Creating and Processing Web Forms with Flask

by Mateo Prigl

Start Course

Bookmark Add to Channel Download Course

Course Overview

Course Overview

[Autogenerated] Hi, everyone. My name is Mattel and welcome to my course on creating in processing Web forms with flask. What forms are the main point of interaction between the client and the Web? Server in this course will learn how to create them and process disability information with the flask Web framework. S W T. F. Beckett will allow us to utilize all of the form processing functionality from the W T forms. Some of the major topics that will cover include the creation of four muse processing form, data input, validation, file uploading and Ajax requests. But end of the course you'll know howto handle all kinds of four men put inside of the flask Web application before beginning. Of course, you should be familiar with the basics of bison programming language and HTML. I hope you'll join me on this journey to learn about flask Web forms with creating and processing Web phones with flat scores at plural side

Creating Web Form Views and Templates

Overview

[Autogenerated] Hello and welcome to this course about creating and processing Web forms with flask. My name is Mateo Prado, and I will guide you through the process of using flask to create maintainable insecure forms to make learning more intuitive. We'll take a look at the case study off Web shop called Global Man Takes, which we're going to create from the ground up. This will allow us to see how the knowledge from this course will be well, you will. In the process of creating professional Web applications in this morning, we're going to create all of the templates and views required for this application to work. We'll get familiar with Web forms and their purpose in the Web. Development Flask has all of the necessary tools to process requests, which include form data. We will need some kind of persistent storage to store user input. In this course, I will use sq flight data base engine. It is lightweight, and we can start a whole database in one file. Bison also has a built in interface for it. I chose to use the relational database mainly because I wanted to show you howto handle relational later with forms after we learned how to store to use their input from the forms, we will notify the user about the successful submission with the flash message. At the end of the module, we will introduce the flesh W T F package, which is the flask implementation off the ability forms. This package will implement a lot of the foreign processing functionality and we'll dip dive into how each part of it works in the following modules. Thank you for joining me and let's get started.

Case Study: Globomantics Web Shop

[Autogenerated] let me show you the government except we're going to build. This is a Web shop where anyone can submit the items they won't sell. I chose the specific example because these sites need a lot of different Web forms to be functional. The most important form is the one that submits the new item for sale. We can see this form by clicking on the Ed New Item link here. We can input the basic information about the item we want to sell, like the name of the item price and the description. After that, we need to upload the image of the item and choose which category and subcategory decided belongs to. Finally, we need to pass the recapture to submit the item. We can see all of the items on the home page with the information we provided in the form. We have another form one left, which filters the items. For example, we might want to see only the items that belonged to the category food, and we want to see the most expensive one. First, you may notice that the page did not refresh. This is because of the a synchronous requests we implement with Ajax. Later, in the course, you can click on the name of the item to see the page specifically made for decide him alone. If you scroll to the bottom of the page, you can see that we have another form for adding comments in the top left corner. Next to the image. We

have the options to edit or delete this item. Editor Adam Link will take us to another form similar to the 1st 1 which will let us update the information about the item. So let's say that you're working as a software developer for the global Mantex company, and you were given a test to create a Web show prototype, which can be shown to the investors. This initial version of the site will need five Web forms. We need a phone to create an item, another one to update an item and one more to delete it. We also need two forms, which will let us submit filters and comments before we start learning about forms we need to create views and templates. Were the site

Demo: Creating Templates and Views

[Autogenerated] Let's weigh the gloom Romantics website. In this video, we'll go through the creation of the basic application logic and templates. Up until the point, we can start learning about the Web forms. You can download all the coat from plural site, but I think that going over this initial process of creating the application will give you more insight in how everything works. I will assume that you already know how to create basic class caps in your operating system _____. Of course, I will use you been to Lennox, but code we create here will work in the same way on any or Wes. I will create a new directory for the project called Global Man Takes Let's enter the directory and create a new virtual environment. Now we need to install flask with the help of Pip Elizabeth. Freeze Commend to save all of the installed requirements to the requirements T X t file. If someone wants to start a project, he can just run Pip Install that our requirements that t X t in their own virtual environment and all the dependencies will be installed. You should now open this great. The directory inside of her code editor. Let's read the Abduct Pie file, which will hold the script for starting up server import flask and create the EP instance at the home route, which will be the index route of the application by default. Our application will soon decide on the look back address from the Port 5000. For this path, it would only return the word home. Let's ride out, say the file and go to the terminal. I will set up the environment variables to tell flask we're working in a development environment and that the start of script will be at that pie. Now we can start the server open the I p with specified port in the browser. Great. Now that we got the server running, we can start creating a glow meant, except I will first create a separate directory named Template. Inside of it, we can create the base template. I will define the basic Method properties for the view port and the Char said, and the title tag. I don't want all of the pages to have the same title, so we'll add the title block before the Global Man tick string. I will use booster to make the website more appealing. If you're not familiar with Bootstrap, you can read about it on its official website. Go to the get boots. Serve that come and scroll down. To see the bootstrap content, deliver network clings instead of installing. Bootstrap will just do the

bootstrap, CSS and Js from the official bootstrap servers, which means that we just have to include the stags in our HTML. Inside of the bag it tag, we can add the main content block. This will be the place in the page where other templates will put their own content. Let me show you what I mean. Create the home template with you. Hold the markup for the home page. Here we need to use some special ginger methods to extend the base template. Ginger is the Templin Ting engine used by flask. Extends. Keyword tells the view that before rendering the page, it should include all of the HTML from the base template first. So if I had a herring with the text, I'm home inside of this content block. This having should be added by the render before we can try this out. We need to inform the home you to render the home template. Of course, the render template needs to be imported from the flesh package. We'll still have a little bit of our own CSS. So let's great the new static directory, which will hold all the static content like images CSS and JavaScript insider with Create the CSS Directory, which will hold our main style CSS file. I want the top bedding of the whole page to be 56 pixels. Now let's go back and include the CSS file into our base template at another link tag to include the Style Sheet Party, a giraffe attribute. Here we will use the euro for method. The Ural four method prints out the full path for the name of the view with passin. For example, let's leave this and go to the home template. Print out the euro. For with the home string argument, you can see that it just brings out slash. This is because the root for the home you is just slash. You may be tempted to heart code euros by yourself without the help of the euro four, but you shouldn't do that. This will become more apparent when you start building bigger APS. What if you serve some content from other domains, then you would have to rewrite the domain name everywhere any time it changes. Okay, Now that we know how it works, we can include aesthetic CSS file. As long as you put your files in the folder name static, you'll be able to access them. This is because Flask has a built in static view that looks something like this. Don't save the file. This is just to show your heart works. For example, if you want to see the style CSS file, you can just write static CSS style, See, assess. And here it is, behind the scenes steady _____ except the file name Pere Meter. When you write the girl with Beth static, anything after that gets saved inside of the file name para meaner. In our case, this is CSS style CSS. When the static view gets the final in perimeter, it goes into the static directory and set back the file we requested. Remove this example of you to access the same euro with the euro four. We can just pass it the static string and add the final in perimeter we wanted depending from the CSS file is working

Demo: Navigation Bar and Items List

[Autogenerated] every page in our application needs to have a navigation bar. So this is what we're going to great inside with the base template, this booster of classes defined the bars, position and color. If you want to know more about these, you can check out the Bush rips official documentation inside of the now bar we can ever do with the class container. Here. We need to define an anchor tag with the class in our brand. This link will contain the logo of the Web shop, and it should redirect us to the home page, create the new director called Images Inside the Aesthetic Directory and download the image by the name logo PNG from the plural sites exercise files. Now we need to add a static image to din of our brand link you conceptual video and try to do this yourself. Create the image take and inside of the source at you it at the Euro four with the file name path to the image. Our street to height to 40 pixels. Now we can refresh the page and here we go. This is the start of our navigation bar. I will rescind the petting of the logo inside of the style sheet just to make it look nicer. Switch the place of these tooling tags. Our custom style sheet needs to override Bootstrap. Now let's add the toddler Aiken. You can cover this from the exercise files, as this is another boost. Our trick. And this one needs JavaScript to see what this does. Get back to the browser and open these factor. Click on the icon of the mobile phone in the corner. This will allow us to see how the page will look like on different screen sizes and device types. As you can see now, we have the cycle in the now bar. This Aiken is only visible when the screen is too small to fit all of the links from now. Bar. We didn't create any yet, so let's do that. Create another Dave under the button. Now we can add a list off the links. The first link reader access to the home page. The second link is for reading of the new item, which for now leads to nowhere. Let's add a container day around the content block and see what we created. Now this is starting to look like a website. I want this link to be highlighted. If I'm currently looking at the page is leading to to do this, I can add the active place to the list item and you can see that it's working. However, this creates a problem since the active classes card coded inside of the base template. All of the pages will have this link highlighted. We need to let the base template know which list item needs to have this active class. This can be achieved by defining a variable in each time played. Let's go to the home template in the final variable called Active Page with the set keyword. Since this is defined inside of the home template, we want the value of the wearable Toby, the String home. Now we can go back to the base template and other condition to pretty active class on Lee when the variable has the string home. If we did everything correctly, we should still see the homing as the current one. Now the only thing left to do is list of submitted items. This will be shown inside of the home template removed, heading and let's create the Dave with the class. Ro. This is bootstraps way to create a grade row in Gustav has 12 equal parts. By writing these glasses, I defined the first column to take 3 12 and the second column to occupy nine twelves with Row in the first column at another day with the Class My four. This will put a

