Course 12-780
Advanced Python and Web Prototyping for Infrastructure Systems

Fall 2016

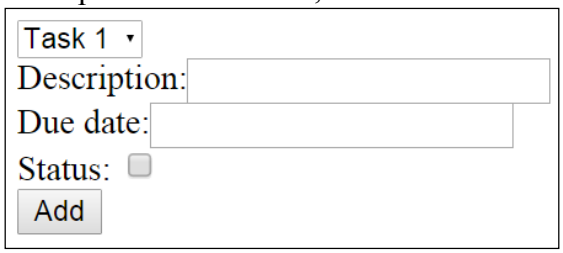
Assignment 3 (Total: 8 points)

Due: Thursday, December 1st, by Blackboard by end of the day. For each day the assignment is late, a penalty of 10% of your total grade will be deducted.

Task 1 (8 point) *A Todo List web application*

a. Please create a Django project named ***TodoListOnline***. Then create a Django application named ***TodoListApp***. Change the ***settings.py*** to include the new application in the project. (0.25 pt)

b. Please create an HTML webpage named ***todo.html***. Save this file in the ***templates*** folder in the Django project. Add the HTML tags so that it looks like the following figure. Change the ***urls.py*** module in the project to show this webpage when user enter this URL in a browser: **127.0.0.1:8000/todo/** (0.5 pt)
Note: There is one dropdown list for the list of tasks, two input boxes for description and due date, one checkbox for the status, and one button.



c. Please create a javascript file named ***todo.js***. Create a folder ***static*** and configure it in the ***settings.py*** so that the Django project can recognize the content of this folder. Save the ***todo.js*** file directly in the ***static*** folder. Then include a reference of ***todo.js*** file in the ***todo.html*** file. Test and see whether the link works in the web browser. (0.25 pt)

d. In the ***TodoListApp*** application, change the ***models.py*** module to add a new table in the database. The name of the table is ***Task***. It should have four columns:

TaskID: a string of the unique id of a task. Should have less than 8 characters.
Description: a string of the description of a task. Should have less than 30 characters.
DueDate: a date value for the due date of the task.
Status: a Boolean value for the current status (true means finished and false means unfinished) of the task.

Note: After creating the database table and columns, please run the

©2016 Xuesong Liu
Please do not copy or upload without written permission

migration functions from manage.py to create this table in the database. (1 pt)

e. Please create a new function in the ***todo.js*** file, with name ***addTask***(). Link this function with the ***Add*** button in ***todo.html***.

   In the ***addTask()*** function, write code to ask users to enter a new ID of a task. Then use a Ajax function to send this new ID to this URL: ***127.0.0.1:8000/checkID***

   In the ***TodoListApp*** application, write code in the ***views.py*** to create a new function named ***checkExistID(request)***. This function should receive the new ID sent by the Javascript code, and check whether the ID exist in the database or not. If exist, return a HttpResponse with string "true". Otherwise, return a response with string "false".

   In the ***urls.py*** module in the Django project folder, add a new URL pattern to link the view function ***checkExistID(request)*** with the URL ***127.0.0.1:8000/checkID/*** (2 pt)

f. Continue with the code written in task e. Come back to the ***addTask()*** function in the ***todo.js*** file, and add an event listener for the Ajax call to monitor the ***onreadystatchange()*** event. Use this event to handle the data received from the Django web server. If the data is "false", show a dialog to the user with message "This ID exists. Please enter a new one."

   If the data is "true", write code to send another Ajax call to this URL: ***127.0.0.1:8000/addTask/***
   This Ajax call should send three data items, ID, Description and Due Date, which are values from task e and two input boxes in the ***todo.html*** webpage, to this URL.

   In the ***TodoListApp*** application, write code in the ***views.py*** to create a new function named ***addTask(request)***. This function should receive the ID, Description and Due date sent by the Javascript code, and save a new record to the Task table created in task 1.d.

   In the ***urls.py*** module in the Django project folder, add a new URL pattern to link the view function ***addTask(request)*** with the URL ***127.0.0.1:8000/addTask/*** (2 pt)

g. Please create a new function in the ***todo.js*** file, with name ***loadTasks***(). Write code to send another Ajax call to this URL: ***127.0.0.1:8000/loadTasks/***
   This Ajax call will not send any data. Add an event listener for the Ajax call to monitor the ***onreadystatchange()*** event. Use this event to handle the data received from the Django web server. The data should be a string of JSON objects, which

©2016 Xuesong Liu
Please do not copy or upload without written permission

contains an array of all records in the ***Task*** database table created in Task 1.d. Save the array of records in a global variable in the ***todo.js*** file. Add the ID of each received record to the dropdown list in the ***todo.html*** webpage. Implement the ***onchange*** event of the dropdown list so that when user chooses a different ID, the description, due date and status of the task will be shown in the input boxes. Link this ***loadTasks()*** function to the onload event of the HTML body tag. This function should be executed when the webpage is refreshed.

In the ***TodoListApp*** application, write code in the ***views.py*** to create a new function named ***loadTasks(request)***. This function should query all records in the ***Task*** database table, and return the array of records in a JSON string as the HttpResponse.

In the ***urls.py*** module in the Django project folder, add a new URL pattern to link the view function ***loadTasks(request)*** with the URL ***127.0.0.1:8000/loadTasks/*** (2 pt)

©2016 Xuesong Liu
Please do not copy or upload without written permission