

# Lab Activity: Flask Api

## 1 Introduction

In this lab activity, you will create a Flask Server API that mimics a Ticket Master-alike website. The main features of this site should be:

- Displays theater plays been performed at Benedum concert hall.
- Displays detailed information of any given play.
- Help the customer to select a seat in the theater
- Print (in the screen) the information about the purchase

Your application shall mimic (in a simple way) the site shown in the following URL (you can grab the images, descriptions, etc. from that site)

<https://trustarts.org/calendar?genre=Broadway%20&%20Musical%20Theater>

## 2 A general Note

Trying to be “as nice as possible” I am providing a skeleton that has all the pages that your project will need, including the HTML, CSS, and images. However, besides you need to create the Flask/python program to serve those pages, you will need to move these files to the correct folder, as Flask requires, and also modify few lines in the HTML files to add the required project functionality. Please open all the HTML files and look for the keyword “PBF”. This keyword is used to mark what lines you need to add your own code.

DO NOT HARD-CODE ANY PATH, SUCH AS

```
<a href="/buy_tickets.html">
<link rel="static/css/main_page.css">

```

These paths might work ok in your computer, but if you deploy this project to a server provider, such as Amazon Web Services, they probably will not work at all. Use Flask function `url_for()`, as discussed in class.

## 3 A quick video about the App

Please watch the video linked below for a visual explanation on what is required for this project

<https://youtu.be/pfpWasQvBfg>

## 4 Steps to complete your Flask application

### 4.1 Move the HTML files

- a) Flask applications use a default location to store all the project HTML files. This has been discussed in class. So go ahead, create a default directory what will hold your HTML files
- b) Move the HTML files provided in the lab skeleton to that directory.
- c) Open each HTML file and update the `img src` attributes, links, etc. based on where the links are in your project. Please refer to Section 2 above for guidance on what and how needs to be done.

### 4.2 Create and activate a virtual environment

Based on what we discussed in class, create a virtual environment for this project. After creating, activate it.

### 4.3 Download Flask Library

From the VS Code terminal window, install the Flask library to this project.

### 4.4 Save this project environment setup

As discussed in class, use freeze command to save the environment variables into the `requirements.txt` file. This file can be used by other developers to install your current libraries, instead of using the ones that have been installed in their computers.

### 4.5 Create your python program

#### 4.5.1 Create the python file

- a) Create a new file that will hold your python program. Name it as **ticket\_master.py**.
- b) Place an **import** command in this file that gets the necessary Flask components that your application will need (if you do not know which ones at this time, that's ok, as you use call them in your code you will know what needs to be imported).

#### 4.5.2 Add the "/" (root) route

- a) Add the root route in the python file: this route shall display the main (HTML) page.
- b) Run your python code and see if you get the main page. If the links you added in Section 4.1 are correct, you should see the main page with all its images.

#### 4.5.3 Add the Christmas\_story route

- a) Add the `christmas_story` route in the python file: this route shall serve the `Christmas_store` page.
- b) Update the `main_page.html` page to reflect this new route path. Again DO NOT HARD CODE the link there, use the `url_for()` instead.
- c) Update the `img` link found in the `Christmas_story` page. Do not change the "form action" link yet, since we do not have the route for that link at this point.
- d) Run your application and on the main page, click the Christmas store ad and see if your server sends you the `Christmas_story` page back.

#### 4.5.4 Add the buy\_tickets route

- a) Add the buy\_tickets route. Two things that are important in this particular route:
  - i) This route accepts an additional URL path, i.e., /buy\_tickets/<play\_name>.
  - ii) Remember, this route shall accept HTTP **Post** requests. You need to specify it!
  - iii) **play\_name** is used as an input for the route associated function.
  - iv) When rendering the buy\_tickets.html file, you will need to pass along the play\_name, since the HTML file references this variable.
- b) In the buy\_tickets.html file:
  - i) Update the head >> CSS link, using url\_for() functionality.
  - ii) Update the img src link, using url\_for() functionality.
  - iii) Update form action link to the order\_summary page. Use url\_for() functionality. The order summary page needs the play name, so put this info inside the url\_for too.
  - iv) Update the first H1 found in the body section of the page to have the play name displayed during run time.

