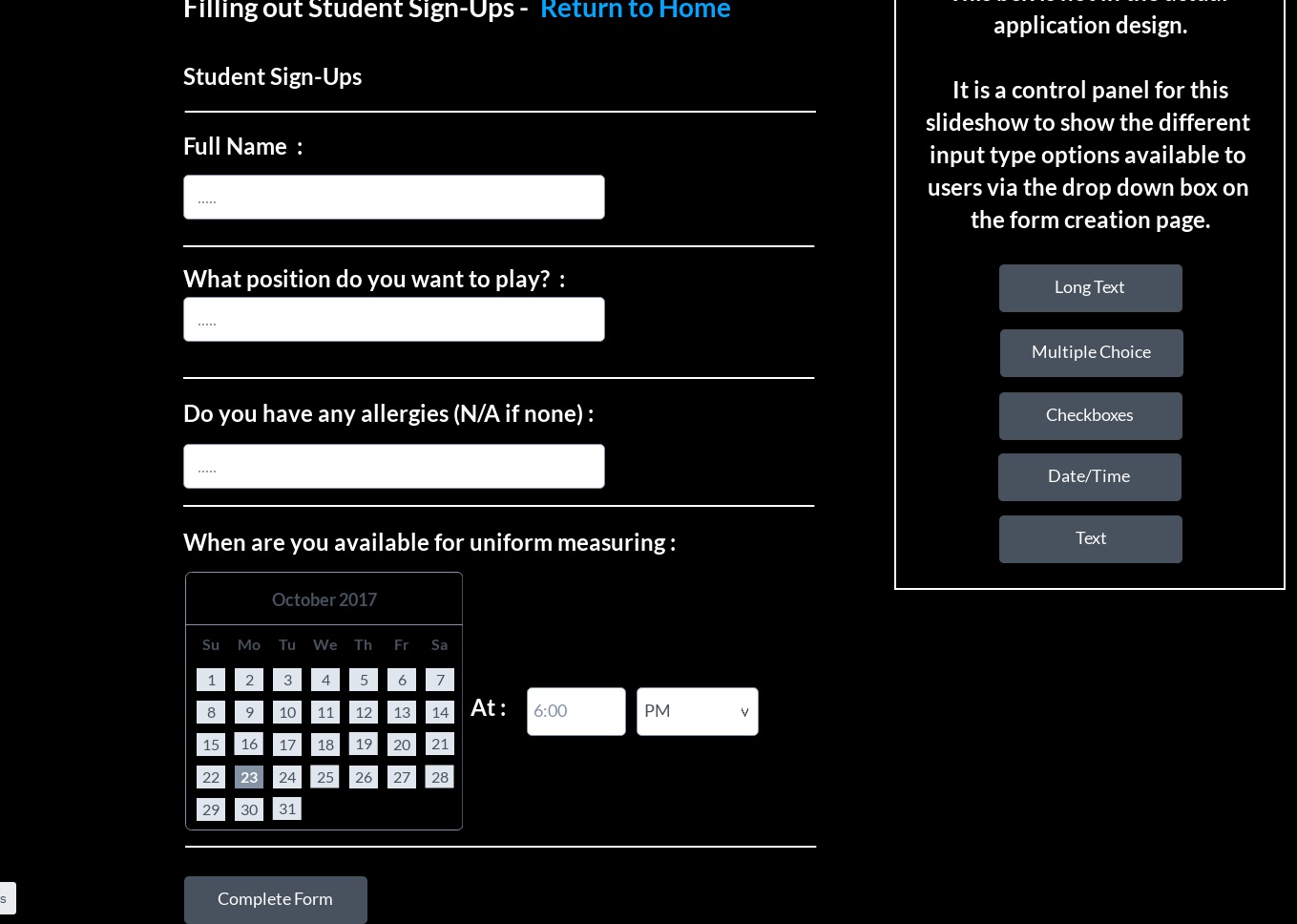
* + Multiple Choice will generate a dropdown box will the user’s specified options as selections

#### Fill-out Checkboxes Example



* + Checkboxes will list all options in a list with a clickable checkbox next to them. This allows the user to make multiple selections.

#### Fill-out Date/Time Example



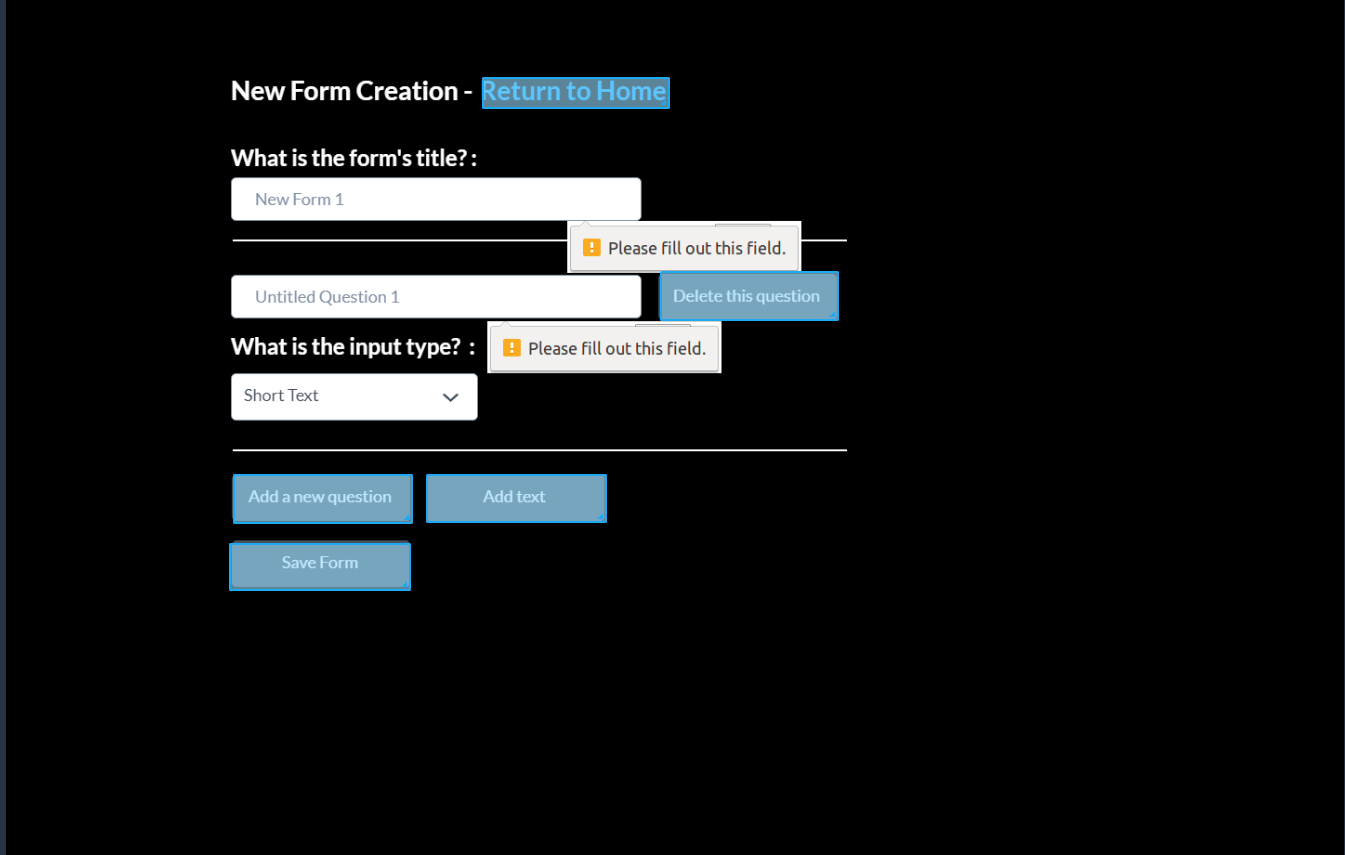
* + Date/Time will generate a calendar input box for the date and a textbox with validation for the time and a dropdown box with AM or PM.

#### New Form Creation



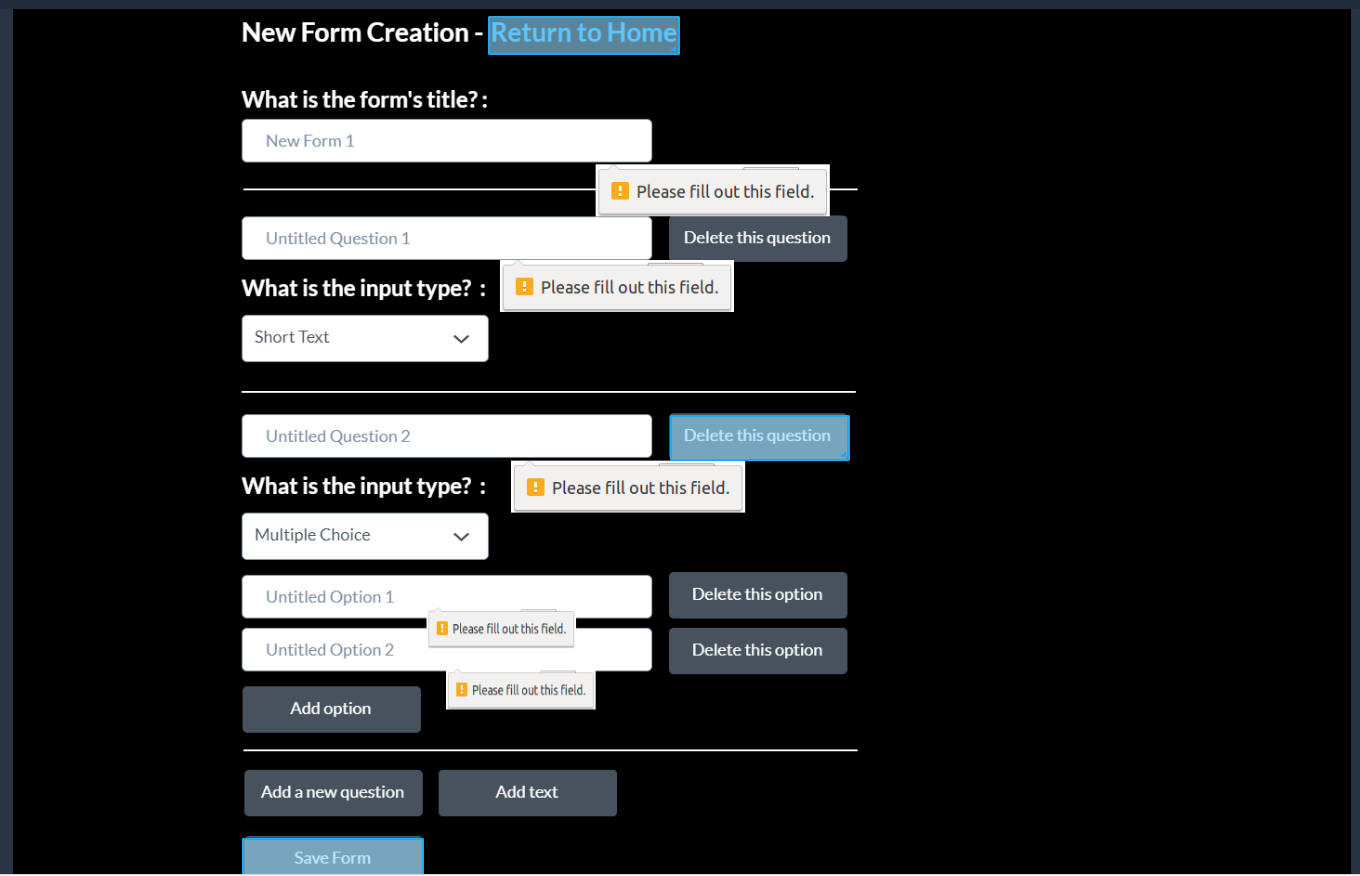
* At the top, the user will have to label their form with a Form Title.

