# Problem 2 – Sprint Planning

**Environment Specifics**

Please be aware that every JS environment may **behave differently** when executing code. Certain things that work in the browser are not supported in **Node.js**, which is the environment used by **Judge**.

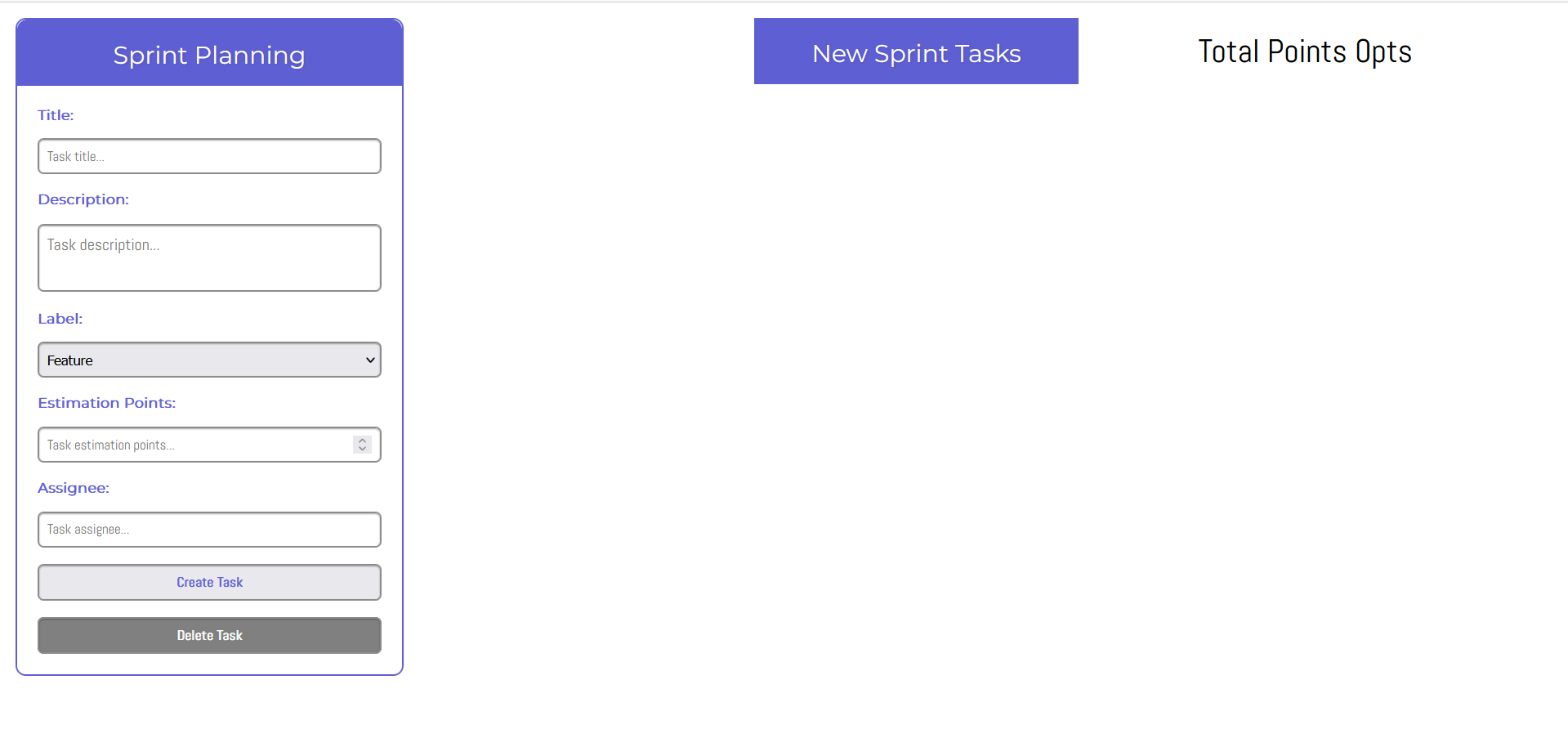
The following actions are **NOT** supported:

* **.forEach()** with **NodeList** (returned by **querySelector()** and **querySelectorAll()**)
* **.forEach()** with **HTMLCollection** (returned by **getElementsByClassName()** and **element.children**)
* using the **spread-operator** (**...**) to convert a **NodeList** into an array
* **append()** (use only **appendChild()**)
* **prepend()**
* **replaceWith()**
* **replaceAll()**
* **closest()**
* **replaceChildren()**

If you want to perform these operations, you may use **Array.from()** to first convert the collection into an array.