little bit of space on top of the day. Inside of the day, we'll create a heading for filters. We'll create the filters later, but this is where they will replace. Second Column Will hold a list of items will lose bootstraps card company into show them. First, we need to create a day which will act as a rapper for all cards. This Dave has a row class, which means that each card will have the appropriate column size Class L G an M D stand for large and medium, which indicate the column with four different screen sizes. M B class defines margins between each item in the row. Now we can define the day which will hold the card. 100 defines the height of the card card height. Will the 100% of the parent Dave, Since we don't have a database yet. Let's great a placeholder for the image. Don't let the place fall the _____ from the exercise files and place it inside the Aesthetic Images folder. We can now reference that image in the H E mail. Under the image will place the card body and the cart. Further car body holds the title price and the description of the item. The footer will show the category and subcategory. We can see what the card looks like. This is okay, but I don't want the car to accommodate the image, but the other way around. Users will upload images with different kinds of proportions, so we should crop the image. If it's the large making all of the images take up the same amount of space. We can use the boosters magic for this repped image with another Dev and at the appropriate embed responsive classes to both elements, Card image class should have the object fit property so we'll add it into the global style ship. One last thing to do is linked to each item page. We'll make the page for each item later, but for now we should drop the item image and the title with the anchor tag. I will also remove the text decoration for links. I don't like the underline when someone hovers over them, but there's just a personal preference and that's it. We're done with the navigation in the list of items. We can make a copy of the card to see how the home page will look like with more items. This is the base of the side that we're going to build on throughout the course. You can see that it also looks good on mobile devices. Navigation bar is here collapsible to take up less space. If you did not code along, you can download all of the code from floor site when you're ready. I was hearing the next lesson.

The Basics of Web Forms

[Autogenerated] servant. Only static pages is not that interesting. What if you want users to had content to decide? This is where you would need wept forms. To create a form, you need to have a form tag with some input fields. Input fields, sometimes called widgets or form controls, can define what type of information we want the user to send us. These can be text fields, check boxes, radio _____ and others. Usually you will need a submit button at the end of the form to submit all of the information to the server. However, before the form can send the data, it needs

to know where and how descended inside of the form tag. We need to define two attributes. Method in action method will define the http method for sending data action will contain the U. R L that will accept the data forms will usually be submitted with the post method. Post requests are used to send data to the server. All of the information from the foreign fields is collected and encoded inside of the post request body. You can also use forms with the get method. Get requests will upend information to the past it will add a question mark at the end of the or L separating each field with the Empress stand character. But this method can be a security risk you should never use. Get requests for the purpose of adding some information to the database or authentication. Imagine if you try to make sign, inform using to get Method User's password would be appended to the earl is a perimeter for anyone to see. Get requests can be cashed and stored in the browser's history, So this is an obvious security risk also. What if you want to let users upload an image like a profile picture? This image would need to be encoded into binary. Imagine a pending something like this to the U. R. L. However, this does not mean that there is no purpose for to get method in the context of the forms. What if you need a search box, for example, you concert some social network for people with the name Luna. In this case, your input is not changing anything. You're just sending the name string, which can be used to filter out all of the people with that particular name. Maybe you want to use more filters to see only the people that live in your town. There are some other methods, like Putin delete, which are particular useful when great arrest. Well, FBI's. However, HTML standard form on Lee accepts two methods get imposed. To sum up, you would only use get through three of information and posed to send information which will potentially change something on the server. In flask, you can access the girl perimeters through the arts dictionary. Get request. Perimeters will be stored inside of this dictionary as key value pairs. To access the submitted data from the post request, we can use request form dictionary in the same way Jeremy in the next lesson to see how to implement these in practice.

Demo: Requests with Form Data

[Autogenerated] flask offers us a simple way to extract form data from the incoming requests. In this demo, we're going to create our first form for creating new items. We will learn how to create views capable of accepting post requests. We also need to create a form template first, to give users way to input data. To start out, we first need to create a new wheel which will be available when someone clicks on the ed New item link we created in the previous lesson. Path to the new item view will be item slash new and we should render the corresponding new item template. This page will also extend the base template. Let's set the active facial area vault to new item. We can

now update the list item in the base template to highlight the link. When we get to the new item page. Since we're already there, we can make, then protect pointed the new page. We can also change the page title. We did not use this yet since the home page title is just love romantics. But I want to have a different title for the new item page. Okay, now let's create the content additive with the class Ro and another Dave inside of it. This day will define the width of the form. I want this form to take 7/12 of the row and after two places and left so that the form can be near the center of the page, I will also add some space at the top. Inside of the deal, I will let a heading which will describe the purpose of the form and separated from the rest of the page with the horizontal line. Now let's create the form tag itself. We mentioned in the last lesson that the form needs to attributes, action and method method will be post and the action will be the path to the view which will handle the foreign submission. For now, we will add to input fields to the form. The 1st 1 will accept the title of the item and the 2nd 1 will be the text area. For the description, I will make the text area three rows high and remove the option to resize it manually inside with the style sheet, we also need to add the submit button so that the user consent the data surrounded field with the form group Dave and had the phone control class to the field itself. The submit feel doesn't need the rapper do Weaken separated with simple HR element at additional classes to the submit button to make it more appealing our lead labels before each field to show dues or what needs to be in third. Let's refresh the page. Now this looks like something usable. We no need to configure the view to also accept post requests. This is easy to do. You just need to add the second argument to do out decorator. This argument name is methods, and it is a list of allowed http methods for the specific view. Important request object from the flesh package. This will allow us to inspect the current request interview now that we configured everything. Let's see what happens when we submit information to reform. To inspect new code. I usually use the _____ package for Python called P d be important package and set race. At the beginning of the view, this method will stop the execution at this point in code so that we can inspect the current state of the request, say the file and submit some data through the form. If we now go to the terminal, you can see that the service stop working and we're presented with the P T. B interpreter. Now we can see the incoming request. Since we are inside of the view, we can reference the request object. As you can see, the representation of the request object gives us the information of the request that Beth and the http method. You can get the method specifically by typing request method. Since this is a post request, we can seethe sent data inside of the request form Dictionary, it's obvious that this is not a regular python dictionary. Immutable multi addict is the class provided by fax. Oi! Excellent is a Web application. Library and flesh would not be able to work without it. It means tool in German. So what is the advantage of using this class instead of a regular dictionary? Well, in this case, it's the get method to see this in

action. Let's try to get the title from the arguments laced. Seems like it's working. But what if your reference some key in the code, which is not sent by the user like, for example, the password field. If you try to do this, you will get a key error. However, let's try to do the same thing with the get method. As you can see, the missing key fell silently without an error. Let's inspect to get request in the same way, to continue to code execution, right? See and press enter. Change the four method to get and click on the link to add new item again. Of course, we need to let the _____ know that we want to see the page so right, see and press enter to continue the code execution. The form looks the same, but now it uses to get method right in some random data and quickens have made. If I type in request now, you can see that the path is longer. This is because form data is appended to the end of the U. R. L. These perimeters are separated from the rest of the past with the question mark. M percent character is visible between each perimeter request method should give me get Let's see the request form dictionary since this is a guest request, the dictionary's empty. However, in this case, we can use the Ark's dictionary. We can access a single field in the same way. Continue to code execution and remove the set trays. Now that we know how the request object works, we can start writing some code. The form work for usually includes a condition to check which method is used to excise the view. If the method is posed, it means that the users have made to the form. We should them process the form data and redirect news or to some other page process informed. Ate up means doing something with the given information, but we're now we'll just ride it out in the terminal. Since this will be taken from the post request, we need to use the form dictionary before I forget. Let's change the four method back to post. To redirect user to some other site. We will need to import to methods from flask redirect, and you're all four now. We can use them to redirect the user to the home. You save the file and refresh the form template. Let's try to submit something after the submission were redirected to the home page. If I go into the terminal, we can see that the form data has been received and processed