#### 4.5.5 Add the order summary route

- a) Add the order\_summary route. Two things that are important in this particular route:
  - i) This route accepts an additional URL path, i.e., /order\_summary/<play\_name>.
  - ii) Remember, this route shall accept HTTP **Post** requests. You need to specify it!
  - iii) **play\_name** is used as an input for the route associated function.
  - iv) When rendering the order\_summary.html file, you will need to pass along the play\_name, date, seat zone and seat sector, customer name, and ticket price (just hard code to \$100.00), since the HTML file references these variables.
    - I would pass just one structure containing all this information. Create a JSON object before rendering the order\_summary page, and pass it along with the render\_template call.
  - v) Update the order\_summary.html file to accept the JSON object.
  - vi) Run your project, select Christmas story play, click on buy tickets, choose seat and complete the purchase with your information, and then check if the order summary displays the right information.

## 5 The main page

The main page (the root directory “/”) shall present a list of plays scheduled for the near future at Benedum. Create a page as shown in Figure 1. You can use the HTML code that you created to show movies in the Netflix lab to create the same visual effect here. Create this page in the templates directory of your Flask project.

Each play is a clickable link. If the customer clicks on it, the customer is sent to another page, such as MrsDoubtfire.html, which contains more information about that given play, as discussed in Section 3 below.

## 6 Detailed information about a given play

If the customer clicks on one of the plays shown in the main page (Figure 1), your server api shall present another page, containing details of that given play. An example of such page is shown in Figures 2 and 3. YOU DO NOT NEED TO CREATE THE PAGE EXACTLY AS SHOWN IN THOSE FIGURES! However, there must be a “Buy Tickets” in those pages, so the customer can go ahead and make a purchase.

You can get all the text and images by visiting the actual benedum page (link below)

<https://trustarts.org/calendar?genre=Broadway%20&%20Musical%20Theater>

## 7 Buy Ticket Page

If the customer clicks on the “Buy Tickets” button discussed in Section 3 above, then your server api shall take you to another page where you can choose a seat zone and section, time and date of the show, along with your name. An example of such a page is shown in Figure 4.

By making all the necessary selections in that page, the customer can proceed to buy the ticket (discussed in Section 5)

## 8 Transaction Information

By clicking on the “buy tickets” button discussed in Section 4, your api shall take the customer to a final page, where the whole transaction information is provided, as shown in Figure 5. The ticket price can be hard coded here to \$XYZ.00. There must be a link in that page to allow the customer to go back to the main page of your site.