**Use the provided skeleton to solve this problem.**

**Write the missing JavaScript code** to plan your next two-week **Sprint**:

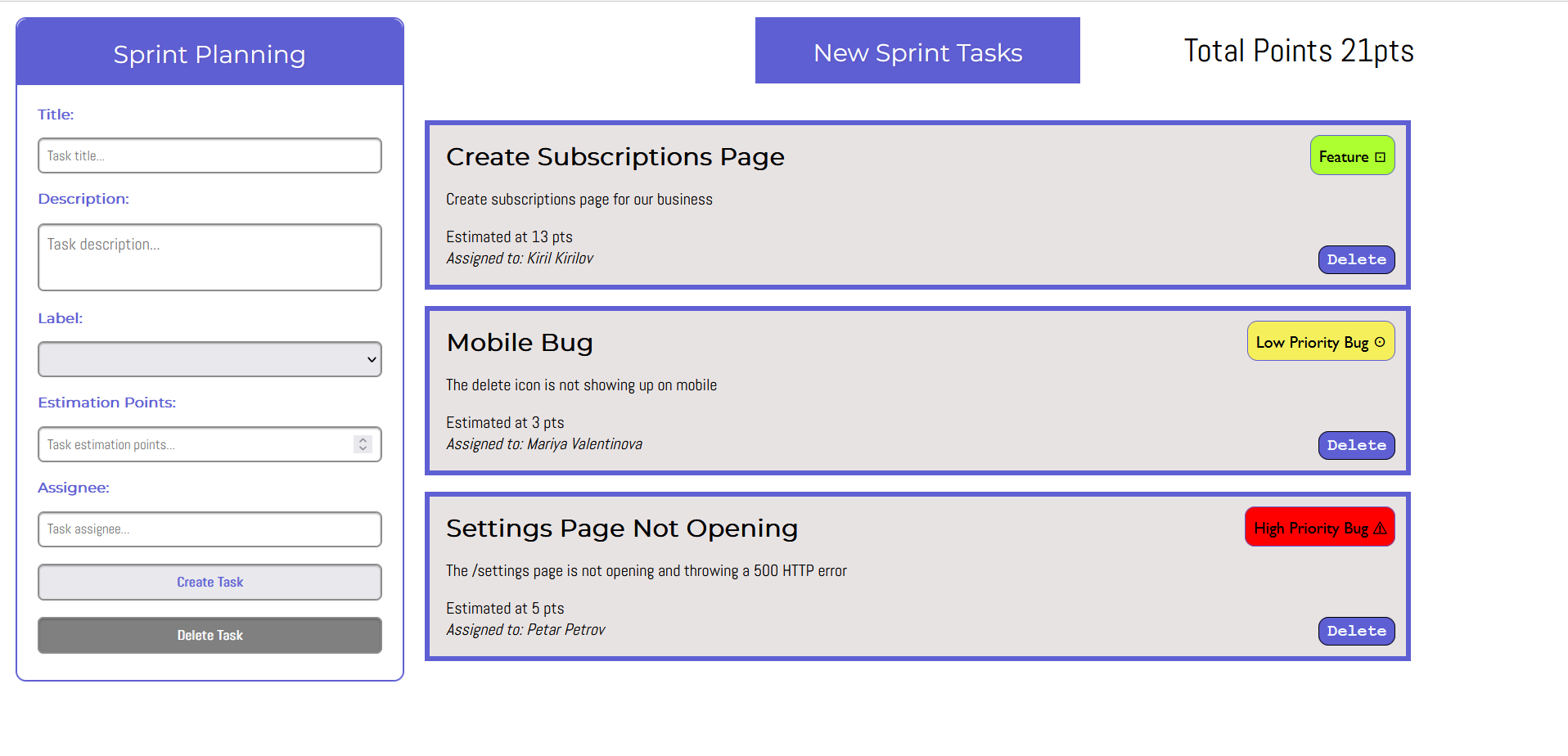


**Create a Task**

* You're provided with a form that contains the following fields: **Title**, **Description**, **Label** (a select field with 3 options), **Estimation Points**, **Assignee**, and **two buttons** – **[Create Task]** and **[Delete Task]**. After the page initialization, the **[Delete Task]** button is **disabled**.
* After the user **successfully fills out all of the fields** and **clicks** on **[Create Task]** **button**, it should create a new **article** inside the <section> with id"tasks-section".
* That **article** has the **following** **HTML** **structure** *(be careful when you create it and add all of the necessary HTML elements and attributes)*:

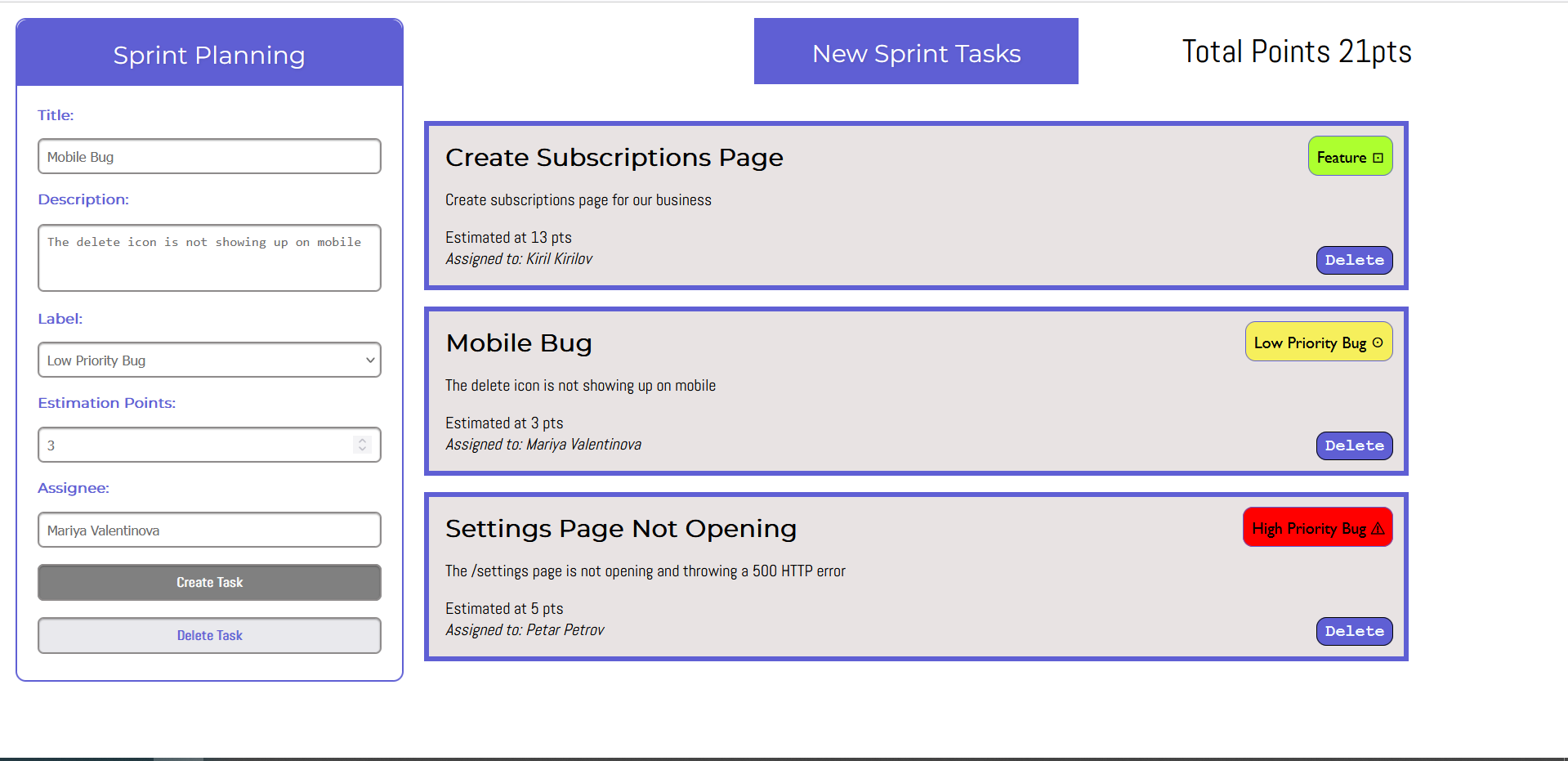
Картина, която съдържа текст

Описанието е генерирано автоматично

* **You have to fill out ALL of the input fields, otherwise clicking on the [Create Task] button shouldn't do anything!**
* **The** <div> **with** class"task-card-label" **has different HTML code icons next to its name depending on whether it is a feature, low priority bug, or high priority bug. Here are the HTML code variations:**
  + **Feature: &#8865;**
  + **Low Priority Bug: &#9737;**
  + **High Priority Bug: &#9888;**
* **The label** <div> **also has an additional** class **that styles it differently, so be sure to add it:**
  + **feature:** "feature"
  + **low priority bug:** "low-priority"
  + **high priority bug:** "high-priority"
* **Each task should also have an** id **attached to the** <article>**, which you have to generate in the following format:** "task-1"**,** "task-2"**,** "task-3" **etc. You will need that later!**
* **Successful creation of a new task should also clear all of the input fields!**
* **An example with all 3 types of tasks:**

