WEB_ICP5

DHARMA TEJA K

Email: dkbmy@umsystem.edu

GitHub Link: https://github.com/Dharmateja183/Web-and-Mobile-programming-spring-2022/tree/main/Webpart/ICP5

Avinash Reddy T

Email: atfkh@umsystem.edu

GitHub link: https://github.com/avinashreddy3/WebDevCourse/tree/main/WebPart/ICP5

In Class Programming:

NODE JS:

Node. js is best suited for non-blocking, event-driven servers, because of its single-threaded nature. It was created with real-time, push-based architectures in mind and is utilized for standard web sites and back-end API applications.

Node.js is a cross-platform, open-source back-end JavaScript runtime environment that uses the V8 engine to execute JavaScript code outside of a web browser.

ANGULAR:

Angular is a Typescript-based programming framework. Angular as a platform includes:

- A framework for developing scalable web applications based on components.
- A set of well-integrated libraries that cover a wide range of functions, such as routing, forms management, client-server communication, and more.
- A set of developer tools for writing, building, testing, and updating code.

It comes as a set of TypeScript libraries that you can import into your applications to implement core and optional features.

Todo_EventCounter

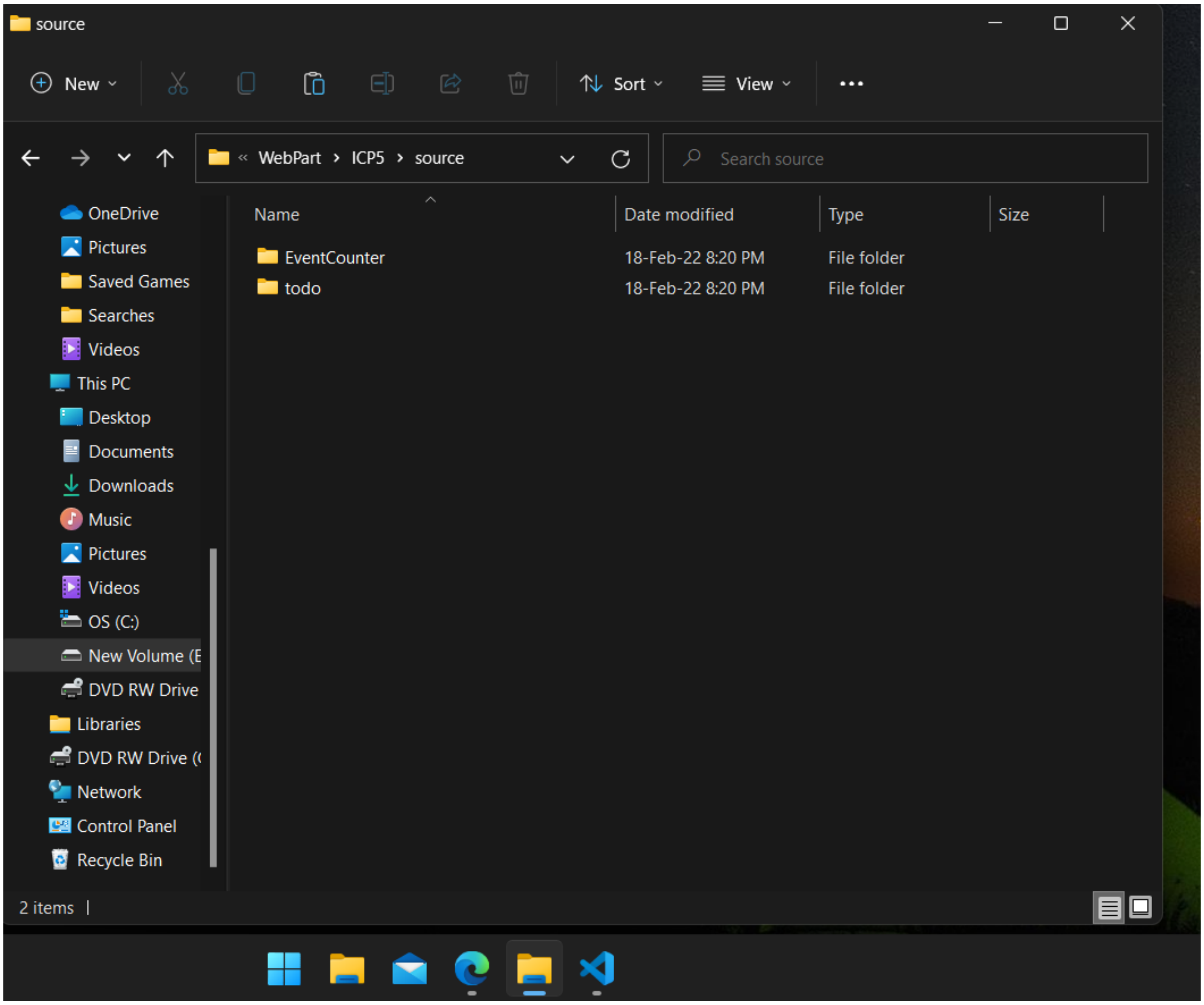
To ensure that people remember a specific task, one can develop a to-do list. It can contain any assignment or goal that needs to be met on time. The to-do list's main purpose is to remind a person of the tasks that need to be completed. It also aids in the reduction of time waste.