#### Form Creation Short Text Example



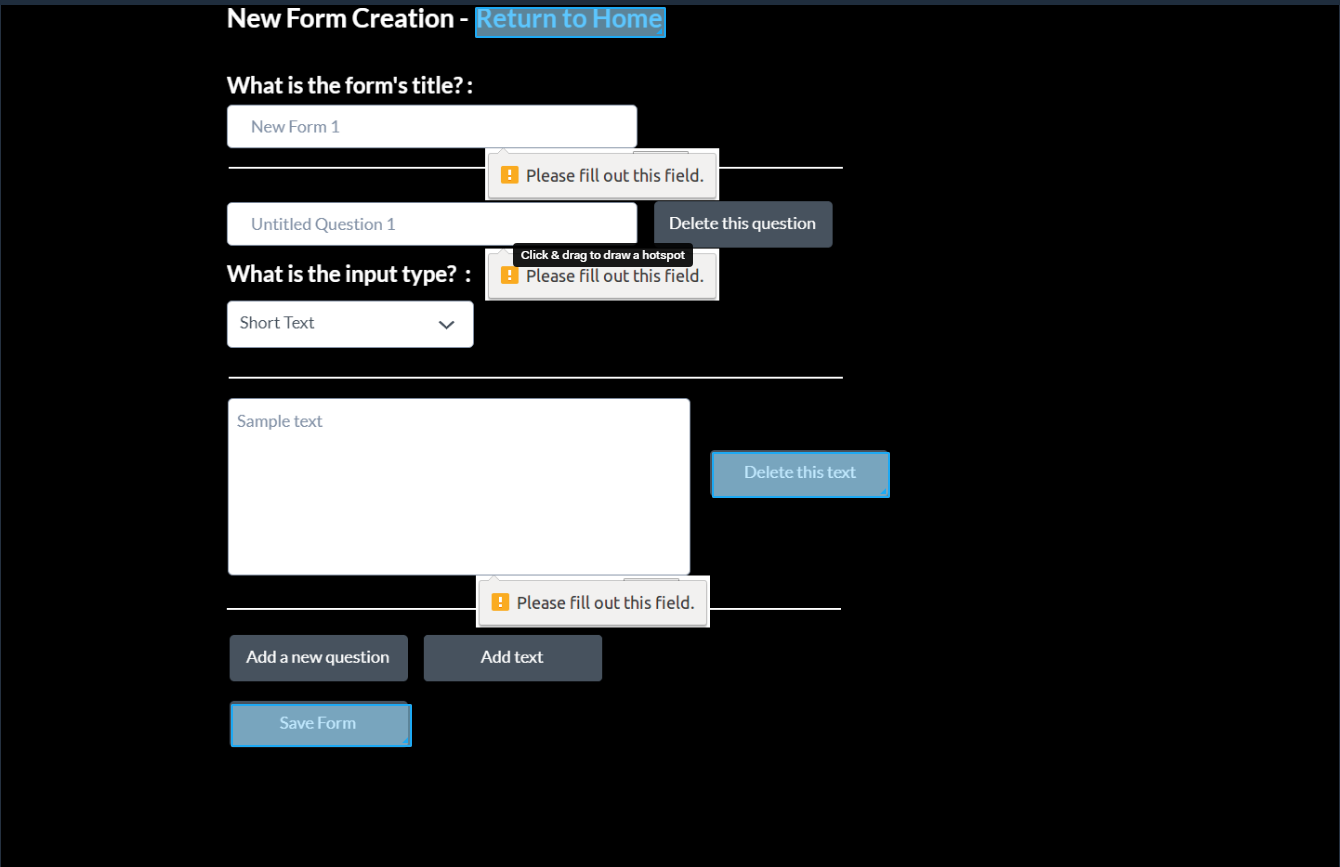
* Users will be able to add a new question, or add static text to their form
  + When clicking on the Add a new question button, a couple of new forms will generate. The first input will be the question (think of it as a label for user input). Next to that input on the same line, there will be a Delete this question, which will de-construct the form input. Underneath will be a dropdown box labeled “what is the input type?” The dropdown box will dictate what form of user input will be accepted.
    - Input Types
      * Multiple Choice -users choose one
      * Check Box-users choose more than one
      * Short Text-single line
      * Long Text-paragraph
      * Date/Time

#### Form Creation Multiple-Choice Example



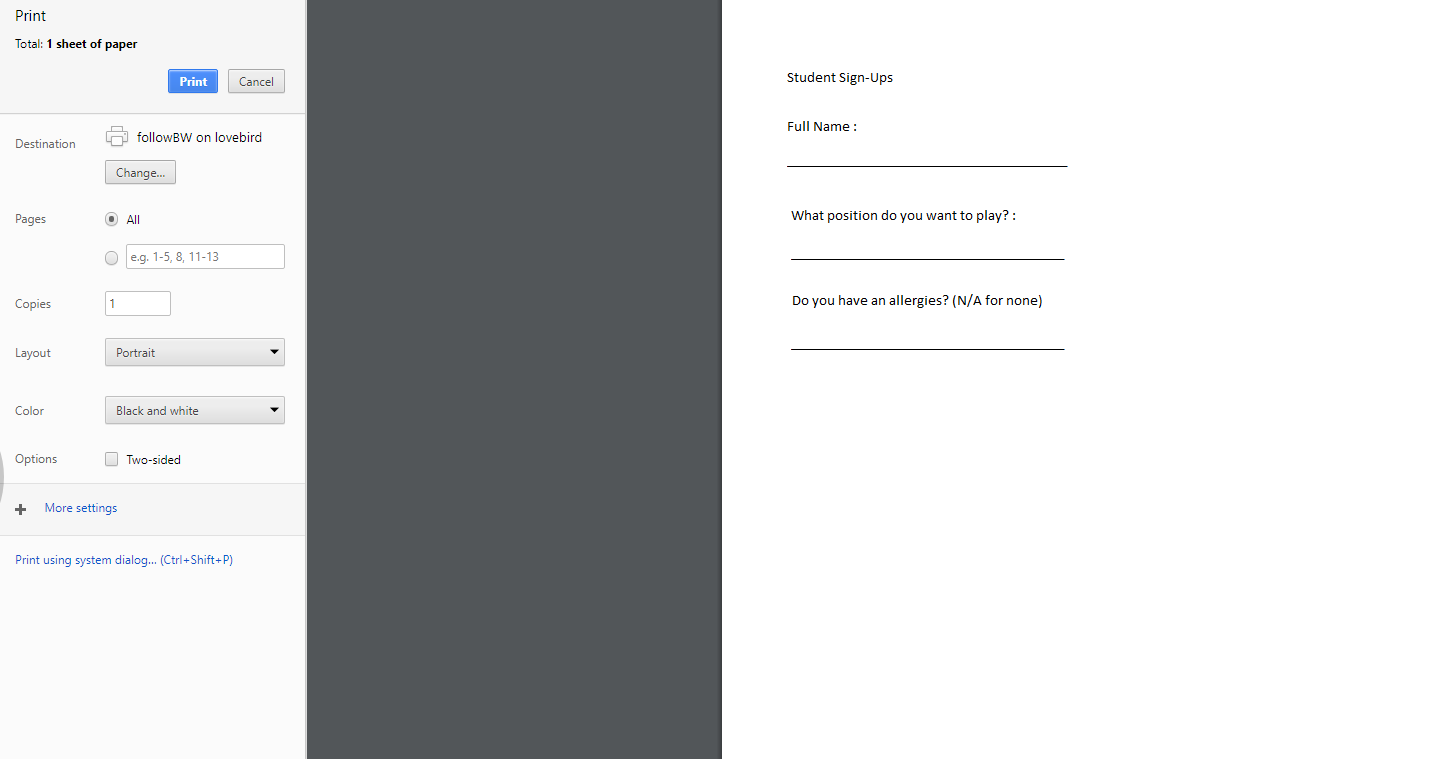
* If the user selects checkbox or multiple choice, more input text boxes will be generated for the user to assign possible answers to their question. Each input will have their own delete button and there will be an Add option button at the bottom that will generate another textbox.

#### Form Creation Add Text Example



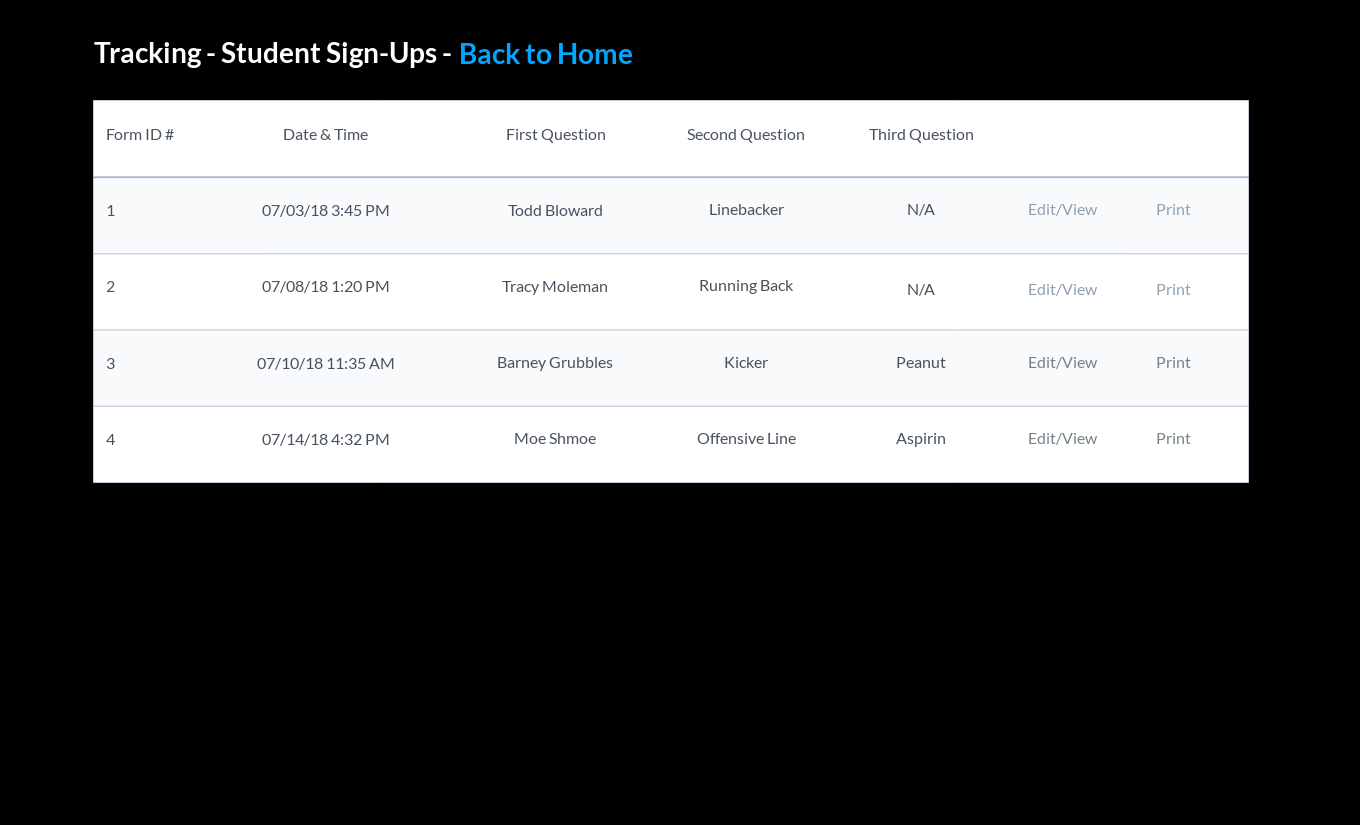
* + When clicking the Add text button, a multi-line input box will generate and a delete this text button. The entry in the input box will displayed on the user’s form as a block of static text (legal, notes, etc.)
* After the user is completed with their form creation, they can push the save form button, which will generate the appropriate SQL tables and entries based on their design, and it will reroute to their homepage. Their homepage table will refresh with the newly created form.

#### Print Blank Form Copy



* This allows the user to print a blank form.