Demo: Setting up SQLite for Storing Form Data

[Autogenerated] any real world application needs some type of permanent storage. Getting form data does not make much sense. If you can a store it somewhere, we need to start a form data permanently inside of the Escalade database. Python has a built in package, which allows us to use it. I created two scripts, one to initialize the database and the other one to print out information from the tables. The reason I chose to include Eskil queries in this course is because you will often be faced with handling relational data from the user input. I will assume that you have some basic knowledge about databases and how to use them in python if you never worked

with SQL or the sq lie three package, don't worry. I created a pdf minute tutorial, which will teach you everything you need to know about. Escalate for this course. Maybe you just need to refresh your knowledge about this. You can glance over it at any point in the course. Whenever you are not certain about some aspect of sq light or its bite on integration, this the scripts are placed inside of the debrief. Older Debbie in it will create all of the tables and initialize them with some random data. Let's run this script. This will create the database file. If you can't see it inside of your editor, check the editor settings. Files like these are hidden in some editors. Now, you can start the second script to see the information from the tables. As you can see, you can choose which they really want to inspect. I will choose all. And here it is. We haven't easy way to check the database while developing our application.

Demo: Application Context

[Autogenerated] Now is the time to start storing the user input inside of our global Mantex database. We will need to employ some knowledge about the flask application context. This will allow us to establish the connection to the database in the correct way. The application context keeps track of the application level data during a request. So basically you can use it to define some state, which will exist only in the scope of the request. Lifetime. We can use this to close the connection to the database at the end of the request, but first we need to create a function which will let us open the connection. We'll call dysfunction, get TB. This function will open the connection to the database and say with inside of the database property of the G object. Jeez, the object provided by the application context, which means it resets on every new request. Of course, we need to important from flask and don't forget to import sq light. The reason we use this object is because we want to check if the connection is already opened and if it is, get that connection instead of creating a new one, we can check this by getting the database a tribute from the G object. If it's not, that means that we need to create a new connection. If it exists, that means that the connection is already established and we just need to return it now. We can easily open the connection from any of you. You may think that this function is redundant, but consider the situation when you have a big application with a lot of helper functions in the code, some of them may need to use the database. So if you create a new connection forever usage, you will needlessly slow down the whole application. There is one more issue we need to tackle the closing of the connection. The connection needs to be closed when all of the calls to the database are done. This will happen at the end of the request cycle. We can create a function for closing the connection. If the connection exists, it will be closed. To put this at the end of the application context, we can simply at their down at context decorator. This requires the exception para

meter. So I returned. The request is over. This function will close the connection. Now we can start a form data into the database at another input field for the price. Instead of just printing out to the terminal, we will save the user input inside of the items stable. If you remember, this is done with the insert command, provide the information from the request and leave the image column blank. We'll also just put one for category and subcategory. Since we did not create those fields yet, we need to convert the price string to float because we define the price field as a real number in the database. Don't forget to commit the connection. Now let's write to submit some new item. Also bid some kind of fruit, since that is the default category. For now. If I now go to the show table script and choose the item stable, we can see the other input is saved. The inside of the database

Demo: Flashing Messages

[Autogenerated] Now that we are able to save data to the database, it's time to also show it to the user will show the submitted items on the home page toe. Let the user know that his admission was successful. We will create a flash message. This reality cards are hard coded inside of the home template. Let's make this item Lee's dynamic. By taking the item straight from the database, execute the select query on the items table. I will also join these two tables to get the name of the category and subcategory. Notice that I added the order by close at the end so that we can always see the most recent items first, This will give us back the least of pupils each. Do people representing one item row from the database. This is okay, but I want to make it more expressive so that we know the exact property were fetching. Weaken it right through a result of the query and create an item dictionary for each item. After this is done, we can just upend the item dictionary to the items list. Now let's best the list the home template. We only need one item card in the template. This will be a placeholder for each item. We can look for the Adams list. We passed the template Instead of this place holder title, we can put the title of each item do the same for description, category and subcategory price. Can we bring it out with the ginger format filter? This will form it the decimal number to have two decimal places and the dollar sign. We can other condition before looping. If there are any items, then we should loop over them. But if there are none, we can write a message. Okay, let's refresh the home page. And here it is. All of the items from the database are here. Even the one resubmitted. Before we have another item, we should great a flash message to let the user No, the item was submitted. Flash message makes it possible to record a message at the end of the request and exit on the next request and only on the next request. Let's include flash from the flats package and create a message for the successful submission. We can include the name of the submitted item two notice. The second

argument is the string success. This is optional and its use for separating different types of messages like warnings, errors and successful ones like these. That way you can show the message in a different color, based on the type it belongs to. We need to configure a secret grief for this to work. Flesh needs some session information and session will not work without secret key. More on this later in the course flash message and should be available in each site. So we need to print this out in the base template about the count in block. You can get the flesh messages with the get flesh messages. Function with categories. Argument will configure the function to give us back the list of two people's, where each double contains the type and the content of the message. If there are any flesh messages, we can look through them. Each message has a type and content. Put the count in the inside of the bootstrap Aller Dave type of the message will condition this second class, which will determine the color of the message. I did not give this message the type success by accident. This is because Booster has these alert classes in different colors. So, for example, if you want the message to be read like an error, the type of the flesh message should be danger. We can also had the close button. Let's add another item to see the flash message and it works. The item has been successful submitted, and it's visible on the home page.

Demo: Introducing Flask-WTF

[Autogenerated] in this them, we'll take a look at the basic usage of the flesh. W t F package flask ability. F is a thin wrapper around the W T forms package. As you can see, it also includes CSF protection while uploading and recapture more about all of this in the next morn Joel. For now, let's reflect their deformed recreated. We first need to install the package from the terminal. Don't forget to update the requirements. W T F defines each form with the separate class. This class has to extend the flask form class, so let's imported. We can call it the new item form. Each form field is defined as an attribute. The form we created currently has a title price and description field. We can define these attributes with special field classes. There is a class for each type of the field. For now, we need the Stringfield, which takes in the basic text input and text area field for description. Notice that important this directly from the W T forms package s stated the flesh stability F is a thin layer, providing just some basic integration. Classes like flask form the first argument of the field class. They finds the label of the field. That is all we're going to define. For now. We also need the importance of midfield. The first argument here is the text written inside of the submit button. Now that we configure the formal logic, we need to render this in the template the form needs to be. Instead, she hated inside of the view. We can then pass this form, object to the template, surrender the field. You just need to reference the feel that tribute from the form

object, you can add any additional HTML attributes like classes inside of the parentheses. This will render the input field equal to this one. To render the label just add label to the attribute name. This label is taken from the first argument we defined inside of the Stringfield. Now we can raise the elements we created manually Do the same thing for other fields. Don't forget to include the classes for the submit button and the Rose attributes for Description Field. Somewhere inside of the form. We also need to implement the hidden tag method. This will bring out all of the hidden fields we defined in the form class. We didn't find any in this form, but W two forms defines one hidden field automatically toe hold the CSF token, and this needs to be included. Submission logic also has to be changed. We're no longer access in the form data from the request directly. We are now using the form object Texas. The data from the field attributes We can just add data tried to submit a new item, and hopefully they should work in the same way as before. When creating the production ready Web forms, things will get complicated fast and W T forms makes the job easier in the next module will deep dive into this package to discover all the things it has to offer.

Summary

[Autogenerated] in this module, we started the creation of the global Mantex Web shop. We created all the templates and views from scratch. This will serve as a great start, important for learning more about Web forms. Forms are usually submitted with two main http. Methods get imposed. Get a pens that submitted information to do you're out while posting close them inside the request body. Hence the leather is more secure. Form data can be extracted from the request object. All of the user input is stored inside of one of the two dictionaries. Getting the data won't be useful without some type of permanent storage. That's why we used sq light to start Information from the form application. Context in flask gives the way to define functions. In the scope of one request, we can use flash messages to inform the user about the successful submission. At the end of the module, we were briefly introduced to the flask W T F package, thinking for watching and join me in the next module to learn more about this package and its advantages

Processing Form Data

Overview

[Autogenerated] Hi. Welcome back to this course about creating in processing where forms with flask. My name is Mattel and in this module will deep dive into the W T forms package and learn how to use it to process form data. W two forms offer US validator classes, which can be added to the fields. We created a form for submitting new items, but we still need to give users the ability to edit them and delete them all of the basic crude operations. Plastic forms can also be submitted with the get method. This will be useful for filtering items will configure everything to let users upload images on the server, so let's get started.

