**Work List Application**

The Work List Application is designed and implemented through Django and Python by providing a complete User Interface for adding daily tasks that is easily accessible and simple to use. Adding a task is done by writing in the task name, selecting the date due and clicking “Add”. The Sorting method is a dropdown menu that the user can choose to sort the tasks either by date, pending status or alphabetically. There are Edit and Delete buttons for the job, alongside the Set Task Pending button, which implies that the task is overdue; the user can then select completed to complete the task.

The application can be started by accessing the same repository that the manage.py file is in. Then within the terminal, the following command: <pip install -r requirements.txt> should be run; this command installs the required packages. Then this command: <python manage.py runserver>, should be used to run the server for the application. Afterwards, the application can be accessed with the URL: http://127.0.0.1:8000/

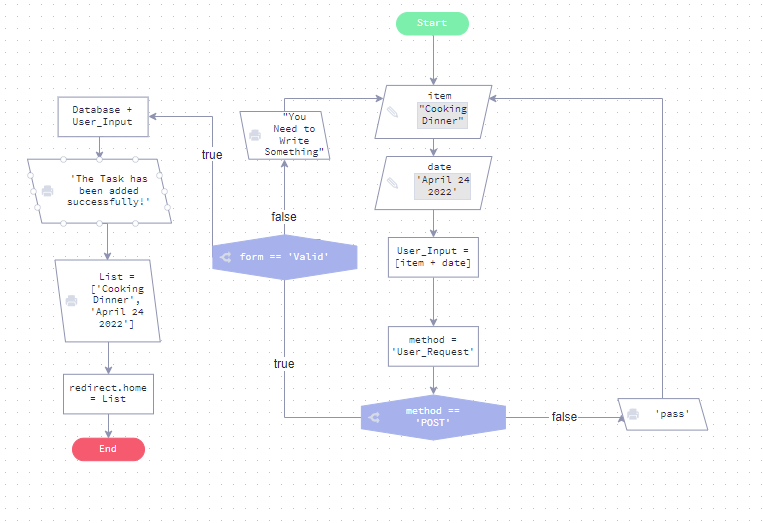
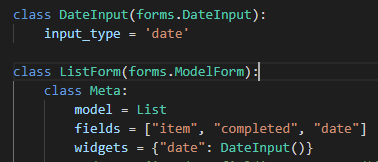
The insertion process starts by receiving the User Input, the inserted data. There are two required inputs; The task and the date; it can be something like; “Cooking Dinner” and 24 April 2022; a form will be created using the name as the item, the date as the date of the item and a unique item\_id will be made for each task. Then the method will be checked if it is a ‘POST’ method. Django will automatically set it to a ‘POST’ method when data is being inserted, thus resulting in data insertion. The input is checked whether it is empty or not; if it is, the user will be redirected to the home page, and if it is not, then that information is passed through, and the item is added to the database. Afterwards, the database will be updated with the new list of tasks and dates. That new list will then be called, and the user will be redirected to the home page again with the updated list. This process represents the insertion flow process of the application.

Figure – Flow Chart of Insertion Process

A screenshot of a computer

Description automatically generated with medium confidenceThe request.method == “POST” checks whether the field is a post field; the variable ‘text’ is then declared for the item posted, representing the task and date. An if condition is used to check if the field is empty; if it is, it redirects the user back to the home page. List.objects.create() creates the new item (Task + Date), and then the form is checked to see if it is valid by checking empty inputs. After that, the new variable form consisting of all the inputs is declared.

Figure 3

Figure 2

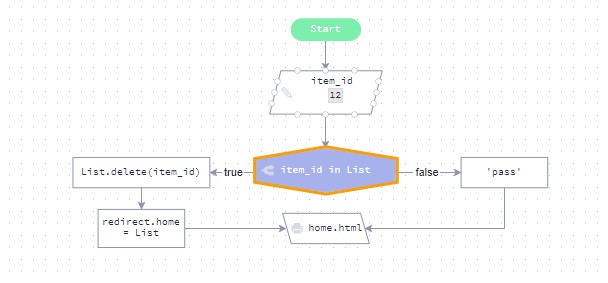
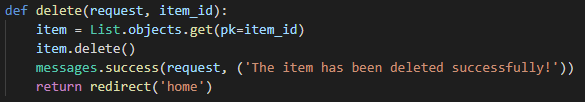
The deletion process of the application works by receiving the item\_id of the required item to be deleted as input. The item\_id is then checked within the database; if the item\_id matches, that item identified by the id will be deleted from the database. Afterwards, the new List of tasks will be forwarded from the database, and the user will be redirected to the home page.

Figure 4 – Flow Chart of Deletion Process

A sorting process is used within the application; the sorting process works by first checking the method of sorting that was requested; within the HTML file, the methods correspond to a new URL. When ‘date’ is selected, for example, the function sort\_date will be chosen, then the function will return a call to the home function with new parameters of “date” added. Then, ‘date’ parameter is added, and the function checks with an if condition whether there is a ‘POST’ method; in the sorting process, in this case, there is no ‘POST’ method, so that it will go to the else statement. The else statement will create a new list of objects, and a .order\_by(sorting) function is used, which is a method of ordering that takes in the parameter sorting, which contains “date”. Afterwards, a new render to the homepage is requested by reloading the home.html page, and the new list is being used, which is “all\_items” this list represents the new ordered list.