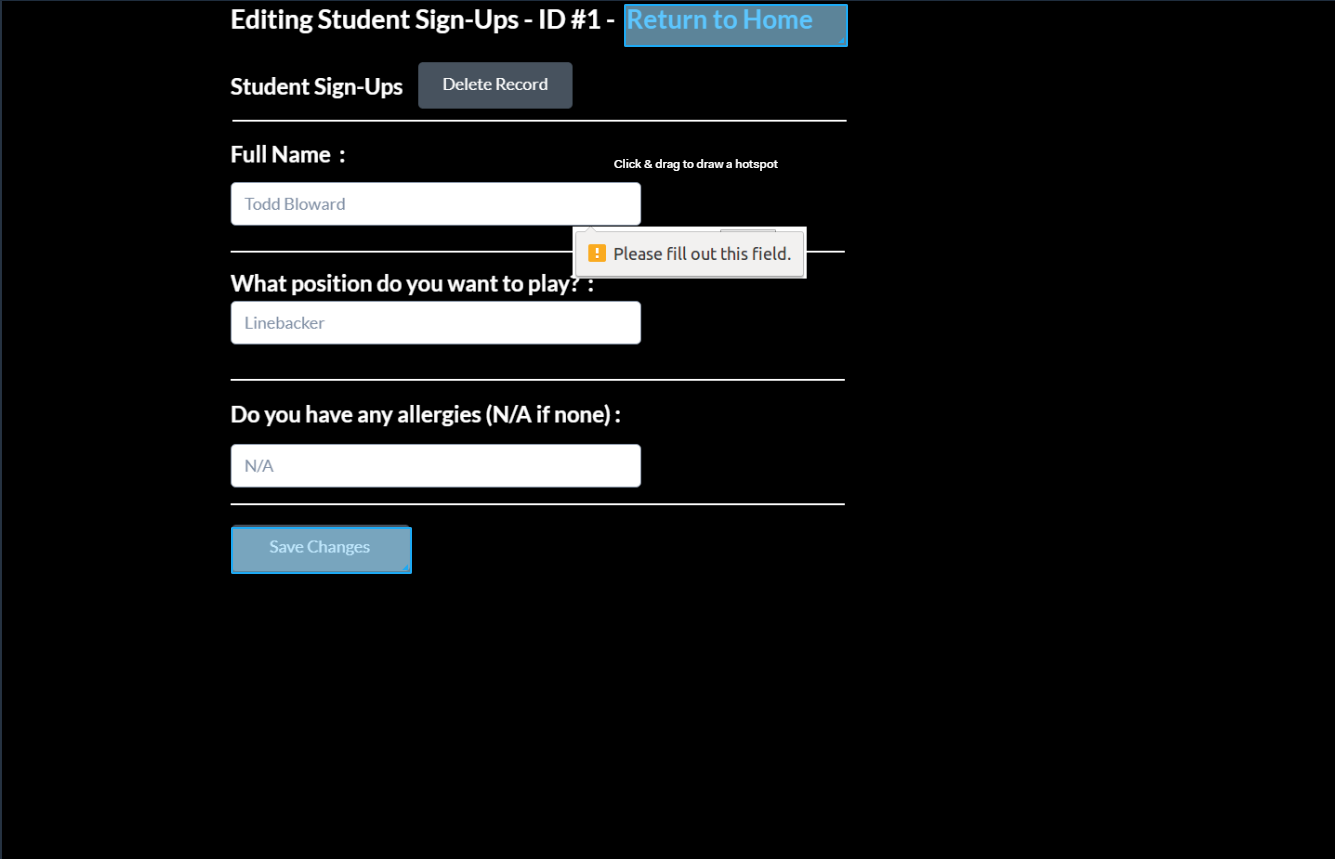
#### Tracking Home-Page



#### Tracking

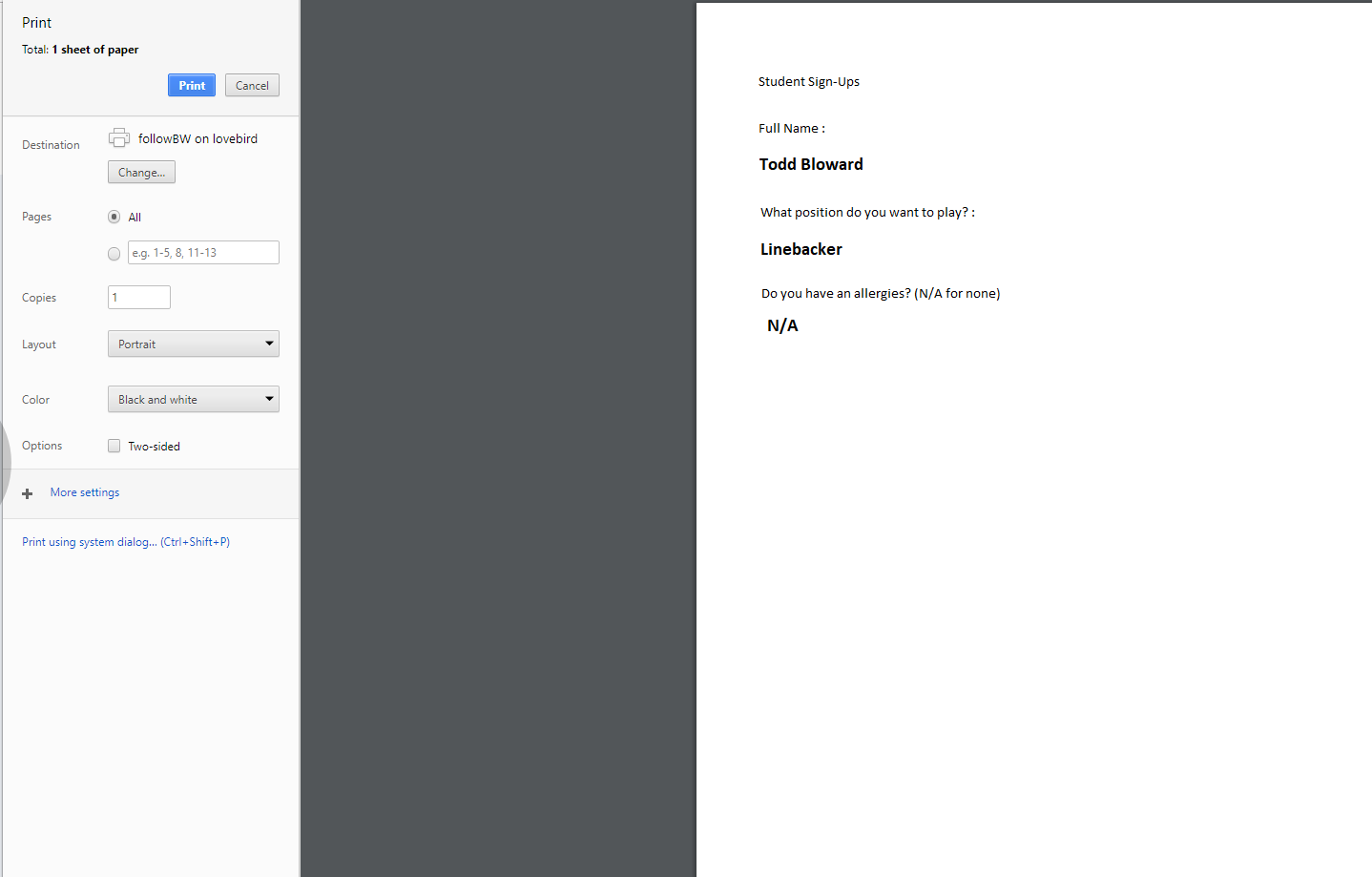
* Each created form will have their own tracking page. At the top, a label of tracking followed by the form title and a link back to the homepage will be displayed. Underneath, the tracking page will display every instance of the form being filled out in a table, listed by their incremented form IDs, date/time of completion, first-second-and-third question answers, an edit/view button, and a print button. Clicking the print button will generate a printable page with all the form fields filled out with the instance’s answer.

#### Edit Form Instance



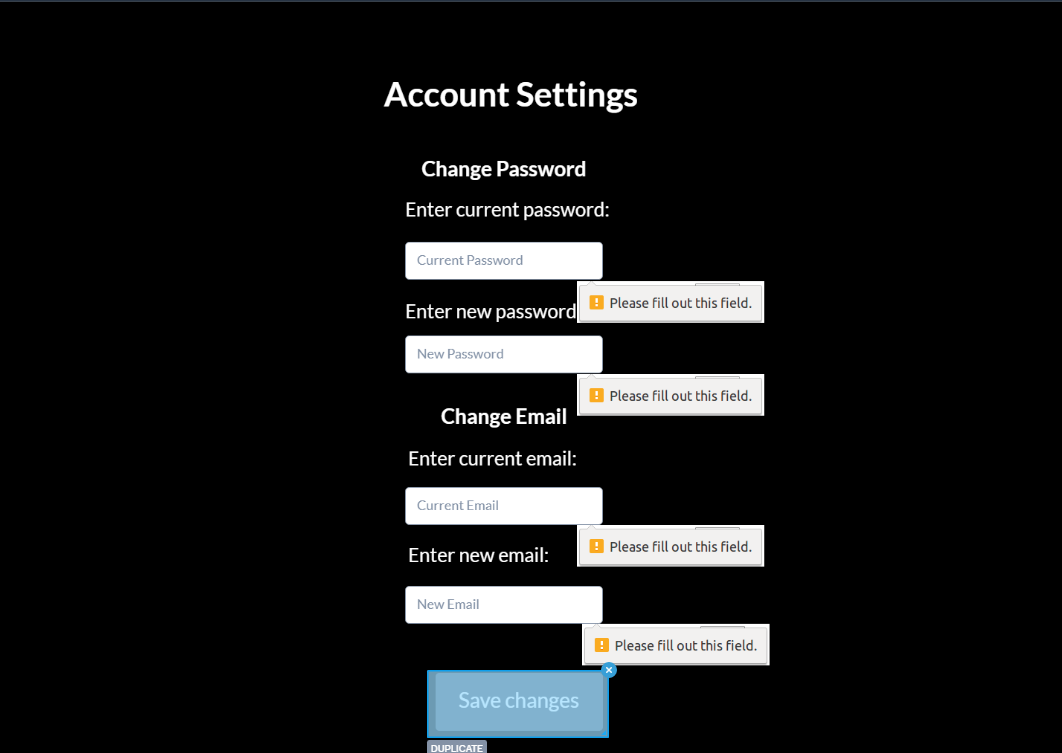
* This page will essentially have the same functionality of the fill-out existing form page except with the answers being automatically filled out with the instance’s answers. At the bottom will be a button to save the answers, which is replace the instance’s answers with the inputted ones in the SQL tables. There will also be a button to delete the record that will lead to a confirmation page and will allow the user to delete the form instance from the database.

#### *Print Form Instance*



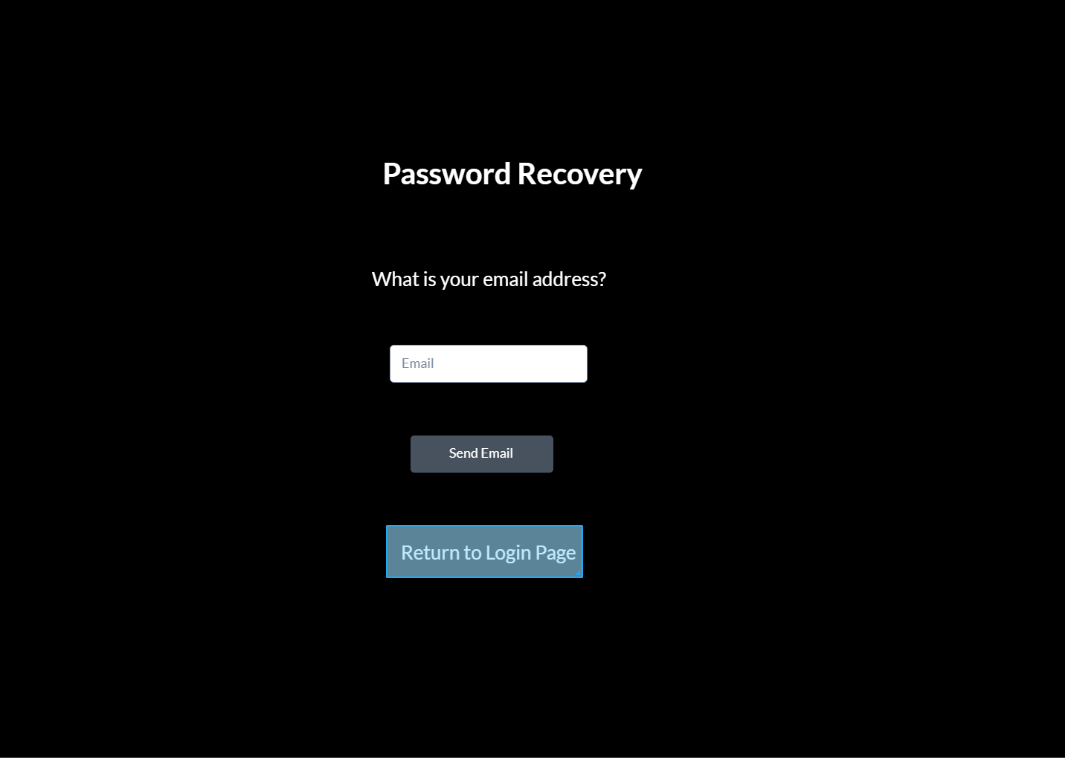
* This page shows that the user can also print a form instance.

#### Account Settings Page



* In this page the user is allows to change their password and email, once the save the changes it takes them back to home page.

#### Password Recovery Page



* When the user clicks on forgot password link it will take them to this page, after they entered their email and click on send email, the user will receive an email to help them recover/change the password.