Demo: Input Validation

[Autogenerated] before we started user input. We need to be sure that all of the enter data is valid. Input validation is one of the core concepts of the W T. Florence package. Well at the select fields to the form and validate all of the incoming data. Our flesh W T F form for adding items on Lee has three fields. Let's add the two new select fields for the categories and subcategories. Don't forget to import the select field class. We can read all of the categories from the database inside of the new item view. Let's fetch the result of the query to the categories Variable. Now we have a list of categories represented by two pools with an i D in the name of each category. Reference the category field from the form and assigned this list to the field. Choices do the same for the sub category, but don't forget to include the where close to the query. We don't want all of the sub gatherers to be shown just the ones that belonged to the first category. Now we can replace these cargo. The numbers with the idea of the choice picked by the user Of course, we need to render these fields in the template. As you can see now, we have a select field with all of the category choices from the database. Subcategory field should change based on the chosen category, but we will do that later with Jake weary now that we created the fields, we can talk about the input validation W T forms allow us to well, it Aidan. But with the well date method, last W T F also gives us the East submitted method, which checks if the request uses the post method. This condition is so common that we got a compound method which basically just combines. These two form data will be persisted on Lee. The request descend with the post method, and all of the fields have a valid input. If any of the fields are not well, it the four mirrors list will be populated with all of the errors. So there any errors. Let's bring them out, is a flash message. The type of the message will be danger just so we can have the bootstraps, red collar _____ field. We created courses, the data in the background. This means it changes the type of the data, if required. The default day. The type for this electoral choices is string. When we get this category to pull from the database, I these are not strings there. Integers, and this confuses the field. The solution is to tell the field that we want incoming data to be course two integers.

But what if someone tries to confuse our application by changing the category I d from the source markup? Instead of submitted one of the existing options, a militias user could change it to some other number and submit the form. As you can see, the validation figured out that this is not a valid choice. This is the courtesy of the select field class. If I go to the definition of the class inside of the W T form source code, we can see that it has a pre validate method. This method. Jax. If the day that we received is equal to any of the choices, and if it's not, it raises an error we saw inside of the flesh message. The method is called pre validate because it does some field related validation before the actual validators. We did not discuss w deforms validators yet, so we'll do that in the next lesson

Demo: WTForms Validators

[Autogenerated] we can see all of the available field classes in the official W deforms Documentation 1,000,000,000 Field will create the check box. There are also two fields for storing date and time. Most of these fields, except simple text is an input, but they know what type of string they need to elevate. Take this decimal field is an example. This field will actually rendered input text field by the day. That will not be valid. If the impulse string is not a decimal number, we can use this class sport of price field, import the decimal field class and assign it to the price attributes. Now, if you try to import a random string inside of the price, field will get a validation air informing us that the price field needs to have a decimal value. As you can see, fields already have a lot of built in pre validation, but we can also add additional validators Well. Daters are special callable classes, which instruct the form relegate method what needs to be checked for each field. Every field class has a validators least argument, which you can use to pass all of the required validations. Two of the most important fella. Laters. Are data required, an input required. These elevators will make any text will required. Let's import both validators from the W T forms Validators ad. Deen puts required validator to the title field. You can write a customer message is a first argument. If we now try to submit form without any title, you can see that we get an immediate response saying this well should not be empty. However, this is just Boutros validation when you make some field required validator automatically, as this required a tribute to the field, and then bootstrap prevents you from submission. But what if we raise this manually and something to form? As you can see, the server side of elevation works. Now let's try to submit a few white spaces. Unfortunately, this worked it past the validation. This is because input required on Lee looks at the data before coercion. We can see this in the source code. It checks a fraud ADA sent from the input is truth E, so the white spaces can pass this easily. On the other hand, we have a data required rally later, which also strips the input off all white spaces and then

check. If it's true. Data attribute is actually the chorus value from raw data. This means that the data required validator will be sufficient for the defense against white spaces, so we can add this to the description to. You can also set the maximum and minimum length of the input. We can require the lands between five and 20 characters for the title, with the appropriate message, couple the same validators to the Description field and restrict the description to the maximum of 40 characters. These river elevators are the most important ones. Others are just for special cases, like checking the format of an email. Now let's try to input white spaces in the title field and a short string in the description field. As you can see, the validators are working. Let's re initialize the database before next lesson to get a fresh start.

Demo: CRUD Application

[Autogenerated] Now that we know how to relative income and data, we can create other forms needed for the application will create a template for items which will include the product reviews curd Acronym stands for create Read update in the lead. These are four basic functions off persistent storage we implemented create. By letting users submit new items through the form, we can create a neither mu to represent each item alone. This route will have a neither my d perimeter. By taking this, I d. We know which item we need to show in the template. Let's get the database cursor to get everything about this item from the database, you can see the past this path perimeter to the query use Fetch one to fetch the item. Since only one item should have this, I d. We can now start a result into a more readable item Dictionary. This will be fine if the item exists. However, if someone gave us a non existent I d. This indexing will give us an error to prevent. This will put it inside of the tribe block. If there is no item than set the item toe on empty dictionary now we can render the item template, but I only want to do this if the item exists. If the item doesn't exist, we can redirect the user to the home page. Don't forget to pass the item to the template. This template will be named item HTML. You can copy the basic HTML from the exercise files, but the setup is not complicated. I will set the page title to the title of the current item. The content will again have a row with two columns. Second column will contain a card similar to the cars in the home page, but a little bit bigger with the margin tough class at the Responsive image, just like on the home page. Inside of the card body we can ever heading for the title in price. Description will be inside of the car text paragraph, and I will put the category and subcategory inside of the age four. To make them look different, surround them with the bootstraps batch class. Under this item card, create another card with an outline secondary class. This card will hold the product reviews. Each review or a comment can be a simple paragraph. We can follow this by saying who posted the comments. But since we won't have any

authentication, it will always be anonymous. We can use a jar to separate it from any other comments. Now we can finally add the link for each item on the home page, bypassing the item I d to W we created. Go to the home page and click on any item to see the new item. Paige. It's not much, but it contains all of the information about the item, including Commons. Now we covered, create and read, but we still need to do update and delete. Best place to implement this is on the item page. Since we only need a neither my d to remove it from the database, let's do the delete to you first. The lead path will also include the idea of an item, and the allowed method should only be posed. Since we don't need a template, open the connection to the database to check if the given idea exists. If it does delete the item from the database and commit the changes, we'll also create the appropriate flash message. If the island doesn't exist, will print it out. But we should redirect to the home page. In any case, notice that I only took the title from the result query because I needed for the flash message and the item is going to be the leader. It anyway since the ideas place there's a path Param, Either we only need the submit button to create this form. We still need to render it so Let's Instead, she ate it inside of the item. You impassive to the template, put this form inside of the first column of the row. Since the Earl includes the item I d past the idea of the current item to the form action. Now we can render all of the hidden tax and the submit button. I will also had this own clique attributes which will create the confirmation books with JavaScript. This should do it. Let's write to delete one of the items here is the confirmation box. If I click okay, you consider that an item is successfully removed. Now the only thing left to do is to create a form which updates and item path will include the idea of an item. We want to update notice that we also need to get method because we need to show the form in a new template. Try to create this form yourself. We don't need to complicate things. Edit form can just have a title description and price. You'll need to create a new template and process the form input to update the existing item in the database, joining the next demo to see how I did it and compare it with your solution.

Demo: Form Inheritance

[Autogenerated] Let's create the edit item view. We first need to check if the idea exists. You can do this in the same way we did it in other views, like item and delete item. If the item exists, we will render the edit item template otherwise redirected the home you create the edit item template by coping everything from the new item template. We need to replace a few things. The title will be added, followed by the title of the item. We can do the same thing for the heading, but let's around the title with an anchor tag, which leads to the page of the item form. Action should be the U. R L for the edit item. You with the current item I D. As I said, we will just let views

or update the 1st 3 fields. So let's remove the category and subcategory. If you're selling an item, you can potentially have a need to change the description title or the price, but it's unlikely you will need to change the category. This template will throw in error because we didn't pass the item yet. We also need to pass the form, so let's grade the edit item form, we can create the general item form which will hold all of the common fields knew an edit item Forms can now just define the additional fields they need and heard the rest from the item. For now, we can Instead, she ate this form inside of the editor you and pass it to the template. Let's initialize the data of the fields before rendering. We are taking the existent information from the dictionary we created and putting it inside of the form. This will pre populate the form with the old information to process the income and form input. We can put the elevator on submit method after the form declaration. Let's flash the potential errors just like in the new item view. If the data passes the validation, we can update the item from the database. Best the new form data to the query. And don't forget to pass the item i d. Because of the wear clause, the only thing left to do is to commit the changes and flesh the success message we condemn. Redirect user to the updated item page. I will add a link to the editor view about the delete form, we can add Barton classes to make the link more appealing. If we now go to the item page, we can see that the link appeared in the top left corner. Let's fix this. Overlap with the delete button by adding the display in line property to the delete form tag defined the class in the style sheet and added to the form. Now let's try out the edit form, update any of the pre populated values and submit the changes, and it works.