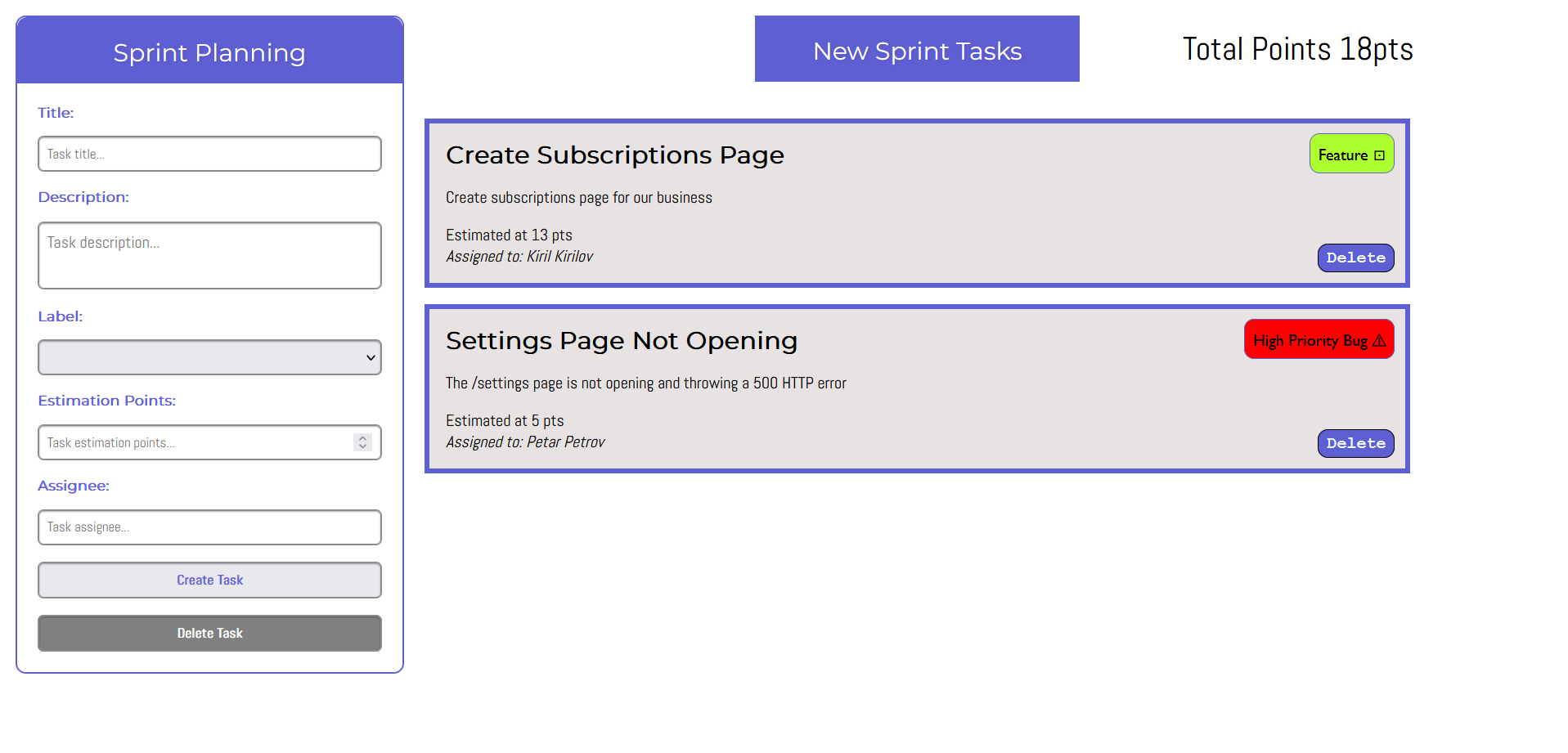
**Load Confirm Delete**

* **Clicking on the [Delete] button on the bottom right corner of a task should load all of the information of the current task in the form on the left.**
* **This action should enable the [Delete Task] button on the form and also disable the [Create Task] button**
* **It should also make all of the inputs disabled!**
* **There is an** <input> **of type** "hidden" **in the form where you should store the task ID you generated when you created the task.**



**Delete a Task**

* **Clicking on the [Delete Task] button should remove the element from the DOM.**
* **Clear out all of the fields and enable them again after deleting.**
* **Enable the [Create Task] button and disable the [Delete Task] button.**



**Total Points**

* **On the upper right corner of the page, there is a total points paragraph that needs to be updated.**
* **When creating a new task add the new estimation points to that counter.**
* **After successfully deleting a task from the form, subtract the estimation points of the deleted task from the total points.**

**Submission**

Submit only your solve() function.