# Functional Version 0.5

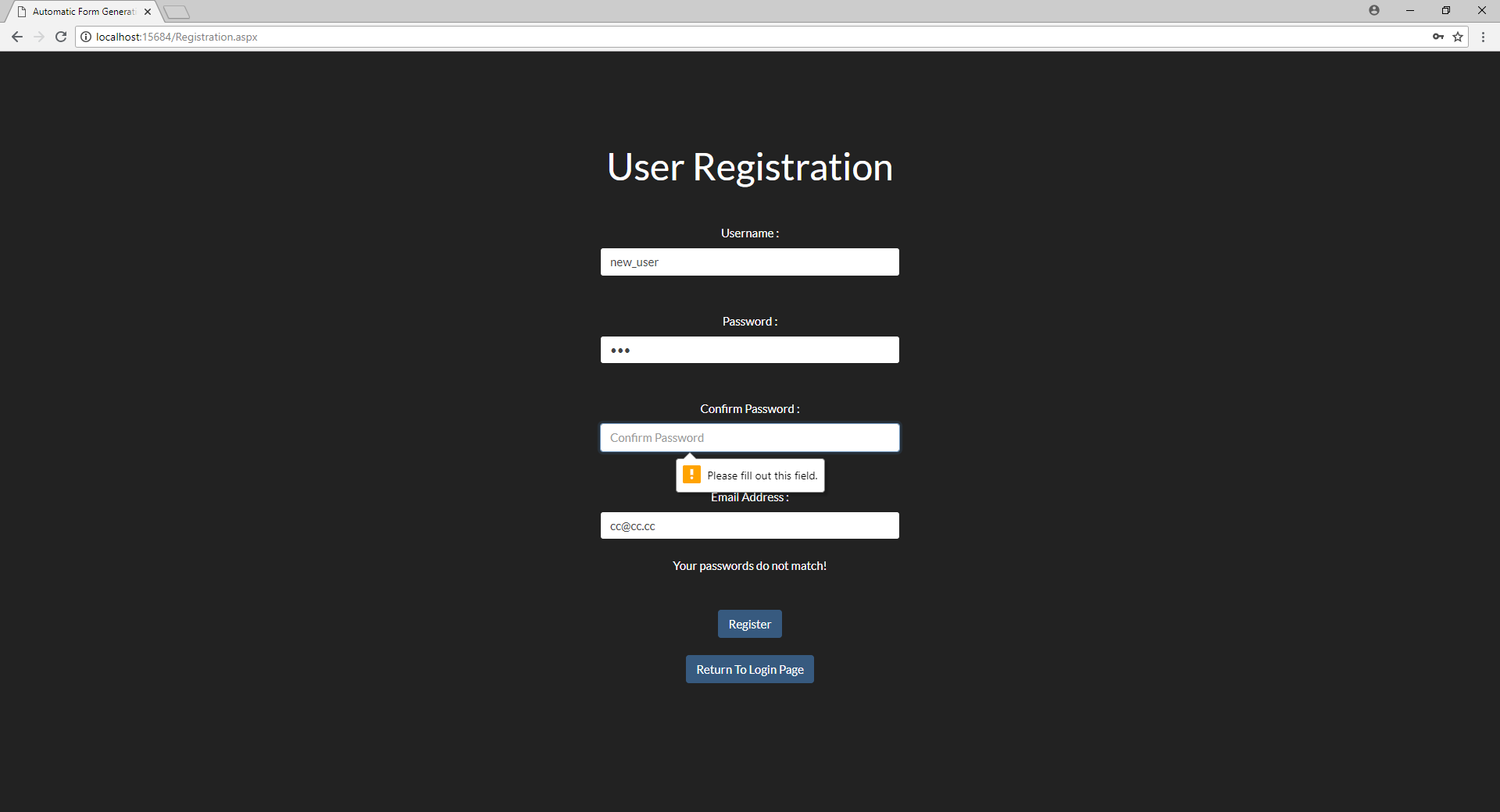
#### The application is powered by a free bootstrap theme - **https://bootswatch.com/darkly/**

#### Default.aspx – Login Page

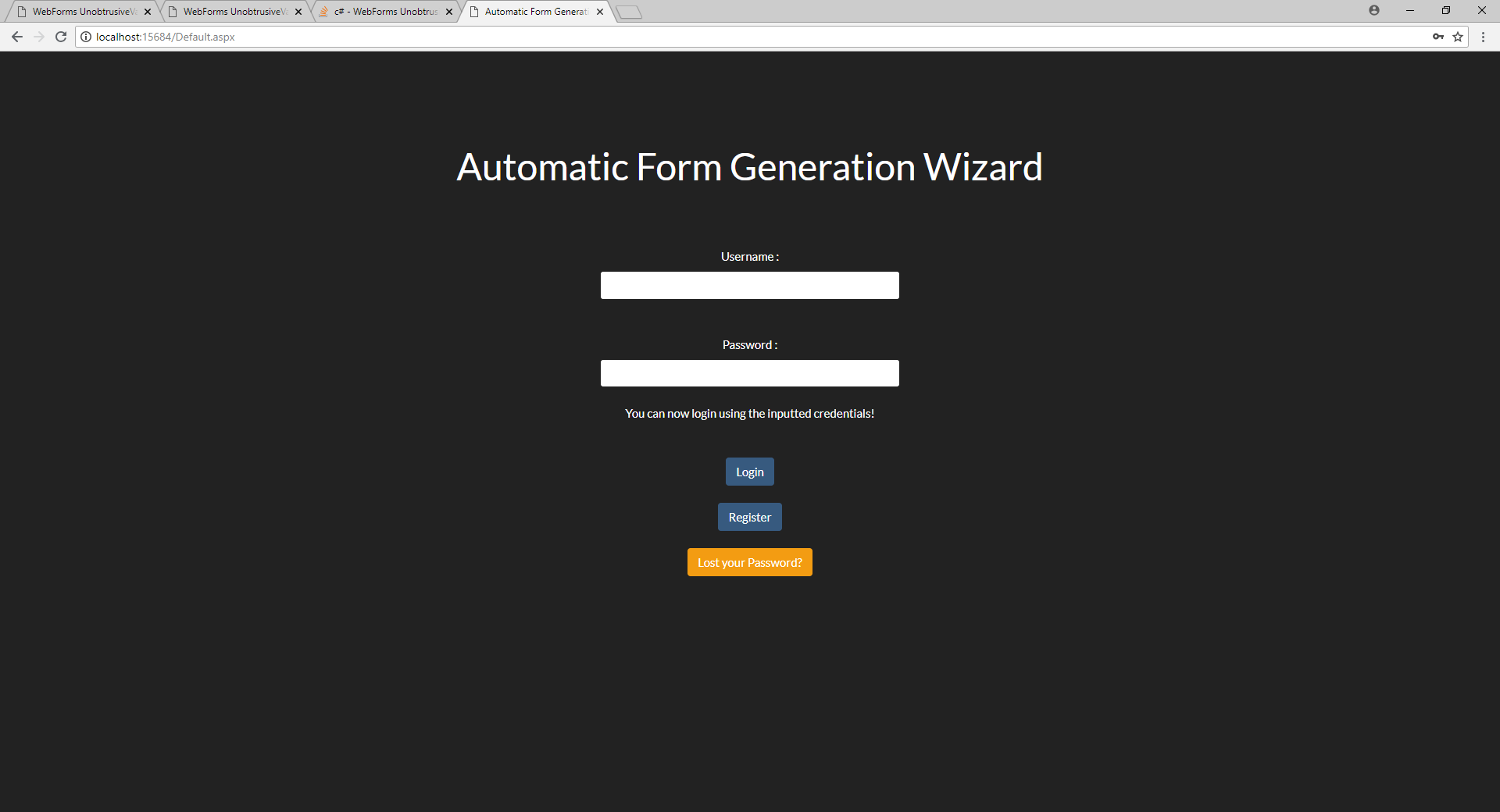
# 

* Users will be able to enter their credentials to enter the application, as well as navigate to a registration and lost password page.
* This application utilizes a built-in basic HTML validation for empty forms. There will be a validation message that appears if the user doesn’t enter in proper credentials.

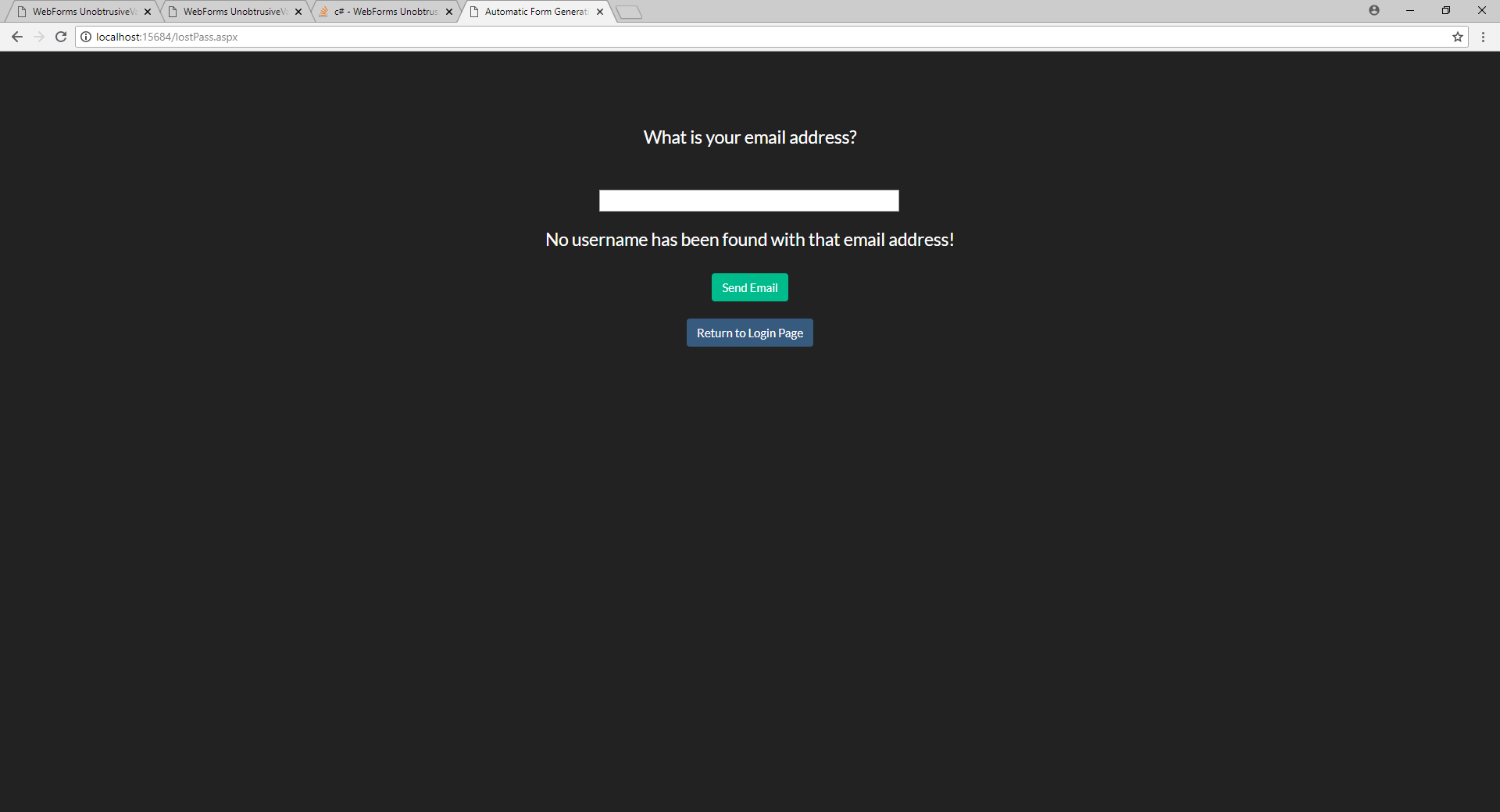
### Registration.Aspx – Registration Page