Demo: Filtering with the GET Method

[Autogenerated] After this point of the course we want, we submit that forms with the post method. However, there are cases when it would be desirable to use the get method in this demo will create a form to filter items on the home page. Filtering will just read the items from the database which have the chosen characteristics. The filter in our case will have four fields, title price category and subcategory. Title field will take a sub string and check if any of the items have it in the title column Category and subcategory are self explanatory. That will just feel 30 items by the chosen options. Price field, however, is a little bit different. This field will let us sort items by the price, with two given options highest, lowest or lowest to highest. So let's create the filter form. Title field will be a simple Stringfield Price Field will be a select filled with two choices. Well, technically three. But I added, the 1st 1 is a placeholder. When nothing is chosen, remaining field definitions should be familiar. Now we can instead _____ this form inside of the home. You We want this form to be submitted with the get method. Since the submission does not change anything in the database to find a solution, let's look at the form class definition from the W T

form source code. This glass is used to create forms. Notice the first initialization argument formed eight Argument passes, the submitted input to the form. Instance. That's how the form knows where to look for the user input. The reason we never had to explicitly do this is because we use flash form class to create forms. Fast form is a subclass off W T forms form. It also says that it formed data is not specified. This form will automatically use the request form dictionary. It'll also use request files, but we'll get the death in the next lesson. Since we don't want to submit the form with the post method, we need to explicitly pass request arcs to this form data. In the first module of the course, we learned that this dictionary holds all of the Ural PIRA meters. We'll also disabled CSR of protection more on this later. Now we need to get all of the category and subcategory choices from the database before Ric Best, the results of the field choices. Let's insert a placeholder option at the beginning of the list. We can now past the form object to the template. I will remove these heading from the template to render the filter form. Action will point to the home you, and this time the method will be get. We can now render all of the fields notice that I didn't use hidden tags method. This is because we don't need CSR. Effort is form, and I even disabled it explicitly in the code. We'll discuss why in the next mortal refresh the home page and submit the form with some random values. Now look up inside of the U. R L Bar. As you can see, all of the well youse were submitted and stored as your L perimeters. Since the home you on, Lee allows the get method. It doesn't matter if you get to this page by clicking on the home link or by submitting the filter form. Both will render the same page, but we want the list of items to be different. Based on the given mural para meters. We can't use the well rate on submit here because this method will only return true if the http method is not get. Since you want to elevate get perimeters, we can only use the validate method. This will validate if to use. Air picked the allowed options from the select fields, but let's also restrict the title. And 2 20 Now, let's see how we're going to filter the items if the validation is past than at filters to the original query else, just to the original query. When I say the original query, I mean the common part of the SQL query that both cases will definitely have, we'll definitely need to select these columns from the item stable and do an inner join. So let's cut this out and put it in a query variable as a string. If the validation fails, we'll just show all of the items. This means that we need to execute the original query, plus the order by Klaus. If the validation past, we need to check which fields were submitted and formed. The final query based on that data Great Tool is named Filter queries and Para Meters Filter careers list will hold the specific query strings with the question mark placeholders perimeters list will hold the actual data that needs to be injected into these strings. Since the fields are not required, we need to check each field separately. I will also stripped the title later, since the string with only white spaces would be considered as true. If the title was submitted, then we'll add like operator to the filter least. We'll also upend the submitted data to

the pyramid. There's list. Don't forget the percentage characters like Operator needs them to know how to check for sub strings. Then we need to check if the category or subcategory were submitted if they were upended. Comparison operator and the data to the respective lists. After this is done, we need to check if the filter releases empty. If it isn't, we need to add the workforce to the original query and join all of the filter shrinks with E and logical operator. Finally, let's check the price option. If it's not submitted, we will order the items by I D. Just like before. If it is submitted, check which option was chosen, and based on that option, create the order by price clause. Now we can execute the query with the given para Meadors execute method expects a tube. Oh, so let's convert the pyramid. There's least let's bring out the query itself just to see if it's made correctly quick on the Globe Romantics logo to get a clean home page. Now we can input some well use and click on filter. We can see that the filter is working. Let's check the query inside of the terminal. This looks like you're correct, Syntex. We can remove the brain function now the filters are finished.

Demo: Uploading Files

[Autogenerated] it's time to let users upload images of the items they want to sell. We'll learn how to enable file uploading and how to serve the uploaded files from the view. To create this, we need three things. First, we need to add the including type A tribute to the form tag. This will allow files to be sent through the post method. When the files are sent, we can extract them from the request by using the request files property. This property holds all of the uploaded files, and they're all stored in memory as a file storage instance. File storage is a file rep a class provided by factoid. We can then use the same method to store to file on the server. Now, before we save the file, we need to check for things the file needs to have a name and that name should be allowed, meaning it doesn't contain some malicious code. Then we need to ensure that lets the type of the file we expect, and the size of the file is not too large. Lucky for us. W two Forms package has a file field. Remember when we were inspecting the flask form class and saw that it passes the request files to the form data as well. Phile Field makes the formal wear off all these uploaded files. Let's have the file field to the item form, since the both new and end it form will need it. We'll lose this feel to upload images for the items. Don't forget to import of Phil class. Let's also important file allowed. Class from the flask W T F file file allowed is a validator, which checks if the type of the file is allowed, create the allowed image extensions list and store it inside of the upcoming Vicky. It's a good practice to put things like these inside of the configuration. Now we can add to file allowed the elevator to the image field and pass it a list containing the allowed file extensions. Second argument. Is there a message? I'll put images only this takes care of the file

type Check class provides us with the special Mex content length. Comfy, icky. This will globally set the maximum upload size 16 megabytes. This means that we got the file size covered, too. Now let's do the first step of allowing file uploads at the encoding type a tribute to both item forms. Of course, we need to render the image field, too. Plus, LA Beauty of Package has one more five elevator named file required. This will make sure that the image needs to be uploaded. Otherwise it will throw an error. That means that we can be sure that the file has a name. The only thing left to do now is to check if the file name is allowed and save the file. I will create an upload colder, since we need a place to store the uploaded images, import the OS packets to get the absolute system path through the project. We can then join this path with the uploads holder and started inside of the image. Uploads Come Fiqi. Now that you have a path to the applause folder, let's process they uploaded image. We can upend the current daytime and some random string to the file name to make sure that every uploaded image has a different name. Import the daytime package and the token hex method from the Secrets package. Now we can use this package is to create a unique file name. Let's do this inside of the new item view after the form gets relegated technical and daytime. This is the format I chose. But it doesn't really matter. I will also generate a random Hexi decimal number Two bites will be enough. The file name Well known me all of these things combined, but I'll still include the original name of the upload that file at the end. Now we have a unique random file name. But since the original file name is included, we're not sure of death. Are these militias to make sure that the founding the safe will use the secure found a method? This method is provided by facts of utilities. We insured all of the file checks from the initial list. Now that we know the file name is safe, we can save one image to the uploads folder. We know the final game of the image, so let's start it in the database instead of on empty string. To serve these uploaded images, we need another view. Let's name it uploads. This view will take in the file name from the path and send it from the applause directory. This is similar to how the static you works, if you remember it from the first marshal sent from director Method will serve the file from the given directory. It needs to be imported from the flask package. Now that we can upload files and serve them, let's create a condition inside of the image source of the items on the home page. If the item has an image willing, the up close view bypassing the image property is a file name. This will serve the correct image from the uploads folder. However, if there is no image, we can still bring the place holder from the static directory. We can do the same thing for the image on the item page. No, let's see the new item form seems like the image label is placed next to the file field. We can separate them with the brake line. Let's write to submit a new item with an image, we can see the uploaded image inside of the new item. This means that the file upload is working and the apple of you is serving the correct images

Demo: Extra Validators

[Autogenerated] not things that I didn't create the image processing logic inside of the additive you. This is because I didn't want an image to be required. When someone is updating an item, you might think that we can simply remove the father required validator, but this would remove the requirement from both forms. The new item form needs to have this field required because we want to encourage users to upload an image when you call the form Relative method Death method is actually calling relatives on each field. Every W two forms field class has its own validate method. This validate method can take two arguments. The 1st 1 is the parent form, and the 2nd 1 is a tupelo off any extra validators. So is the father required. Validator is only needed for the new item form. We can explicitly validate the image field, bypassing the validator through the extra validators pupil. Since we're going to save images in the editorial, too, let's put all of this file saving logic inside of a method we can call it the Seif image upload method. The image field will best is an argument, so let's change the court accordingly. After the images saved, we will return to generated file name. Now. We can just call this method by passing in the form image. The method will save the image and give us back the file name. Since the other form doesn't require an image, we need to explicitly check if the images submitted. If it is, then save it to the disc and get back the file name. If it isn't, we can't initialize the file name to the old one from the database. Now we can also update the image inside of the database. Let's try to update an image off some existing item, go to the edit form and uploaded. Seems like it's working. We can also see the updated image on the home page.