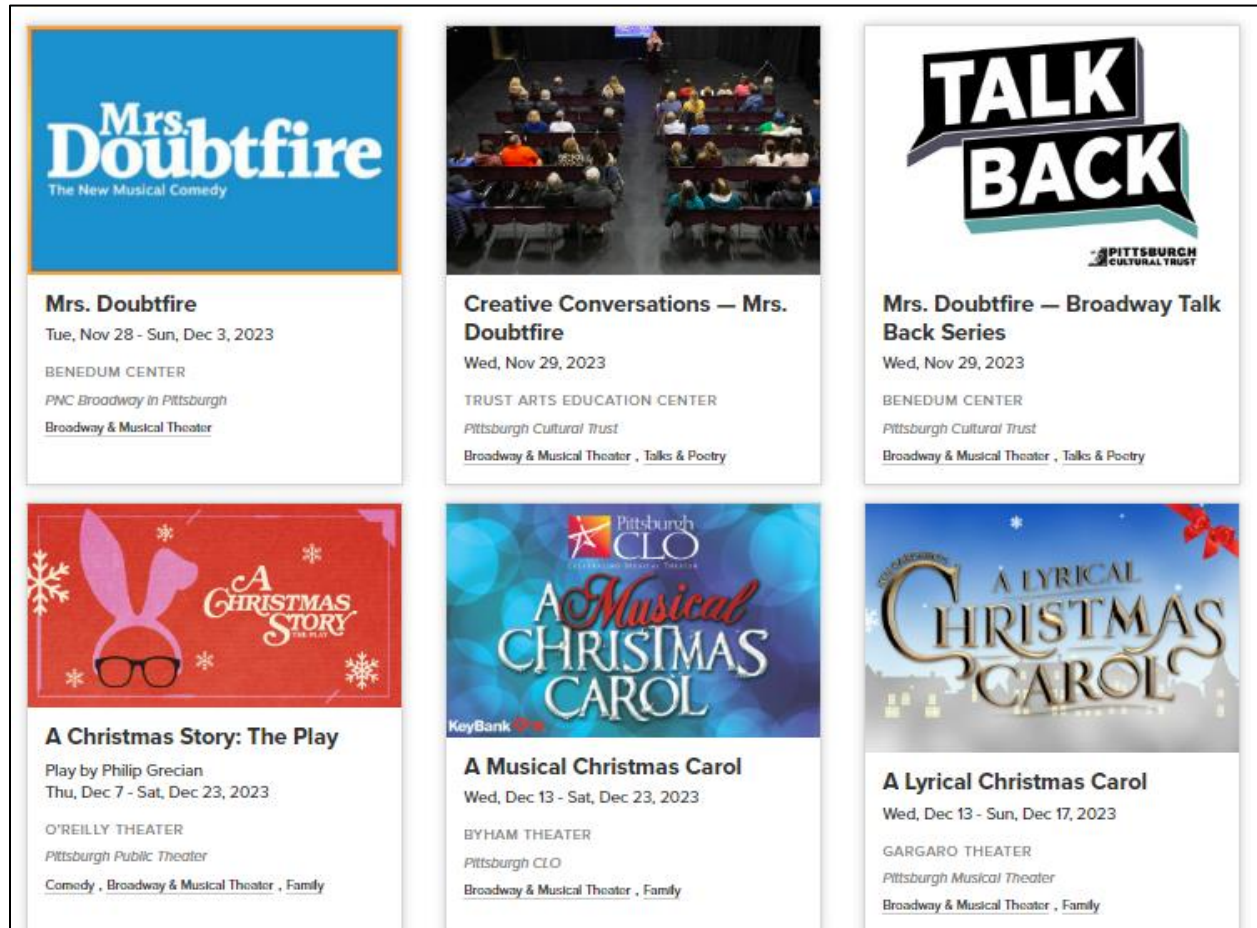



Figure 1: The main page served by your Flask project





## Mrs. Doubtfire

Tue, Nov 28 - Sun, Dec 3, 2023

"What a hilarious tour-de-force this show is." – Buffalo Theatre Guide

"A hilariously fun time showcasing the power of fighting for your family in whatever form it takes..."  
– DC Theater Arts

A new musical comedy about the things we do to stay together.

Everyone's favorite Scottish nanny is headed to Pittsburgh! Rob McClure will reprise his Tony-nominated Broadway performance on tour alongside co-star (and real wife!) Maggie Lakis in this

[Read More](#)

[Get Tickets](#)

PNC Broadway in Pittsburgh  
**BENEDUM CENTER**

Broadway & Musical Theater

f t e

Figure 2: Detailed information about a given show



## A Musical Christmas Carol

RETURNING as  
Ebenezer Scrooge  
**MICHAEL CERVERIS**

KeyBank

A Musical Christmas Carol

Wed, Dec 13 - Sat, Dec 23, 2023

**A HOLIDAY TRADITION FOR THE ENTIRE FAMILY!**

We've taken the spirit of the season, wrapped it in your favorite yuletide melodies, and tied it up with all the holiday magic and wonder of your childhood. Join Scrooge, Bob Cratchit, Tiny Tim, and a host of colorful characters for **A MUSICAL CHRISTMAS CAROL**. With dazzling special effects, holiday charm, and ticket prices even Scrooge would approve of, this Charles Dickens classic is a wonderful way to celebrate all the tradition of the season.

**Accessibility:** [Braille](#), [Large Print](#), [Wheelchair Seating](#), [Sensory Friendly](#), [Signed](#), [Assistive Listening](#)

Note: All services may not be available at all performances. Click the link above for accessible performance schedule or contact customer service for further assistance.