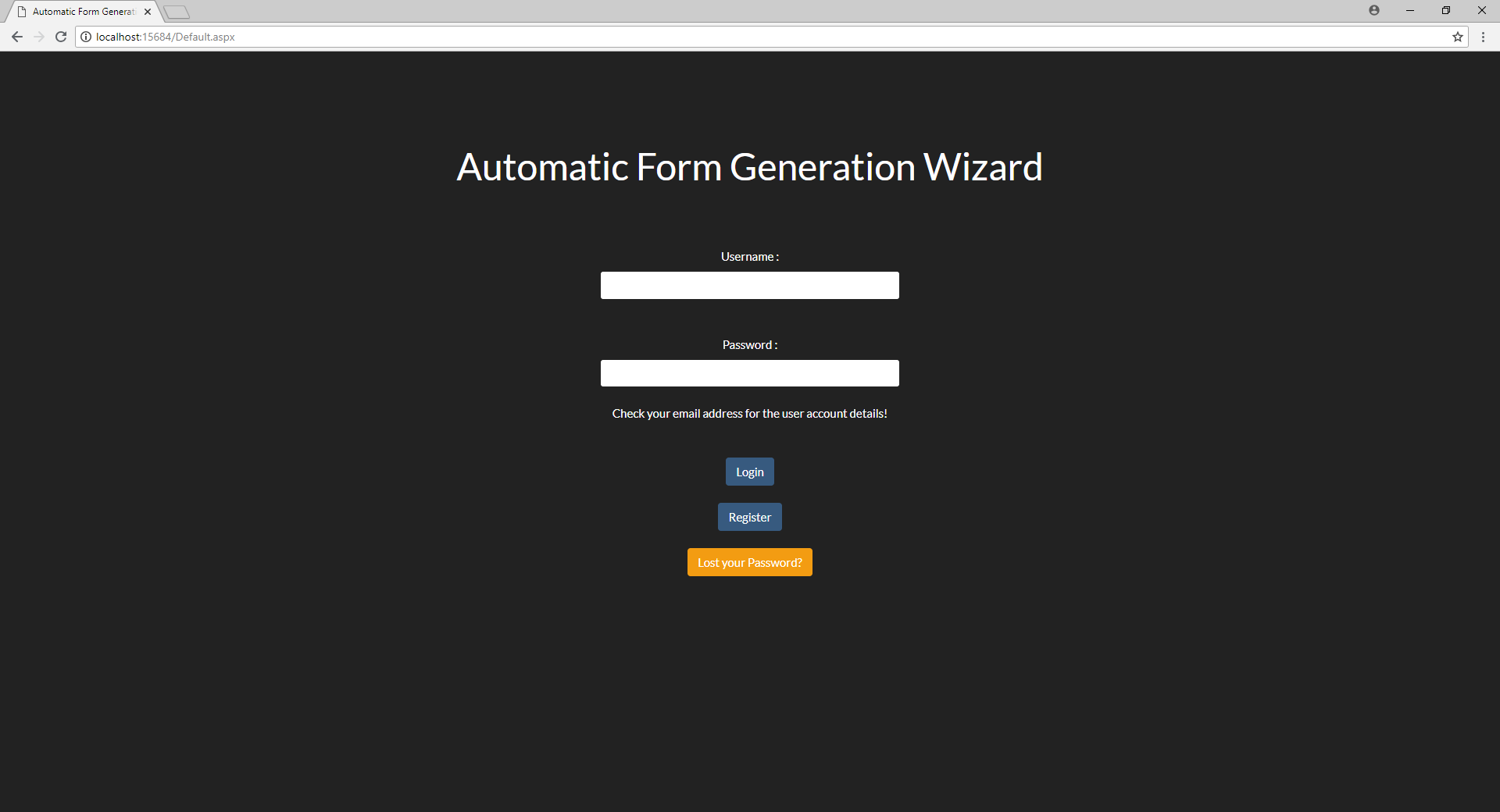
* This page is used to register users into the database. It asks for a username, password, password confirmation and email address.
* It uses HTML validation for empty fields as well as has a warning message that appears between the buttons and the inputs that tells the user if their password and confirmation inputs differ.
* At the moment, there is no validation to check to see if it’s a valid email address.
* The user can complete registration into the database, which shows a message to the user telling them they are registered, or navigate back to login page.



### lostPass.Aspx – Account details retrieval page

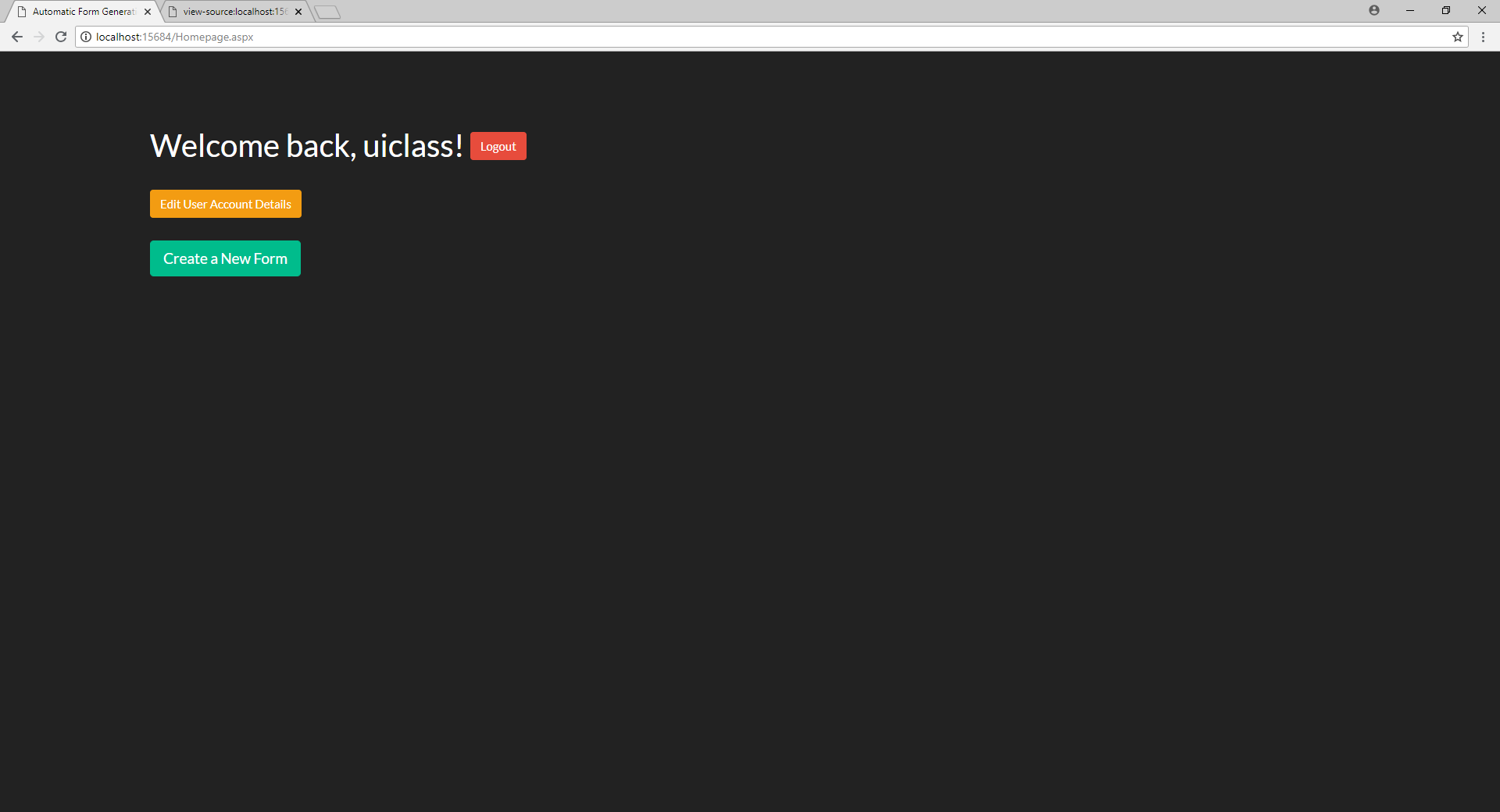


* This page has a simple title and one centered input box for the user to input their email address.
* This input uses a basic HTML validation for empty fields and has a warning message to the user if they input an address that isn’t stored into the database or spits out an SQL error if their address is not a valid address.
* From here the user can submit their address, which re-routes them to the login page and displays a message telling the user to check their email, or just return to the login page without submitting an address