Summary

[Autogenerated] These are the things I wanted to remember from this module. We learned how to validate the user input before story in it permanently to the database. W T Forms package offers us a range of different validators which can make this job easier. We also implemented all of the crude operations for the items. This includes the item page and three forms for creating, updating and deleting an item. If some of the forms have common fields, we can use form inheritance to give the fields maintainable forms don't have to be submitted with the post method We created the filter form with the get method file filled allows us to easily save submitted files. We need to check the file name, file type and file size before we start image to the disk. I hope you find this mortal helpful and I'll see you in the next one

Securing the Form Input

Overview

[Autogenerated] hello again and welcome to this module about securing the form input. My name is Mattel, and in this part of the course, we will take a look at different kinds of attacks people can employ to use the form. Input for malicious purposes will create custom validator to prevent the submission of items which have the subcategory that doesn't belong to the chosen category. Then we will learn about cross site scripting and how to prevent it by escaping the user input. We have already implement that C S R F tokens, but now we'll see why we need them. Esko injection can have disastrous consequences. That's why it's important to always construct queries in a secure way. We'll prevent spamming by implementing recapture test with flask. W T f from security is important, since the forms are the main point of interaction between your application and the world

Demo: Custom Validators

[Autogenerated] In this lesson, we will learn how to create custom validators. W Deforms provides us with a lot of core validator classes, but sometimes that's not enough. Bigger applications will sometimes require special type of custom validations. For example, let's take a look at the form force of 1,000,000 items will eventually implement this subcategory field to change its options based on the select category. However, even if we do that, a malicious user could easily just change the markup of the page. We need to validate if the chosen subcategory belongs to the chosen category. Double deforms package is powerful but also simple and extendable. You can define inline validators inside of the form class. Definition name of the validator should always be relative underscore and the name of the field that needs to be relegated. Validators always need to accept two arguments parent form and the field they need to validate. You can see how this works inside of the well. A date method definition for will check if there are any methods defined with this specific validate, underscore prefix. If they exist, it stores them inside of this extra dictionary. That dictionary then gets Pastor, the parent well, a date method. This method is responsible for calling validate on each field, but it also passes the in one validators to the extra validators of the field. Well, a great method. We have already used this extra validators argument to pass file required validator to the image field validate method inside of the inland validator. We can open the connection to the database and check if there are any subcategories that have the chosen category and subcategory. If there are none, that means that

someone tried to submit an item with a subcategory that doesn't belong to the chosen category. We can then throw validation error, which will be appended to the least of four mirrors. This class needs to be imported from W T forms Validators. Now let's try to submit an item with non matching select field. Seems like the validation is working, since the well Gator is just the method with two required arguments, we could have put this definition outside of the form class. Now we can name it to anything we want. Like belongs to category. For example, this thing needs to be passed to develop taters List of the field. Let's test it. It's still working. If we wanted to have a custom message, this could be achieved with the closure method. For example, the only argument could be the custom message. This method will return the validator itself. Since the validated displaced inside of the method, it will have access to the custom message. Put this message inside of the validation error. You can then pass this method to the elevators list with the custom message. The double duty forms will accept this because it gets what it needs. A method with two arguments. It doesn't matter if that method is returned from a closer or not. We can see the custom error message. This is great, but we should always strive to make custom validators. Reusable method we created is a special case and it always checks the same tables from the database. I want to make this more general to create a more complicated value. Later we can define it as a class. Remember, As long as the field gets the validator method, it doesn't matter how we construct it so well defined. This valley later is a call noble class I will name. It belongs to other field option, innit? Method will take four arguments. The 1st 1 is the name of the table that represents the venerated field. The 2nd 1 is the name of the field that acts as a parent field. In our case, this is the category. Field developer can also passed the foreign key and the message, but these are optional. If the required arguments are not passed in, we can raise a natural bit _____. If they are, we can set all of the past. Well, it was to the instance, Properties. We need a foreign key to know which will to check. But if the argument is empty, we will assume that its name is the name of the parent field with the I. D. Saffet CKs. Let's also define the default message. Now we can finally define develop later method, which will be the class call method. This makes the class call a ble. We will use the same query for the validation, but now we need to pass through things table name and the foreign key can be passed directly. Since this is only available to the developer, we can then pass the chosen option from the select field and the option from the parent field. Get this with the belongs to Variable, which holds the name of the parent field. We should put this inside of the tri block, just in case something doesn't work. If the query fails, we can raise a naturally _____. Finally, add the validation error if the subcategory doesn't exist. Now let's best this new validator to the subcategory field. The table of the field is subcategories, and the parent field is category. I will also had a custom message. Try to submit the wrong options

one last time. It's working. It's always a good idea to make code reusable. Now you can use this elevator class to check the relation between any two fields.

Cross-site Scripting (XSS)

[Autogenerated] cross site scripting attack is a type of injection in which militia scripts are injected into websites. A malicious user could potentially submit some HTML inside of the item description. Brother would usually interpret this as HTML, but seems like the input was escaped. Ginger, 10 plan tic engine, other medical escapes, all of the data pesto, the template. This is why this HTML did not get interpreted. However, if we take a look at the item in the database, it still has this UN escaped HD mail. It's a good practice to escape any user input. Before we start on the server, you never know how you will use this submitted data in the future. Maybe you have a content management system. That kind of site would need to get a lot of custom html from the database, so you would probably use the ginger safe method. This method tells Ginger that you trust this variable so the engine doesn't have to escape it. But what if we submit some malicious code for example, some javascript. You can see that the script got executed because the input is not escaped. This alert will be visible to anyone who is is the home page today. It's just the playful message. Tomorrow this could be something more sinister, like stealing confidential data, joining the next lesson to see how to solve this problem.

Demo: Escaping User Input

[Autogenerated] In this demo, we will learn how to escape user input. Hector Utilities Package has the escape method. This method replaces all of the special characters in the string, which confused to manipulate. HTML. Let's import the method. Now we can use it to escape the description before it gets stored inside of the database. Try to submit an item with some Java script. As you can see, the script didn't get executed, and the description is bring it out in the same way we rolled it. Let's check this item inside of the database. Now here is the difference. All of the special characters get sanitized by the escape method. This way, the browser knows what the show. But it also knows that this should not be interpreted as a steam out at this escape method to the title and description field. We should do this for the new item form every tight inform anyone filters. One last thing to mention here. What if I try to update this new item? As you can see the pre popular, the description is escaped. But since this is an input field, we're seeing the encoded version. The solution is to use the an escape method, which does the opposite of escape. We have already sanitized the value, and we know that this would be injected inside of

the input field. UN escaped the description and refresh the form. You might think that this is not a big deal, because if someone tries to heck us, it doesn't matter what they see here, but this is not for them. This is for the regular users that will probably use the special characters without any malicious intent. You should always escape any input from the user.