[Get Tickets](#)

Pittsburgh CLO

**BYHAM THEATER**

Broadway & Musical Theater, Family

f t e

Figure 3: Detailed information about a given show

Zone Green Sector Righth

Customer Name:

Date and Time:

Figure 4: Seating selection, date and time, customer name

Where:

- Zones: green, blue, purple, and orange
- Sections: Right, Right Center, Center, Left Center, Left
- Time and date: just a text field, you manually type your selection here

## Transaction completed:

### Ticket information:

- Show: Christmas Story
- Date: 11/20/2023 - 7:00pm
- Seat Zone: green
- Seat Sector: right
- Customer Name: paulo brasko
- Ticket Price: \$XYZ.00

[Back to the main page](#)

Figure 5: Transaction Information Page

## 9 General code structure and hints

### 9.1 The main python/flask application

Your application shall have the following routes. Their associated functions are not implemented in the code shown below. You must implement it yourself. Most of the implementation here is basically one line code, so if you see yourself placing dozens of lines in these functions, you are making hard on yourself, rethink your approach.

```
@app.route("/")
def default():
    your code here

@app.route("/christmas_story/")
def christmas_story():
    your code here

@app.route("/buy_tickets/<play_name>", methods=["post"])
def buy_tickets(play_name):
    your code here

@app.route("/transaction_page/<play_name>", methods=["post"])
def transaction_page(play_name):
    your code here
```

## 9.2 The HTML pages

Your application shall contain the following HTML pages:

- Buy\_tickets.html
- Christmas\_story.html
- Main\_page.html
- Transaction\_page.html

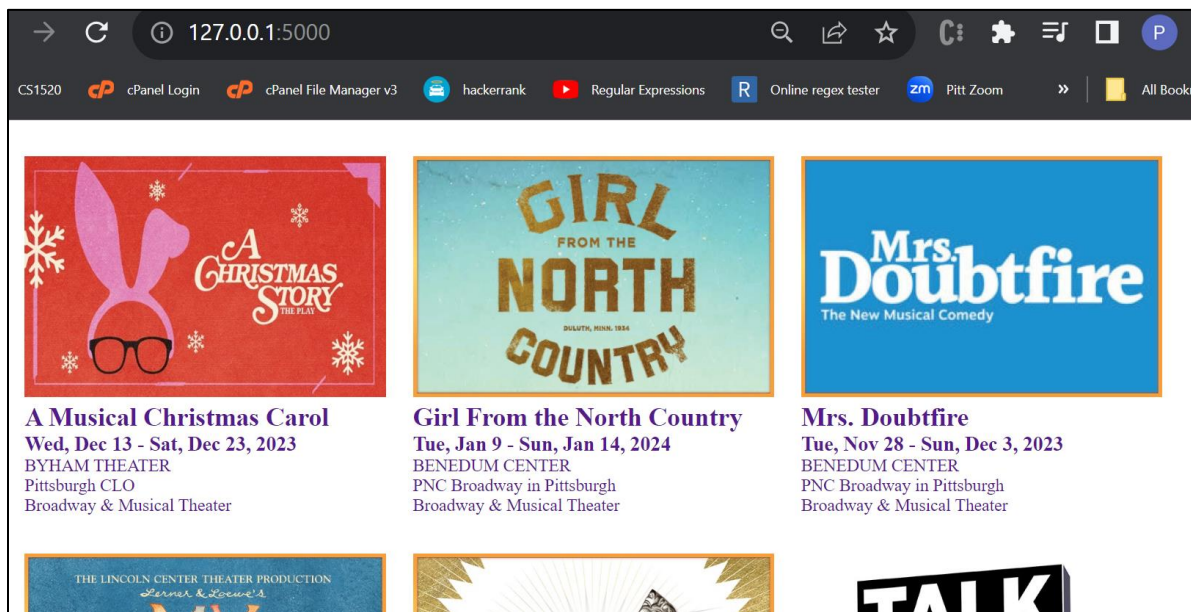
Note #1: you do not need to create pages for all the plays shown in Figure 1. Just create one page. Knowing how to create one shows that you can create all the other ones in a similar way, so do not need to create them.

Note #2: in the places that you need to add an URL reference in your pages, use jinja2 as discussed in our hands-on exercise: Examples of such situations are: in the form “action” field, instead of doing action=“/somepagehere/”, use action=“{{transaction\_page}}”, where in this case, transaction\_page is a variable that contains the URL.

Another example would be: <a href="{{refToMainPage}}">

## 10 A Benedum API Example

The figures below show my API pages, please pay attention to the URL found in the top part of the Browser.





