**Zach Middleton**

**Generic List View**

Using the Generic List in an HTML Template

Generic Lists are a valuable tool for listing an undetermined number of items on a web page. This is done by passing the list of objects (based on the desired model) into the HTML context, which can then be used to create additional elements on the page.

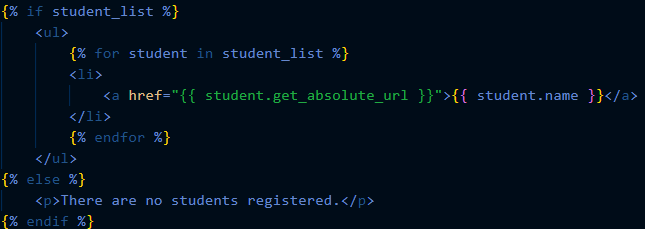
To use a generic list, it must first be imported from django.views, along with any models that will be used, into the views.py file of the project.



Once everything is imported, classes can be written with the parameter generic.ListView, and setting a variable called model to the desired model type.



From there, an HTML template can be created. The template should first check if the object list exists, then create elements for each object in the list. If the list has nothing in it, then additional action can be taken for dealing with the last of a list (like telling the user that there are no items of a certain type).



Once the template is made, a url path can be created in the urls.py file, which will render the template the url is entered.



**Generic Detail View**

Using the Generic Detail View in an HTML Template

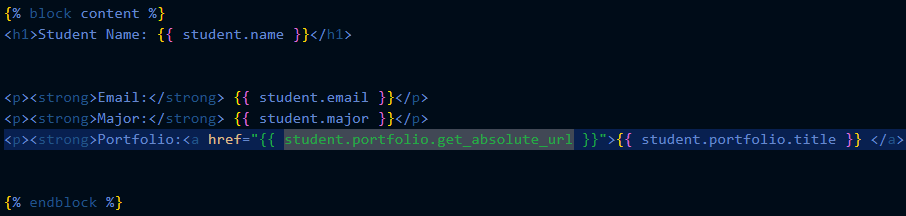
Like the generic list view, the generic detail view is extremely useful for making it easy to create additional web pages that show the details for a given object.

The generic detail view is also part of the django.views file imported for the generic list, so no additional imports are necessary.

Classes are written almost exactly the same as they were for the generic list view, with the only difference being the parameter, which is generic.DetailView.



The HTML template for the detail view is quite a bit simpler, since only one object’s information needs to be displayed. Displaying the values of an object’s instance variables is also quite simple, sharing a lot of similarities with how object data is accessed in other object-oriented languages.



A URL path must also be made for the detail view. The URL for each individual student will contain the student’s id number in the database.



**Replit**

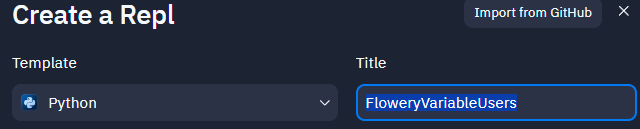
Importing from GitHub to Replit

Replit provides simple GitHub integration so that existing projects can be stored on Replit without the need for copy/pasting them over from a local IDE. The process is quite simple, and has a lot in common with just cloning a repository onto your local computer.

To start, select the “+” icon in the top right corner of the screen, near the account dropdown menu.



A “Create a Repl” window will appear in the middle of the page. In the top right corner of this window will be a button that says “Import from GitHub”, select it.



Then, copy/paste your GitHub repository link in the designated field, click import, and the project will be copied over to Replit.