Cross-site Request Forgery (CSRF)

[Autogenerated] cross site request forgery attack causes the browser to perform. A nun wanted action on a site while the user is logged in. We don't have any type of andthe indication on the global Mantex website that's outside of the scope of this course. However, this attack is closely related, the Web forms, so let's imagine that we do have some authenticated users. I'm logged in as Mattel and my browser has the cookie related to the global Mantex website, which lets me use it without having to log in every time I can submit any item and the server will know that I submitted it because I'm logged in. Now let's imagine that someone tries to exploit this. For example, a person could inspect the code of the new item form and see which feels air needed for the item creation. Then that same person could create a popular website like a blawg. If I tried to write a comment on that blogged, a new item gets created on the global Mantex website under my user name. How is that possible? The bloke creator created the hidden fields needed for the new item creation. Since I'm already logged in global Mantex website sees this form submission as something usual and valid. Our website doesn't care who submit this form the regular way or from some other site. It is still the same post request. This is bad. This person could create a form for leading all of my items or something worse. You can even use Ajax to submit this form in the background without the user ever knowing that he was hacked. The best way to prevent this is with the so called CSR of token. This token is created automatically by flask W T f. This is why we always included hidden tags method in the forms. This is a random string and it gets generated every single time we request this form page server will check the token when the form get submitted and it will allow it on leave. The submitted token was the same as the one we generated in the form. Now the black owner cannot falsify the requests since he cannot possibly guess the correct CSF token. Don't forget to always include this in any form that that's some change on the server plastic W T f will always remind you if you forget unless you disable it like I did for the filter form. Since the filter form is not changing anything, there is no need for the token here.

SQL Injection

[Autogenerated] SQL Injection Attack Explosive input data to execute queries that application was not intending to execute. To illustrate this, we can use the example from the first module. We have a search box which can filter the social network by the given name. Behind the scenes the server is executing. This query _____ all users where the name is the submitted name. This is really easy to exploit. Instead of a valid name, someone couldn't put something like this. Select Weary will still be executed, but so will the second unwanted query. This will destroy the user stable and erase all of the users from the server. That is why we use the question mark placeholders instead of just a pending the values inside of the query string. If the clear who was constructed like this than the database engine would figure out that you'll intent and refused to allow it always used the pyramid arised queries when you need to include user submitted data

Demo: Implementing reCAPTCHA

[Autogenerated] every popular website will sooner or later. Faced with the problem of spamming, someone could create about which will submit a new item every second this week. Lottery Our sight with the unwanted information to prevent this will implement recapture capture Stands were completely automated, public during test to tell computers and humans apart. It is a test that can be only sold by a human being. The core concept was introduced by an English mathematician, Alan Turing, who proposed this test even before the modern computers were invented. One of the most used implementations of this test is Google's recapture. The test usually asks you to do some tasks that the computer wouldn't be able to complete. The design of this test is complicated, but the implementation is simple, especially since the first W t. F has a native support for it. We need to go to the main Google recapture counsel, just Google recapture. You can't miss it. I'll name it the global Mantex flat scab. Now you can choose what kind of recapture you want. I will use this one here. We need to define the domains. Since our app is only served locally, we need to input the Lubeck address at the local host. Do just in case. Once we submit the information, we'll get these two keys. This is all we need to implement. Recapture with flash liability. F Copy these keys inside of the configuration. Recapture public key and recapture private key. Now we need to import to recapture field from the flats W t f and this is it. The only thing left to do now is to have this field to the new item form we can render to recapture just like any other field. Let's refresh the home page. Here is the recapture. If I try to submit an item without it, the server will throw in error. Let's submit a new item and complete the recapture test. And here we go from Rowan Onley Human skin Submit new items. No more spam.

Summary

[Autogenerated] in this model, we learn how to make forms more secure. We can create custom validators to check special cases. W T forms Validator concept is simple enough to let us do this. You always need to escape the user input to prevent cross site scripting. CSR of tokens protect the form from the unwanted submissions we saw why it's important to use pyramids arise queries To prevent a scale injections first W T f has a native support for recapture. We can use it to prevent the _____ of the forms by militia scripting. Always have these things in mind when creating Web forms.

Improving the User Experience

Overview

[Autogenerated] Hi, My name is Motel. Welcome to the final module of the course about creating in processing Web forms with flask. In this model, we will create our own fields and widgets. Feels used Bridget classes for rendering inside of the template to keep the cold dry. Will implement Ginger Micro's Met Crows will improve the user experience by rendering errors for each form on the site. Subcategory field needs to change Based on the category choice. We will implement this with Ajax and the J Certify flask method. Every item needs to have a form for adding comments. This new form needs to be submitted. A synchronously Ajax will help us to submit the filter form and the common form without refreshing the page, so let's get to it.

Demo: Custom Fields and Widgets

[Autogenerated] W two forms is a powerful package, but its power lies in simplicity. In this Cemal, we're going to learn how to extend and customize the building classes from the package. In the previous module, we implemented this recapture box. Since our website is not in production, let's disable it so we don't have to solve it every time we submit a form in development at this testing key to the EPS configuration and said it too true, This will disable the capture requirement. I will also move the recapture field to the new item form. This is a personal preference. I don't want to render this field in the edit form. Every form needs fields filled glasses hold all of the information about the field, including the submit our data and validators. To render these fields, we need widgets. Widgets are special ability form classes which defined how the

field is going to look like. In HTML, I urge you to go through the W two forms relatively small code base to see how things work behind the scenes. For example, what if we need to create a custom price widget? Let me show you what I mean. I will go inside of the new item template and wrapped the price in put inside of the input group. Dave. This is because I want to prevent another field, which will include the dollar sign indicating that the price that needs to be inputted is in dollars. I will also really find the type of the input to be a number with the step of 0.1 If you're not familiar with this Google, the HTML number input, you can see that the price field now has a better user experience. They immediately know that the price is expressed in dollars, and if you try to write anything inside of the field except numbers, the border will turn red, indicating that this is not allowed. This is great, but now I have to copy all of these Daves and attributes to the edit form. Imagine if you had a big gap with 20 price fields in different forms. If you decide to change something, you would have to do that change on every single field. Let's instead make a price simple twinjet. To do this, we would probably need to extend the import way jet this class renders a basic input field, and it's used as the basis for most of the other input fields. Great. This is exactly what we need. We can see that the A Gmail markup is defined inside of the coal method. This creates a simple input field and passes all of the name arguments next to it. These arguments are the attributes we pass while rendering the field. So this is what we need to override. Of course, this widget class needs to be imported. Copy the call method from the input class. This will lower right the input call method. These lines define some default attributes like an I. D and type. We could just write number here, but let's instead define it as an instance variable outside of the method. Input type needs to be in the class cope. You can see that the text input class does the same thing. I also want to add the step every word Here. These lines just add values and required flags. If there are any now, let's define the markup. I will use the doc's ring here to make it more readable, copy the market from the new item form and pasted inside. We should put an input field here and the percent s placeholder to best the html pere meters. The market class we use here should be imported from the markup safe package. This glass tells Ginger that the market we right here is safe Surrender. Now let's best this new class to the price field through the widget argument, we cannot remove all of the deals, the natural reels from the template. If we refresh the form, you can see that the price we'll still looks the same. The field now uses our custom Bridget instead of his own. We can also create a price field which will inherit all of the behavior from the decimal field. But the difference is this field will have a price simple twinjet removed Avijit argument and replace the decimal field with the price field class. Both forms will now use our custom made price field. In this lesson, we focused on creating widgets. But as you can see, it is also possible to extend field glasses and this goes beyond justifying in a new widget. What if you want to create a custom select field which takes automatically all of the

choices from the database so that we don't have to pester manually in each of you. I will leave this as an exercise for you. See how this electoral class works and try to extend that behavior by adding this new feature you can download the pdf I made about this exercise from the plural side Exercise files. This is where I explain how I created my own custom Select field. There are 100 ways to do this. This is just a example to give you some ideas and inspiration.

Demo: Show Errors with Jinja Macros

[Autogenerated] In this demo, we will learn how to use micros in ginger. Macros will help us display error messages in a nicer way to improve the user experience of the site. Up to this point, in the course of the four, mirrors were displayed as a flash message. Let's remove the flesh message from the new and edit item view. I want to print out all of the errors at the beginning of the form. If there are any, we can create an alert Dave here we can look through all of the fields and right field has following errors. I will make this text bold and add the capitalized filter to the field name. After the brake line, we can look through all of the errors from the specific field. Now let's admit something unacceptable. We have this nice least of all errors organized by the field. I want to have it in the edit form, too. If I copy and paste this, I would have to do this for every new form. This is not where it dry. An acronym for Don't repeat yourself. Showing the errors in this way should be reusable and flexible to accommodate any form. This is why Ginger has Micro's create a new template file and name it. Error messages. Copy the alerted Inside of it. Micro. We'll make this core reusable, and it's really easy to implement. Name of the macro will be error messages, and it will take one argument. This argument contains the four mirrors. Don't forget to close it Now. Instead of referencing these four mirrors, we can just write errors and that's it. To use this macro inside of the new item template, we need to import it from the error messages template. This is similar to what we're doing with bison packages. Now we can use this macro to print out all of the errors by passing it the four mirrors. Let's test this in the form. It's working great. What if we also want to display error messages under the appropriate field? Well, we can use make rose. Of course, I will define another micro inside of the same template. Let's call it field error messages. We'll take in the field, object and check if it has any errors. If it does, I want to print them out in a list format. Each other will be a list item in a red color. Now we can import this macro and put it under each field. Don't forget to pass the field object. I will now submit a description which is too short to pass the validation. And here we go. The air message is right under the appropriate field. We can now import these mackerels in the edit form too. Don't forget to put the fielder macro under each field. Now we have a nice way to display errors for each new form without having to rewrite the same code.