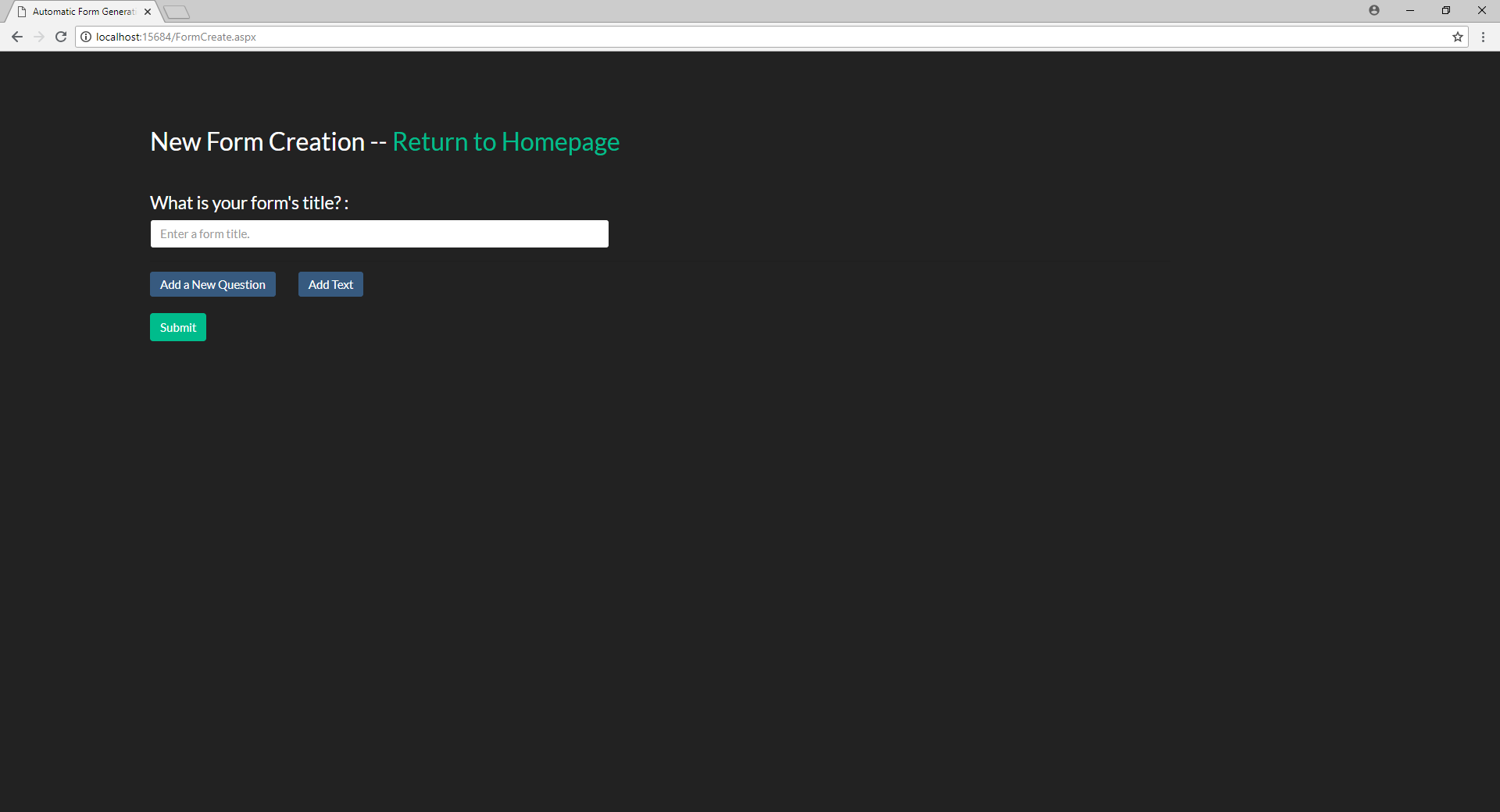
### Homepage.Aspx – User Homepage

### (Unpopulated)

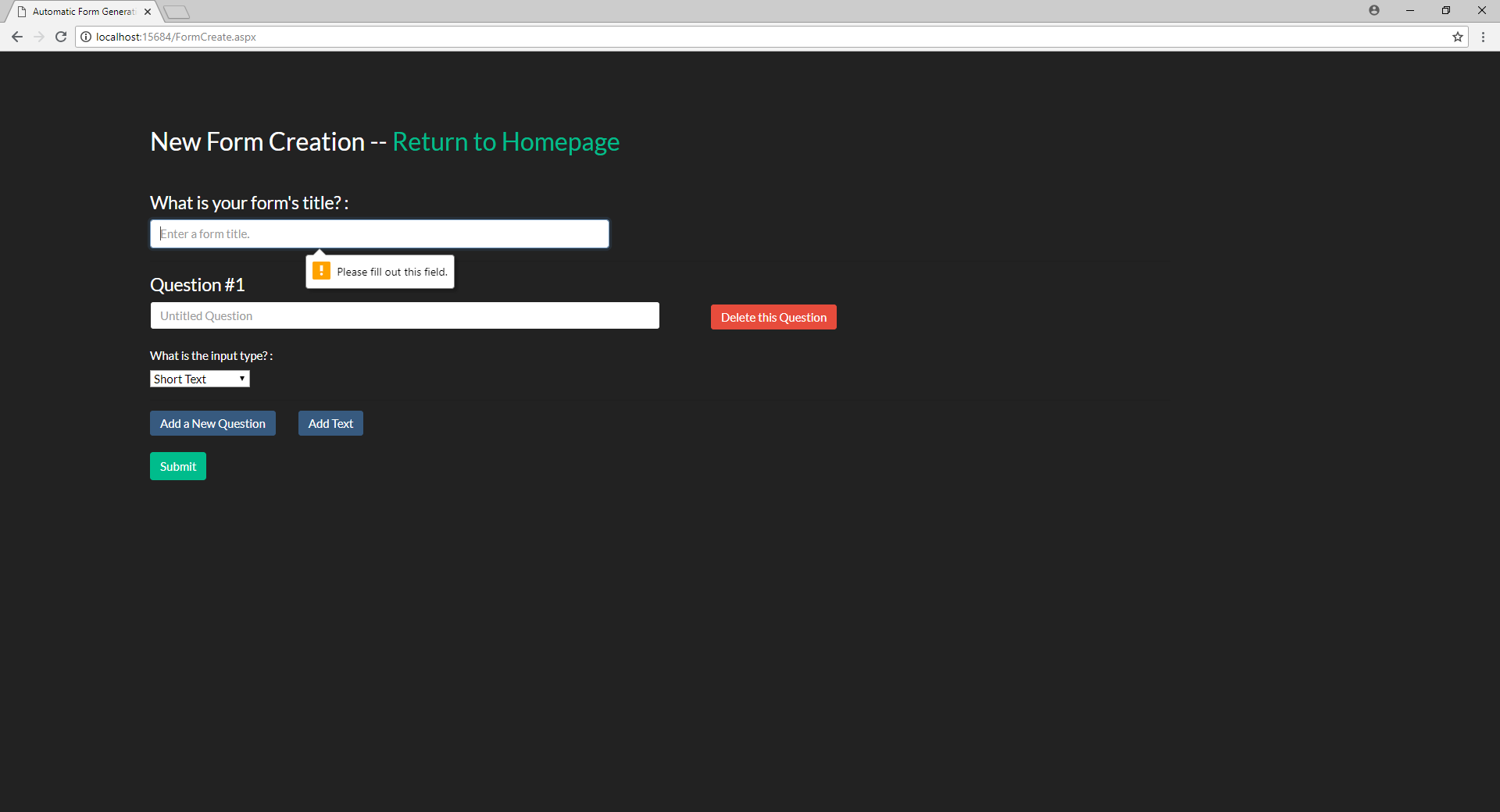


* This is the user’s homepage that displays all of the user’s created forms in a datagridview generated table. If there are no created forms, the table will simply not appear.
* At the top, a welcome message will be displayed and alongside will be a red logout button that destroys the session and returns the user to the login page.
* Underneath the header will have an orange button that redirects the user to an edit account details page
* Underneath the edit user account details button is a slightly bigger green button that leads the user to the form creation page.

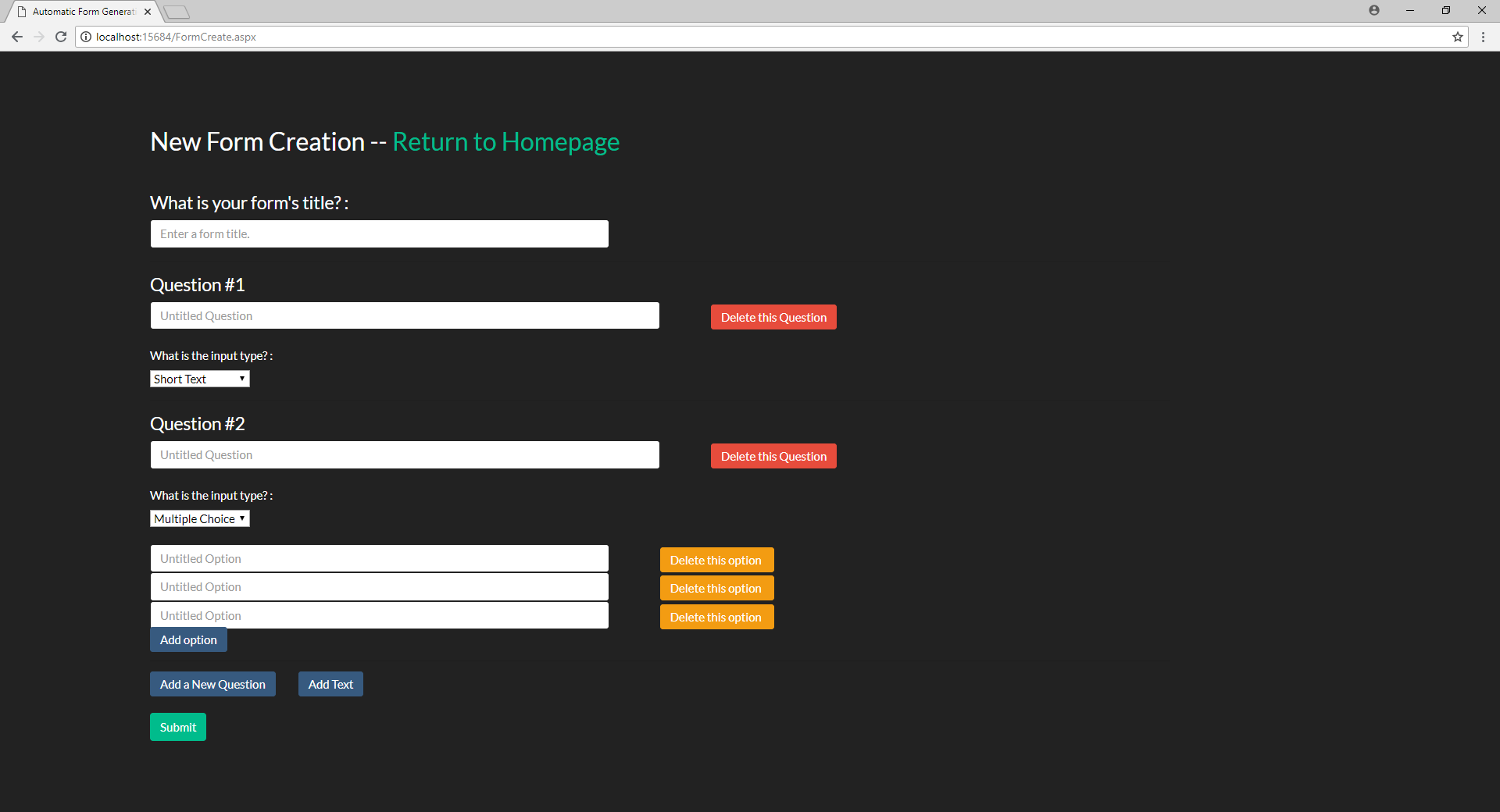
### FormCreate.Aspx – Form Creation Page

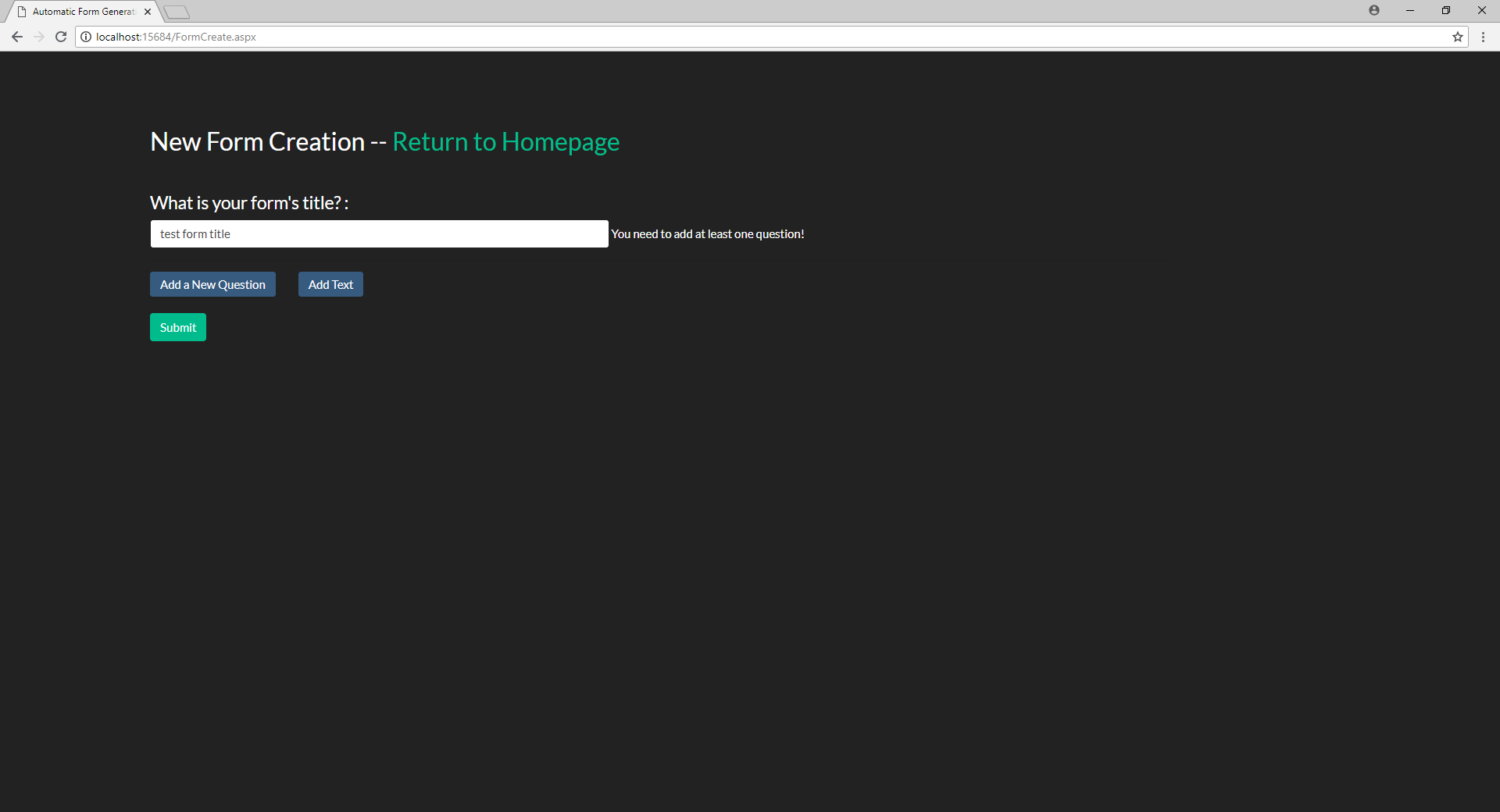


* This is the form creation page that handles the dynamic generation of user inputted questions and their input types. All of the dynamically generated elements of this page are handled in jQuery and it posts back to itself so the server can parse the inputted data and send it to the database.
* At the top is a heading that says “New Form Creation” and has a simple green hyperlink back to the homepage in the same heading size.
* Underneath the heading is a label that asks for the new form’s title and a text input with built in HTML validation
* Underneath the input will be two blue buttons on one line: Add a new question and Add Text
* Underneath the two blue buttons will be a green submit button that submits the structure and redirects the user back to homepage with a notification message



* Every time a user clicks on add question, a new div separated by horizontal rules appears with the question number, an input field for the question’s details, a red delete this question button, a label asking for an input type, and a dropdown with all of the different input types.
* If the user changes the dropdown box selection to “Multiple Choice” or “Checkboxes”, two new input fields, two corresponding orange “delete this option” buttons, and a blue add Option button will be generated. All the textboxes are empty validated with basic HTML, the delete option buttons destroy the input they are in line with, and the add option button generates a new input field and delete option button.