Figure 5

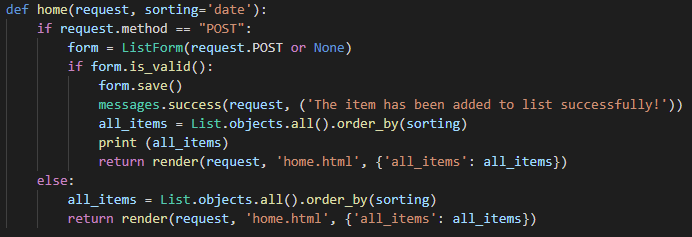


Figure 6

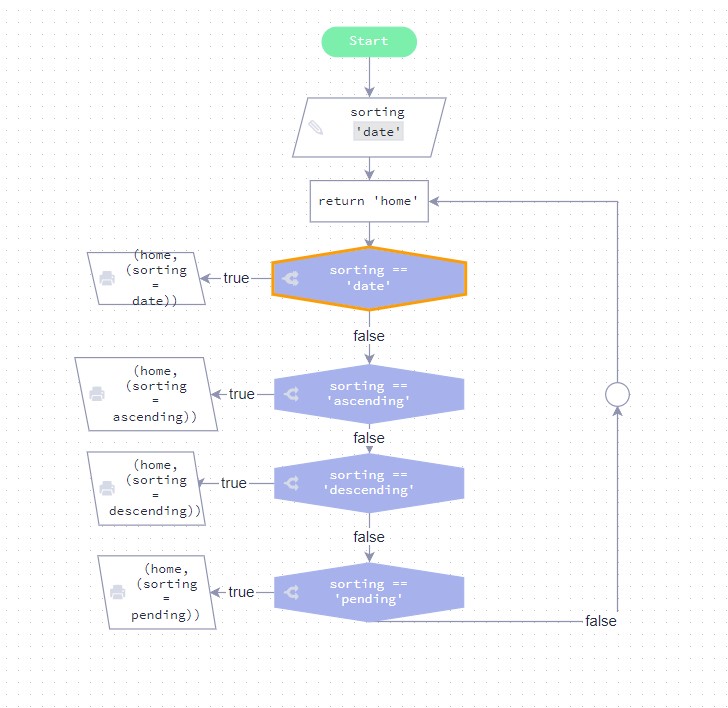
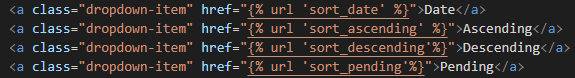
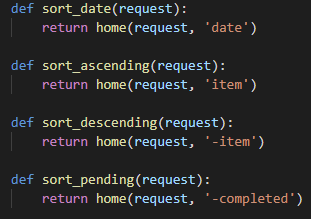


Figure 8

Figure 7

Figure 9 – Flow Chart of Sorting process

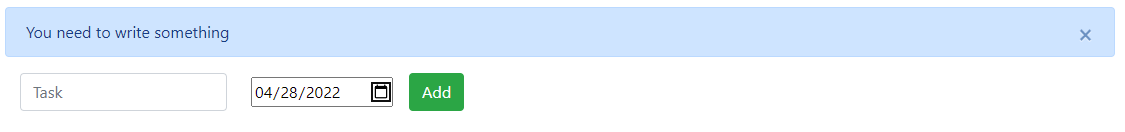
A testing method is applied by using different types of inputs to test out whether the application works as intended or not. One test is done by adding a date and keeping the task field name empty, and choosing to add; this results in the message “You need to write something”, which according to the logic of the program, means that the program identified a missing field which should be the object name, meaning this test worked out correctly as the result is as intended. Another test is adding a task name and not selecting a date then pressing add; this results in the message showing “Please fill in all fields”, which means that the program identified that there is a missing date field that should be entered; thus, resulting in the test planning out as intended. Since both name and date are required fields for the logic of the object to make sense, they must be added as input for the program to have the correct details of the tasks and work as intended.

Figure 10

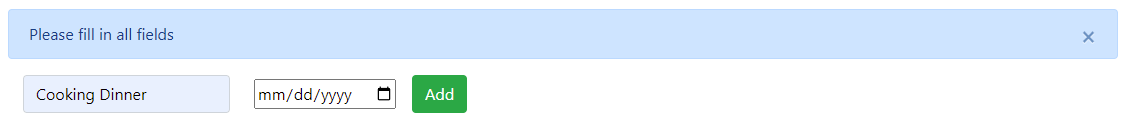


Figure 11

References:

Forcier, J & Bissex, P & Chun, W. (2008) *Python Web Development with Django*. Available from: https://books.google.com.qa/books?hl=en&lr=&id=M2D5nnYlmZoC&oi=fnd&pg=PT31&dq=django+python&ots=vZXECufLTU&sig=gYfr-vsISdMpuz1uj5v7Cs-c9dc&redir\_esc=y#v=onepage&q=django%20python&f=false [Accessed 26 April 2022].

# Dauzon, S & Bendoraitis, A & Ravindran, A. (2016) *Django: Web Development with Python: Web Development with Python : Web Development with Python*. Available from: https://books.google.com.qa/books?hl=en&lr=&id=vKjWDQAAQBAJ&oi=fnd&pg=PP1&dq=django+python&ots=2nSBtzpWyP&sig=swT5OcEAJ5PQMe\_ZE5wZRRdjqUY&redir\_esc=y#v=onepage&q=django%20python&f=false [Accessed 24 April 2022].