Demo: Dynamic Subcategory Select Field

[Autogenerated] it is time to make the subcategory select field functional were lose. Jake weary to send a secret this request to the server. We also need to implement the category of you which will accept these requests and respond with the belonging subcategories. Since we're going to use Java script, let's define the Java script block inside of the base template. Now we can use this block to include JavaScript in each template. Let's great the Js directory inside of the static. We will define all of the neither JavaScript inside of the subcategory Js file. I will lose Jake weary, so we need to make one small tweak inside of the base template. Jacquier Library is already included as a dependency for Booth stripe. However, this is a slim version. This version will not let us use all of the power of Jake weary. So we need to go to the official Jake Weary Side Copy dealing to demean. If I j Query Cdn, I want this group to be executed when the page gets loaded completely. The fineness of gallery change function, which will be responsible for the subcategory field choices. First, we need to take the chosen category i d from the category field. I'm getting the value from the element with the category I d flask automatically as the film name I d to each field. After we got the chosen category, we need to send it to the server to get a list of the appropriate subcategory choices. This view needs to be defined on the server. The name of the view can be category. It will accept the required category. I d open the connection to the database and select all of these subcategories belonging to that received category. Then we can return this list as Jason. To do this, we need to use the J certify method. Import the method from the flask package. Now let's get back to the Java script file and create an Ajax request. Age extends for asynchronous JavaScript and XML. It is a technique for requesting data in the background with Java script. This will be a get request and the girl will be the path of the new category of you with the current chosen category I d. When we get the response from the server, we first need to empty all of the choices from the subcategory field. Then we can eat right through these subcategories Jason List we got from the server. I want to upend the option tech to the sub categories. Field for each element in the laced. The well you attribute will be the first element from the value list, and the text representing the option will be the second element. This value list in Jason is converted from the subcategory Tupelo we got from the database. Jason, if I will turn each do people into a JavaScript laced, we need to make the special condition for the filter form. Since the subcategory field there has a placeholder option two upend. This place called their option only if the category field has a category filter class at this class to the category field of the filter form. Now we can execute this function whenever someone chooses a different category. Let's also execute it immediately after the page gets loaded. Until now, we will only fetch the subcategories belonging to the first category. Now we can remove this restriction and

fetch all of the subcategories instead. You can also do these for the home view. This had to be done because of the sub categories. Pre evaluation method, which will check all of the available choices. If we didn't do this, this subcategory field will throw a validation error for every subcategory that doesn't belong to the first category. Now we can include the subcategory Js file in all of the templates that contain a form, paste this into the new item template, edit item template and the home template. If I go to the new item form, you can see that we got the correct subcategory choices. Let's change the category to technology. Subcategory Field was updated with the correct choices by requesting the subcategory Jason in the background. We also need to test this in the filter form. If I choose another category, you can see that the subcategory field has changed, and it also includes this place called their option because of the condition we added in the Java script,

Demo: Adding Comments with AJAX

[Autogenerated] we still need to implement a form for writing comments will create the common form and render all of the comments for the given item. Then we will use the power of Ajax to make this former synchronous first. Let's get all of the comments inside of the item. You We only need comments that belong to the current item and we only need the content well. You create the list of common dictionaries Now we can pass these comments to the template inside of the item template. We can check if there are any comments for decide. Um, if there are, we can print them out to the template. If there are no comments, we can put this place holder comment. Now let's great! The new common form This form will have two fields. The 1st 1 is the content and the required validators. 2nd 1 will be the hidden field containing the item. I D. Of course. We also need to submit button Don't forget to import the hidden field class. Instead, she ate the form inside of the island view and pass it to the template. I will also pre populate the item hidden field. Since this is the I d off the current item. Now we can render the form inside of the card body. The form will post the data to the Newcomb interview will create this in a second hidden tech method will now bring the CSR F token and also the item i d. Hidden field from the form definition, eh? Thirties. We can add the content text area in the submit button. Separate the form from the comments with the horizontal line. Now that's great of you. For reading new comments open the connection to the database and instead she ate the form. If the validation past, we can insert the new comment with the escaped content and the appropriate item i d. Then we can redirect the user to the template of the item that the common belongs to. Now try to submit a new comment. If we scroll down, you can see that the common got appended to the list, which means that it's stored inside of the database. Noticed that I didn't create any fresh message to indicate the successful

submission. This is because I want users to submit this form a synchronously with Ajax implement in this will improve the user experience of the site. Create a file name common Js. When the common form gets submitted, we want to prevent the usual post request. This indicates the common form so we'll save it inside of the form, very able. Then we will create an Ajax request to the form action with the four Method which means that we will send the request of the same new common to you. The data we're going to send is the serialized data from all of the form fields. But I will also upend this age experimenter. This perimeter will let us differentiate if the request will send the regular way or through Ajax. Nothing's that their reference to come and form through this class at this class to the farm tag. Now we can create the response on the server. Try to take the age experimenter from the request. If it can be converted to the integer, it means that it's an Ajax request since we sent the number one. If it cannot defaulted to zero after the validation, we can create a condition for Ajax. If this is a major request, we're under a different common template and past the content off the new created comment. This common template will just include the markup for a single comment. No, let's upend the done Cole back. We'll get the comment mark up in the data off the response when the data get submitted, will empty the content text area and instead the new comment After Reform, Border at this class to the age, our element, which separates the form from other comments, we'll also hide this place. Holder comment Just in case that this is the first comment on the page back in the common to you, I will define what happens if someone submits the comment. Without content, we can return a string saying that content is required with the code 400. This is an http status code, which signifies that something went wrong. It will activate the fail call back. Let's create the span with the class common _____ after the submit button. Now we can show this class if the validation fails at the response text, which is the error message we passed from the view. After this is done, we can fade out the error message. I will define the color of the error inside of the style sheet. We'll also disabled the box shadow for the content field. I don't want bootstrap to add the red box shadow, since we already have our own error message. Now, the only thing left to do is to include the common JavaScript to the template. Let's try to head the new comment. You can see that the comet was submitted and added to the list without reloading the whole page. If I try to submit only white spaces, I will see our custom made air message.

Demo: Making Filters Asynchronous

[Autogenerated] Now that we know howto handle, Ajax and flask will have no problem with making the filter for my synchronous at the filter form class to the form tag so that we can reference it from the Java script file. The name of the fire will be filtered. Js workflow is similar to

the common form. Get the form object and extract the needed information for reform submission inside of the home. You will check if the request has an age experimenter. If it does, we want to render a different template. Instead of a home template, we will render the items template copy the part of the home page which looked through the items from the beginning to the end of the if statement, this template will list all of the item cards, emptied the rapper class and fill it with the item template. From the response. I will also add this fading effect Since the page will not be refreshed. The home you're I will now have the submitted perimeters. We can fix this with the Java script, Bush state and replace state. This will upend submitted perimeters to the _____ without refreshing the page pushed. It will also store the states to the browser session history. So the brother will think that you were actually on the new page, even though the page did not get reloaded. Include the filter Js inside of the home template. Now let's try to filter items. The filters are working a synchronously. If you take a look at the girl bar, the perimeters are changing based on the form submission.

Summary

[Autogenerated] These are the most important takeaways from the last module. We can create their own regions and fields by extending the existing W two forms. Functionality. Ginger macros will help us keep the coat dry without the unnecessary repetition. We learn how to use Ajax to create dynamic, subcategory, field and a synchronous forms with a little bit of help from Jake Weary, we implemented these four to comment and filter form. What forms are omnipresent in the Web development. This course give you everything you need to know to process them in flask. Thank you for staying with me till the end, and I want you all the best in your future endeavors.

Course author

Mateo Prigl

Mateo is currently a full stack web developer working for a company that has clients from Europe and North America. His niche in programming was mostly web oriented, while freelancing, working on...

Course info

Level	Beginner
Rating	★★★★★
My rating	★★★★★
Duration	1h 37m
Released	4 Feb 2020

Share course