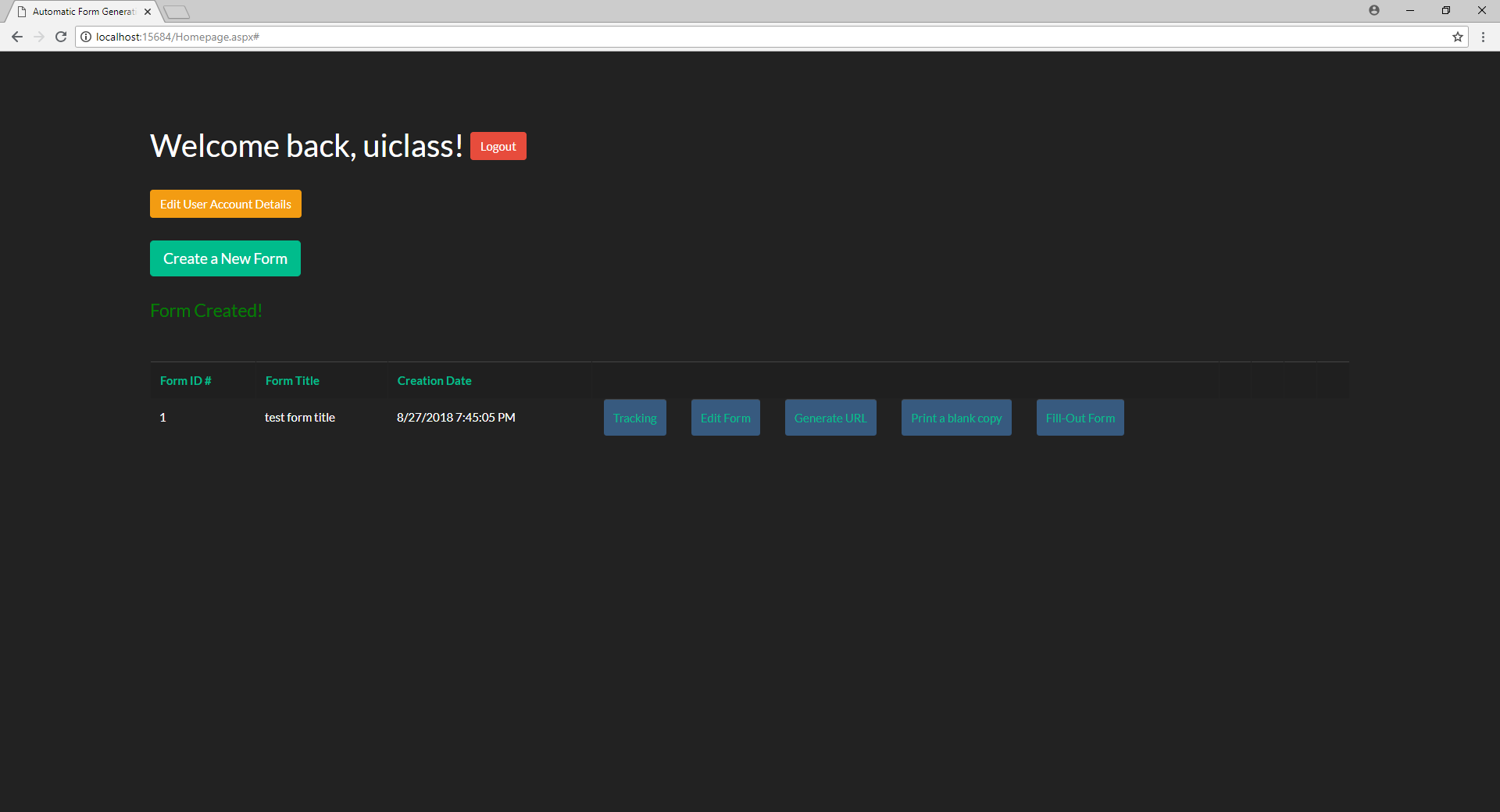




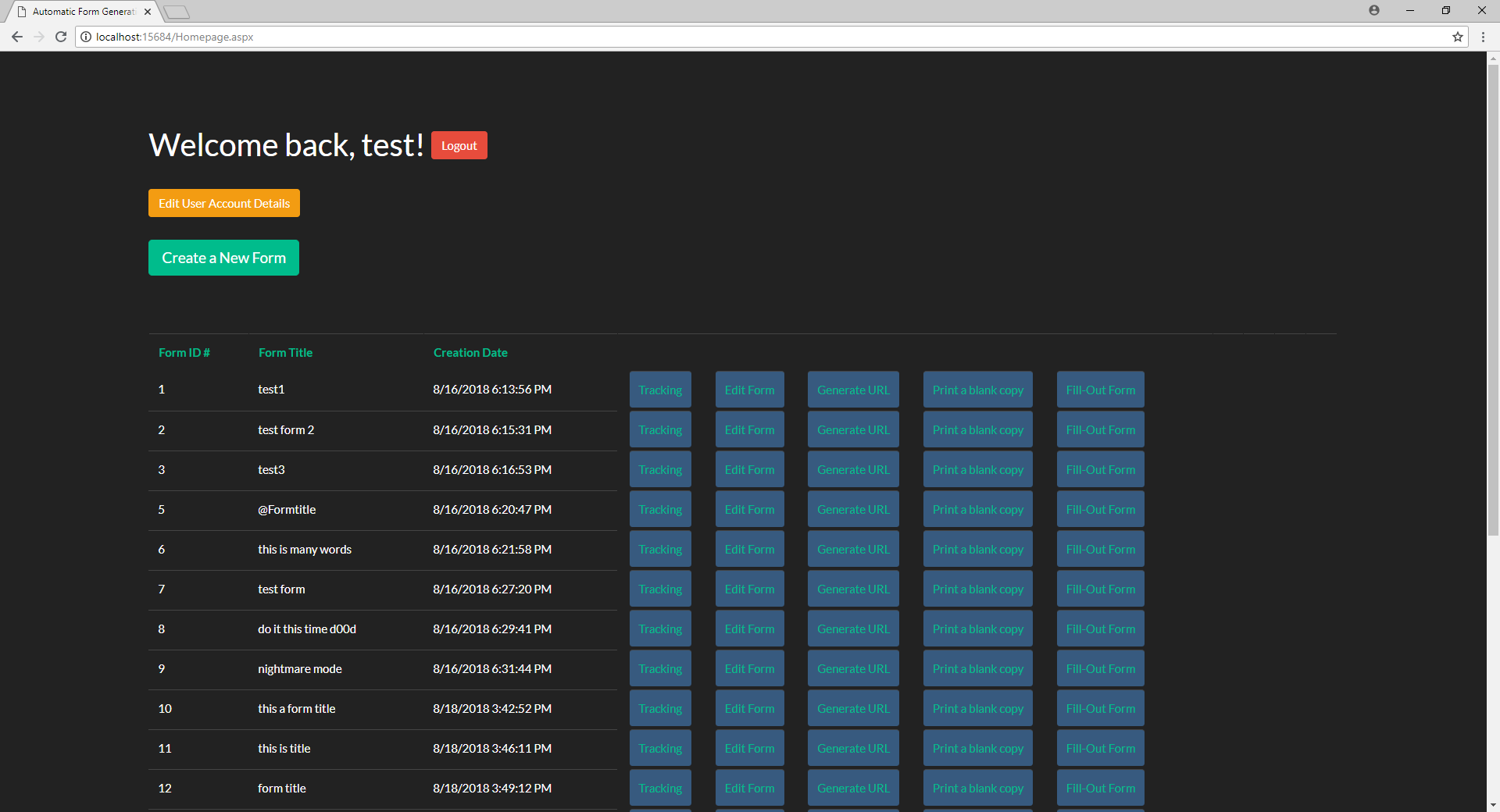
* If the user tries to submit a form with a form title but no questions, a little message pops up on the side of the form title input instructing the user to add a question to the form.
* Once the submit button is clicked and the form is successfully created, the user is redirected to the homepage with a “form created” message in green underneath the “create a new form” button.

### Homepage.Aspx – User Homepage

### (Populated)



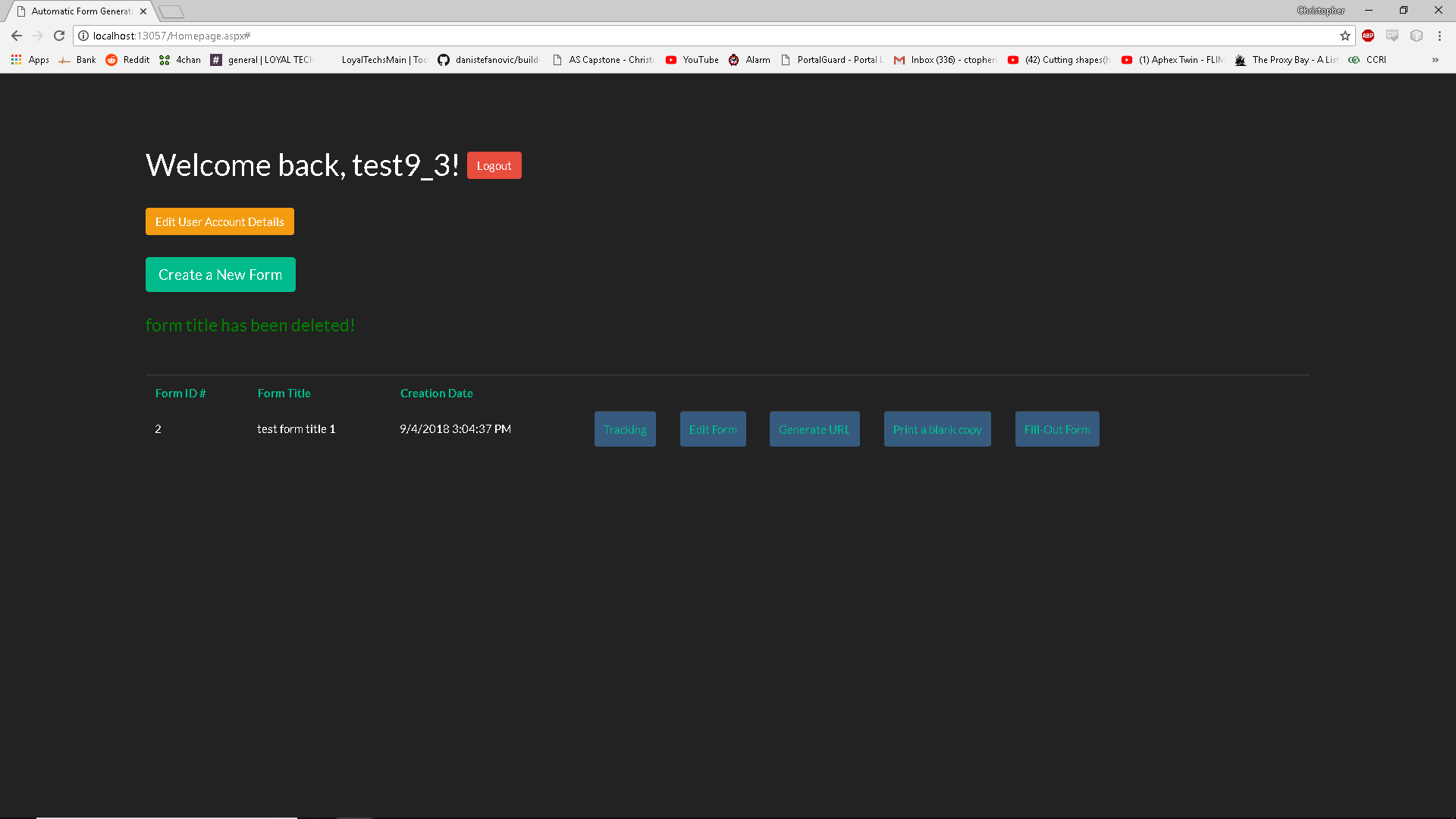
* Once the user has created a form, the table will now generate. The first three columns of the table (Form ID, Form Title, and Creation Date) are links that automatically sort the table ascending or descending based on which column was selected. To the right of those three columns will be five blue button links that navigate to their corresponding pages (Tracking, Edit Form, Generate URL, Print a blank copy, and Fill out Form). Each of these buttons will be generated for every form that the user has created



