ICP – 5 Angular

Team details:

Sarath Chandra Kolisetty – sk83g@umsystem.edu

Github link: https://github.com/sarath98-lab/spring-2022/tree/main/Web/ICP5

Eeshwara Sai Tota – ettkv@umsystem.edu

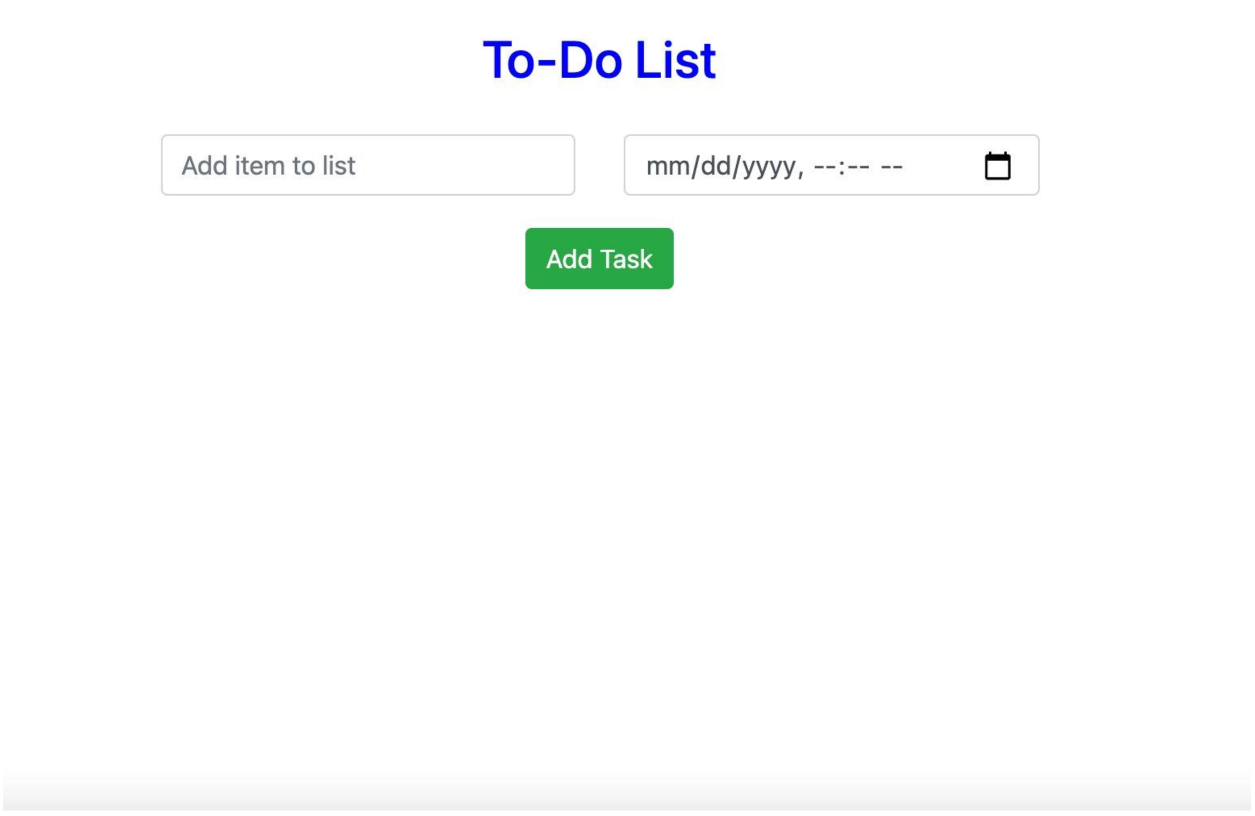
Github link: https://github.com/SaiKicks/WebMobile-Spring2022/tree/main/web/ICP5

Description:

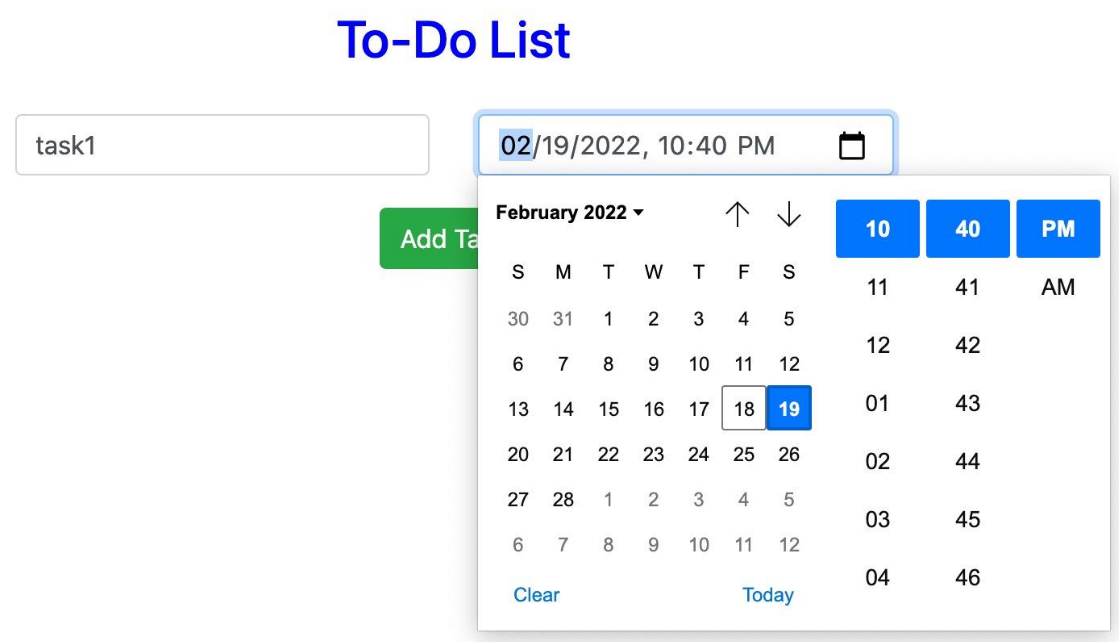
Current ICP helps in understanding Angular. Angular is a popular framework developed by google engineer to develop a single page application. The task involves maintenance of to do list along with the timer. Each task is added with a unique timer. User can add or delete any number of tasks.

To do list:

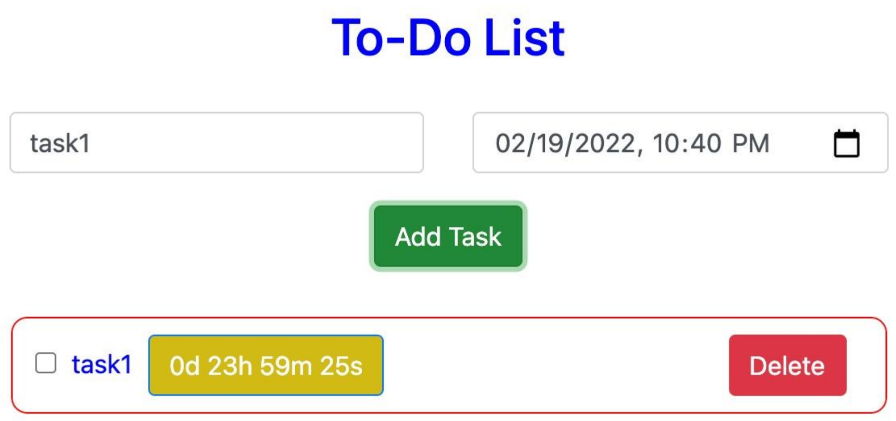
User Interface:



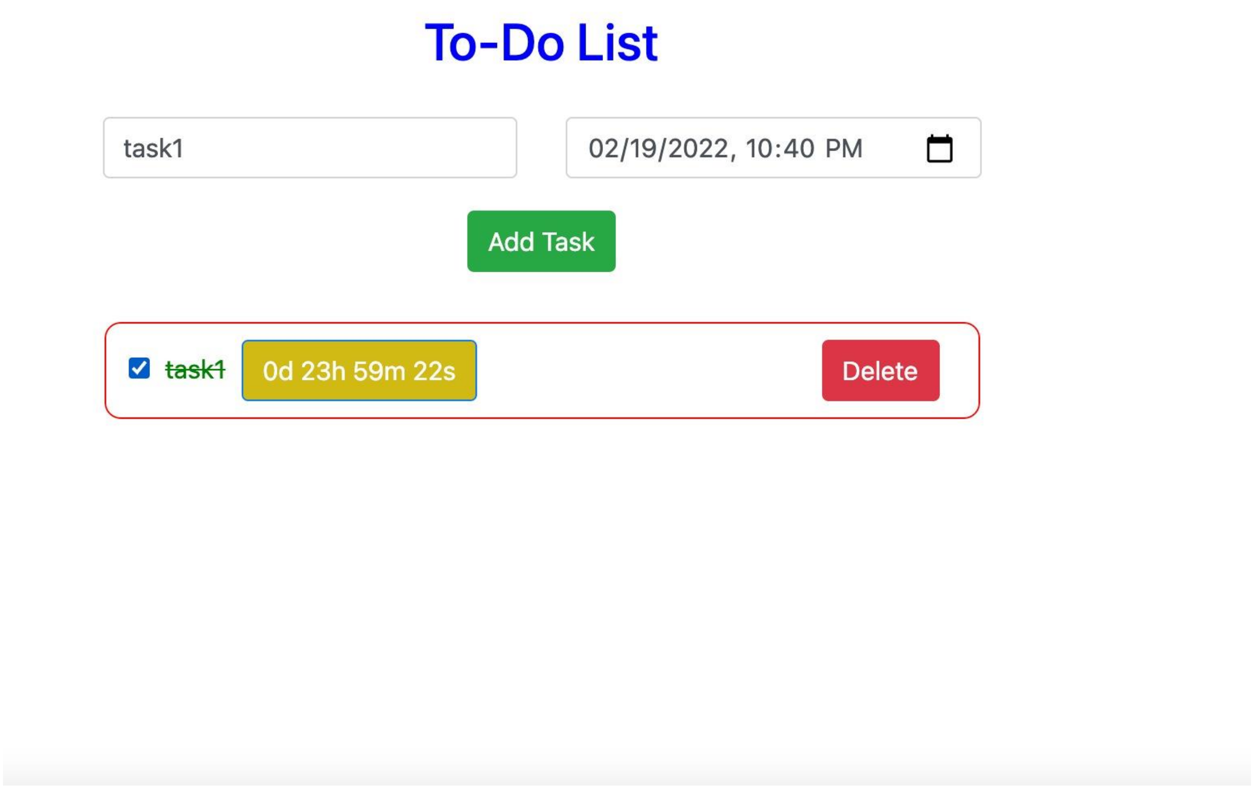
The web page has a text box where user can enter the task and set a timer for a task.