127.0.0.1:5000/christmas\_story/

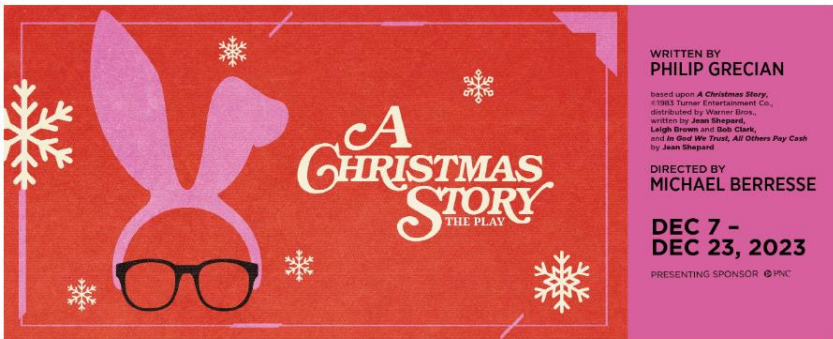
CS1520

cPanel Login

cPanel File Manager v3

hackerrank

Regular Expressions



WRITTEN BY  
PHILIP GRECIAN

based upon *A Christmas Story*,  
©1983 Turner Entertainment Co.,  
distributed by Warner Bros.,  
written by Jean Shepherd,  
Laigh Brown and Bob Clark,  
and to God We Trust, All Others Pay Cash  
by Jean Shepherd

DIRECTED BY  
MICHAEL BERRESSE

DEC 7 -  
DEC 23, 2023

PRESENTING SPONSOR ©PNC

### TIMELESS COMEDY RETURNS TO PUT YOU AND YOURS ON THE NICE LIST THIS

Back by popular demand, last year's holiday hit becomes your newest yuletide tradition as Ralphie Parker and company make their triumphant return moments made famous by the movie, from disastrous family meals — to sticky situations in the schoolyard — to fateful and funny visits to Santaland. This production is not just a Christmas story, it's our Christmas story.

**Rating:** PG for allusion to explicit language, brief and mild violence on stage, themes of bullying, and romantic innuendo. Suitable for most audiences.

**Run Time:** 2 hours, 20 minutes

**Accessibility:** For personal assistance selecting accessible seats or for more information about accessibility for a person with a disability, please contact the box office.

Buy Tickets

127.0.0.1:5000/buy\_tickets/Christmas%20Story

CS1520

cPanel Login

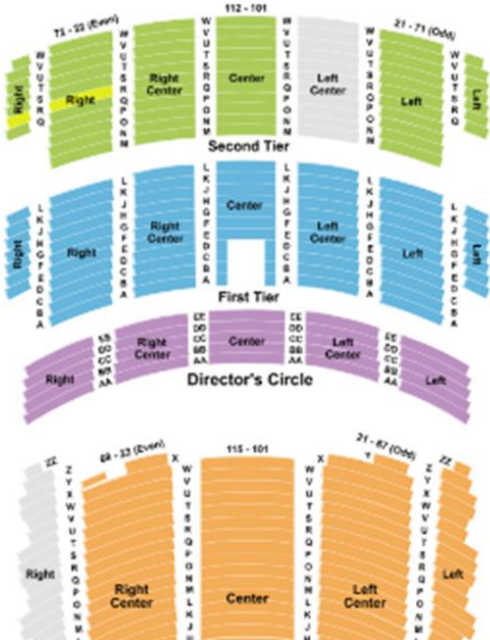
cPanel File Manager v3

hackerrank

Regular Expressions

R

## Buy tickets for Christmas Story



Zone Purple Sector Center

Customer Name:

Date and Time:

Buy Tickets

←

→

↺

127.0.0.1:5000/transaction\_page/Christmas%20Story

CS1520

cPanel Login

cPanel File Manager v3

hackerrank

Regular Expressions

F

# Transaction completed:

## Ticket information:

- Show: Christmas Story
- Date: 11/20/2023 - 7:00pm
- Seat Zone: purple
- Seat Sector: center
- Customer Name: paulo brasko
- Ticket Price: \$XYZ.00

[Back to the main page](#)