### FormCreate.Aspx – Edit Form Page

### 

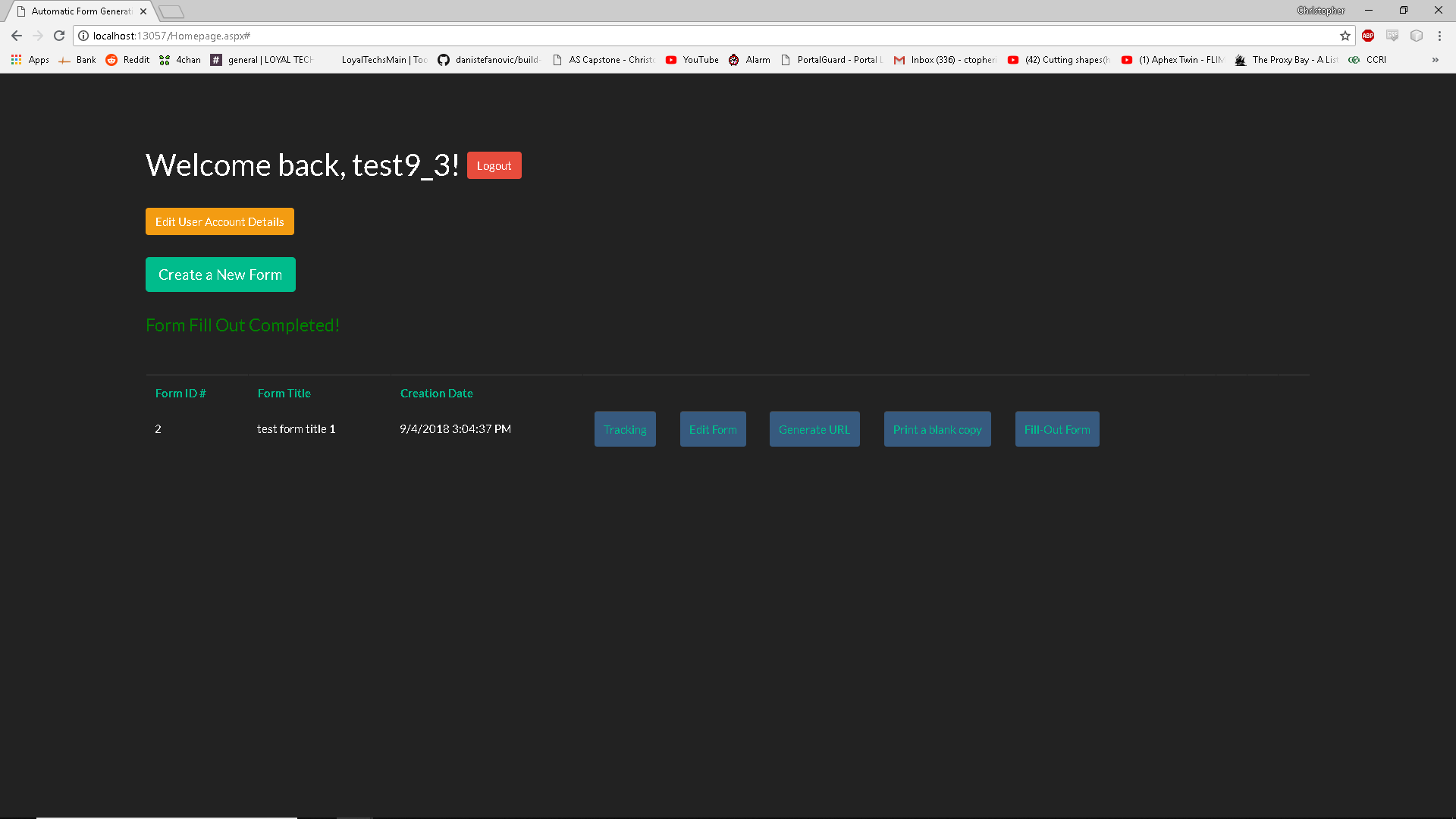
* If the user clicks the Edit Form button on the table, they will be navigated to an already populated form creation page with a new heading that reads “Editing Form \*Form Title\*”. Underneath the altered heading will be a “Delete this Form” button.
* Clicking the “Delete this Form” button will automatically route the user back to their homepage with a success message.
* Outside of these new elements, this page behaves exactly like the form creation page except that instead of creating a new form, it alters the existing form’s structure.



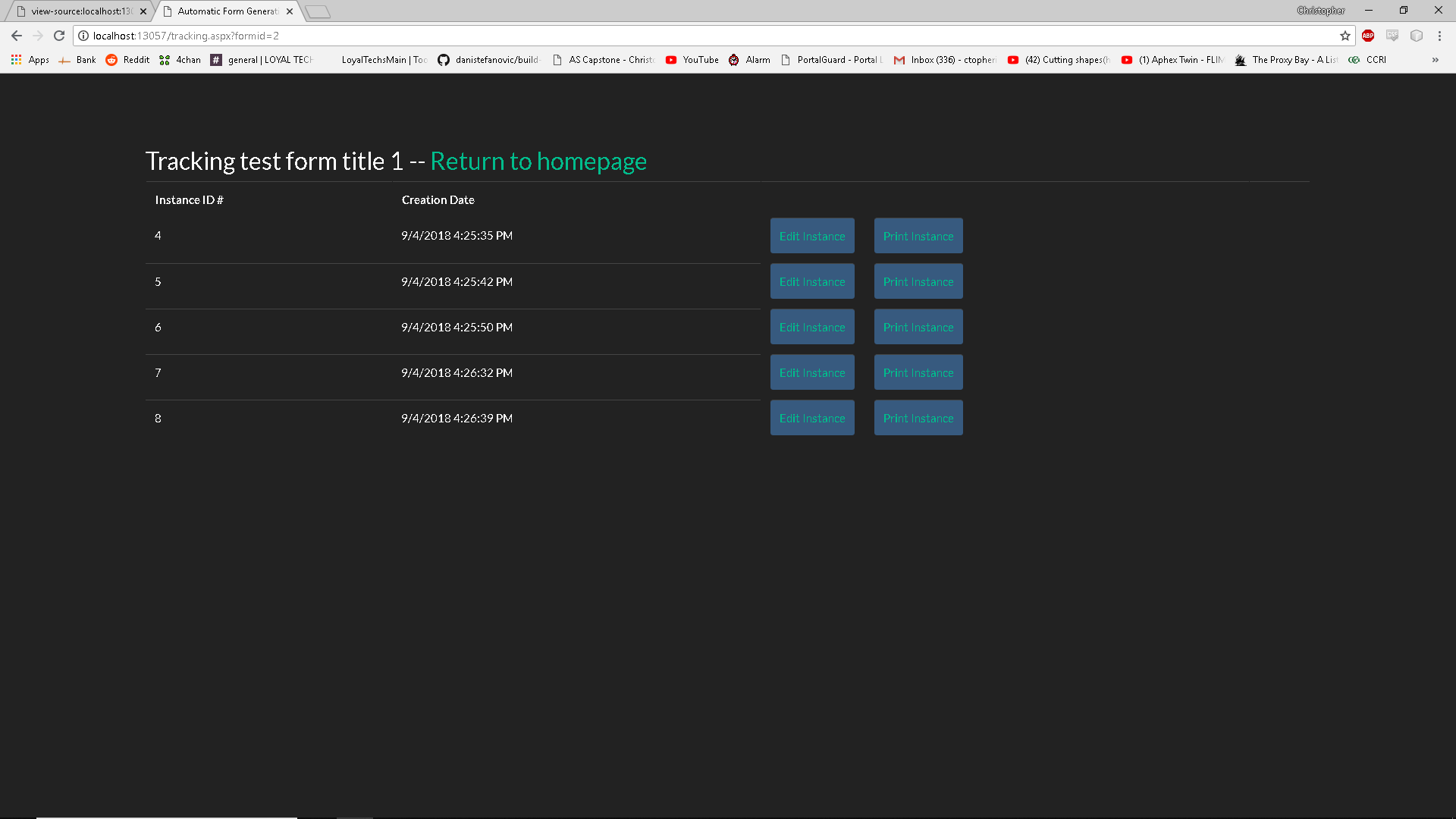
### Fillout.aspx – Fill out Blank Form

### 

* If the user clicks the “Fill Out Form” button on the table in the homepage, they will be navigated to this fill out form page.
* At the top will be a heading with “Fill Out - \*form title\*” and a hyperlink back to the homepage.
* Every question that was created in the form creation page of this corresponding form will have the appropriate user chosen input type (short, long, multiple choice, check box, date & time)
* At the bottom of the page is a “Complete Form” button that will save the answers into a new instance and navigate the user back to the homepage with a success message.

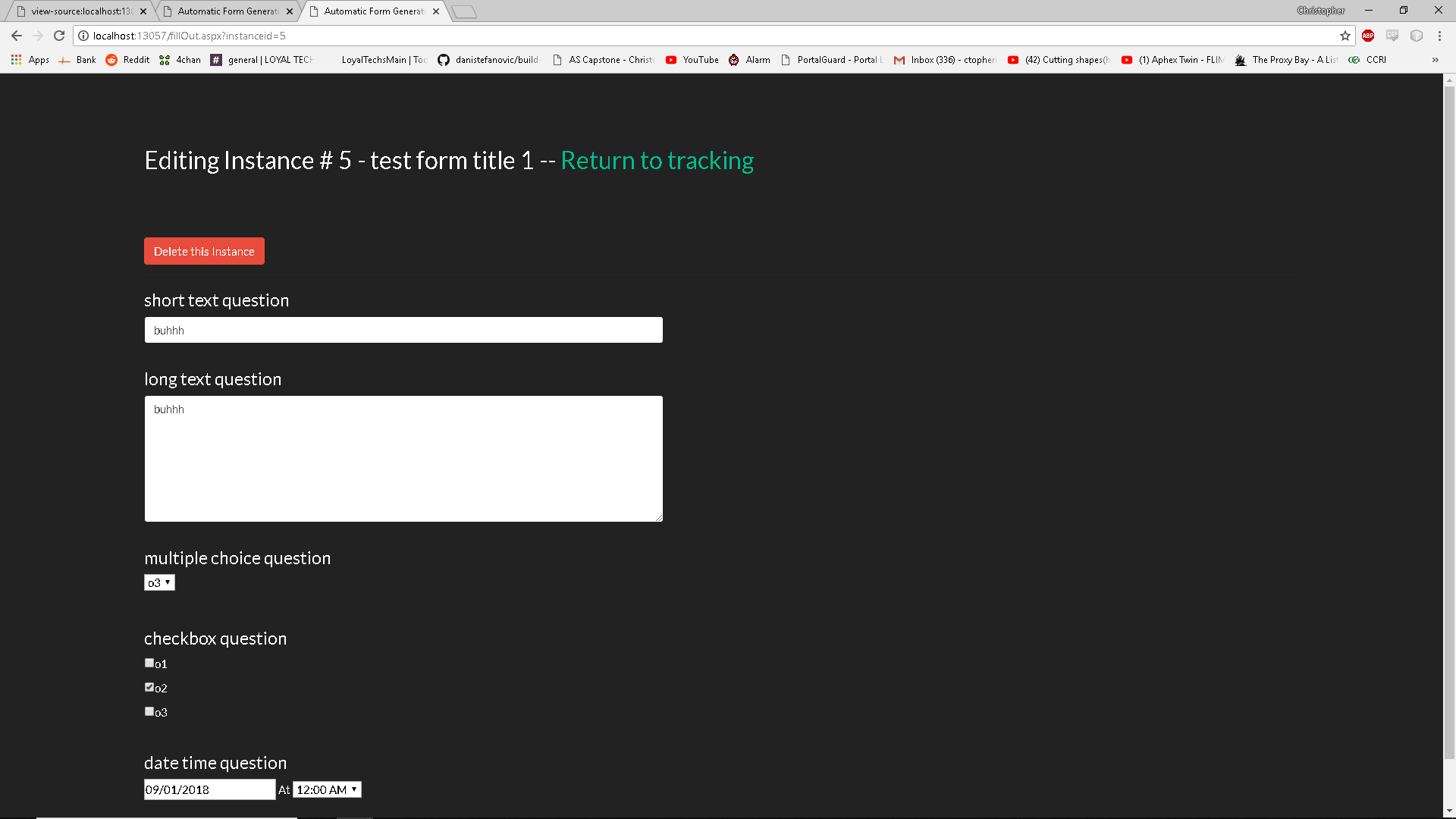


### Tracking.aspx – Instance Tracking Page



* If the user clicks the “Tracking” button on the homepage, they will be navigated to this page, where a table will generate with all the instances of filled out forms.
* At the top will be a header that states “Tracking \*form title\*” and a link to the homepage
* Underneath the table will be generated with instance IDs and instance creation date as the main columns. Each entry will have two buttons generate, “Edit Instance” and “Print Instance”.

### Fillout.aspx – Editing Instance



* If the user clicks the “Edit Instance” button on the tracking page, they will be navigated to an already populated instance of the Fill Out page.
* The header at the top will be altered to state “Editing Instance #FormID - \*form title\*” and a hyperlink to the homepage.
* Underneath this header will be a red “Delete this Instance” button that will delete the instance table, remove it’s entry from the instance master table, and redirect the user to the tracking page for that particular form.
* Everything else will behave the same as the fill out page, except the “Complete Form” button at the bottom will update the form instance instead of creating a new one. Deleting/Updating the form instance will cause a notification message to appear on the tracking page when the user is redirected to it.