When user clicks to select a time a calendar window pops out. Where user can choose date and time.



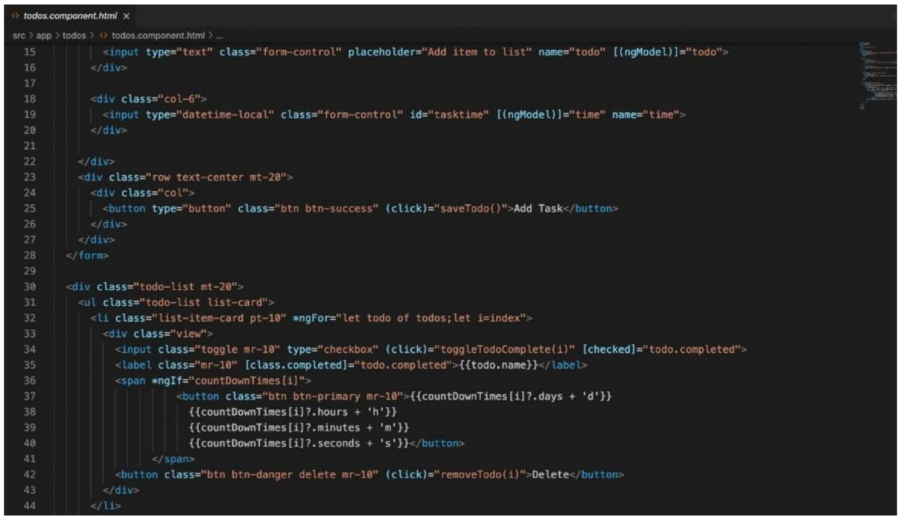
After adding the task, the UI is as shown above. It has a check box indicating if it is completed or not. If user accomplishes a particular task, user check to change the status of the task. The checkbox is followed by the task name. Beside that the timer is shown. At the end of task window there is button for delete.



The above figure shows how the UI looks when a user checks the box indicating he/she accomplished the task.

Source code:

HTML

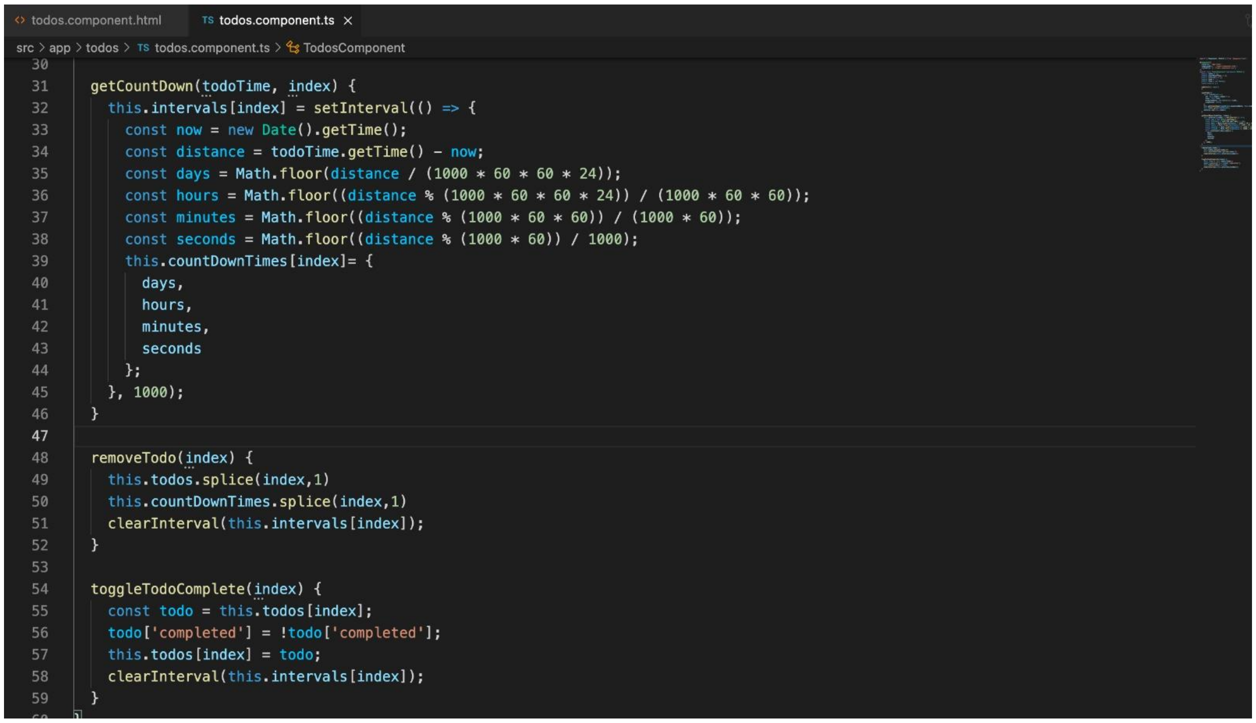


```html
        <input type="text" class="form-control" placeholder="Add item to list" name="todo" [(ngModel)]="todo">
      </div>

      <div class="col-6">
        <input type="datetime-local" class="form-control" id="tasktime" [(ngModel)]="time" name="time">
      </div>

    </div>
    <div class="row text-center mt-20">
      <div class="col">
        <button type="button" class="btn btn-success" (click)="saveTodo()">Add Task</button>
      </div>
    </div>
  </form>

  <div class="todo-list mt-20">
    <ul class="todo-list list-card">
      <li class="list-item-card pt-10" *ngFor="let todo of todos;let i=index">
        <div class="view">
          <input class="toggle mr-10" type="checkbox" (click)="toggleTodoComplete(i)" [checked]="todo.completed">
          <label class="mr-10" [class.completed]="todo.completed">{{todo.name}}</label>
          <span *ngIf="countDownTimes[i]">
                  <button class="btn btn-primary mr-10">{{countDownTimes[i]?.days + 'd'}}
                    {{countDownTimes[i]?.hours + 'h'}}
                    {{countDownTimes[i]?.minutes + 'm'}}
                    {{countDownTimes[i]?.seconds + 's'}}</button>
              </span>
          <button class="btn btn-danger delete mr-10" (click)="removeTodo(i)">Delete</button>
        </div>
      </li>
```

The UI has two inputs one for naming the task and other to set timer for the task.

It also has an iterator to loop through the list of task.

TypeScript


todos.component.html | todos.component.ts

src > app > todos > todos.component.ts > TodosComponent

```
  getCountDown(todoTime, index) {
    this.intervals[index] = setInterval(() => {
      const now = new Date().getTime();
      const distance = todoTime.getTime() - now;
      const days = Math.floor(distance / (1000 * 60 * 60 * 24));
      const hours = Math.floor((distance % (1000 * 60 * 60 * 24)) / (1000 * 60 * 60));
      const minutes = Math.floor((distance % (1000 * 60 * 60)) / (1000 * 60));
      const seconds = Math.floor((distance % (1000 * 60)) / 1000);
      this.countDownTimes[index]= {
        days,
        hours,
        minutes,
        seconds
      };
    }, 1000);
  }

  removeTodo(index) {
    this.todos.splice(index,1)
    this.countDownTimes.splice(index,1)
    clearInterval(this.intervals[index]);
  }

  toggleTodoComplete(index) {
    const todo = this.todos[index];
    todo['completed'] = !todo['completed'];
    this.todos[index] = todo;
    clearInterval(this.intervals[index]);
  }
```

The type script file has all the logic which reacts to the user actions.

It has a separate function for

Adding a task

Completing the task

Remove a task

Count down timer for each task

Learnings from the lesson:

- Angular overview
- Two-way data binding, modules, components in angular.

Issue with the lesson:

Constructors, ngInIt etc features in Angular must have been discussed.

Contribution:

Equal contribution