If you want to avoid work overload, you'll need to make to-do lists. You'll appear unfocused and unreliable to those around you if you don't use them effectively.

Let's start,

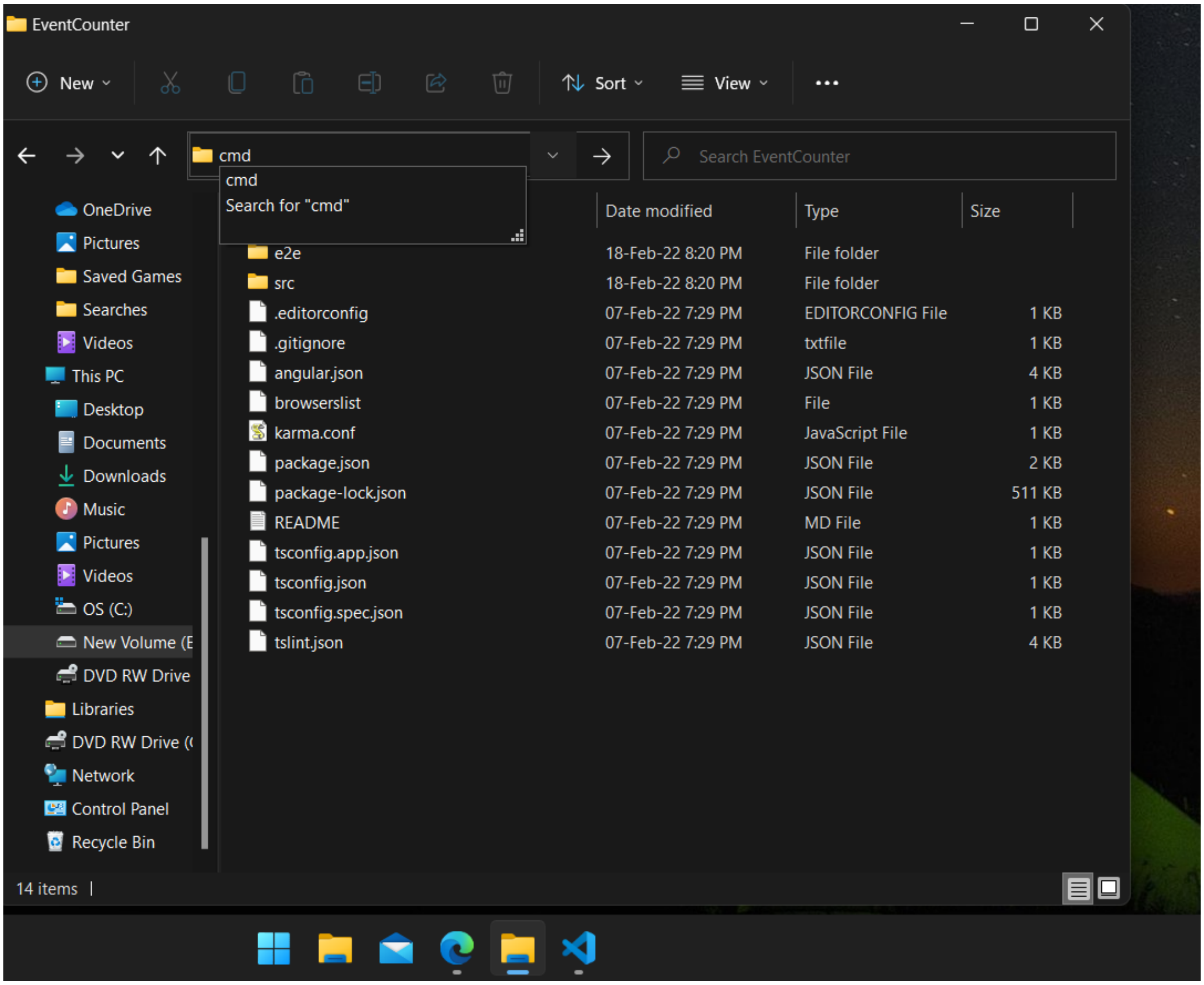
➢ Both the To-do and Countdown Timer in a single application and shown below.

After developing the project,



Name	Date modified	Type	Size
EventCounter	18-Feb-22 8:20 PM	File folder	
todo	18-Feb-22 8:20 PM	File folder	

Open command prompt from the file path as shown below, for reference



cmd

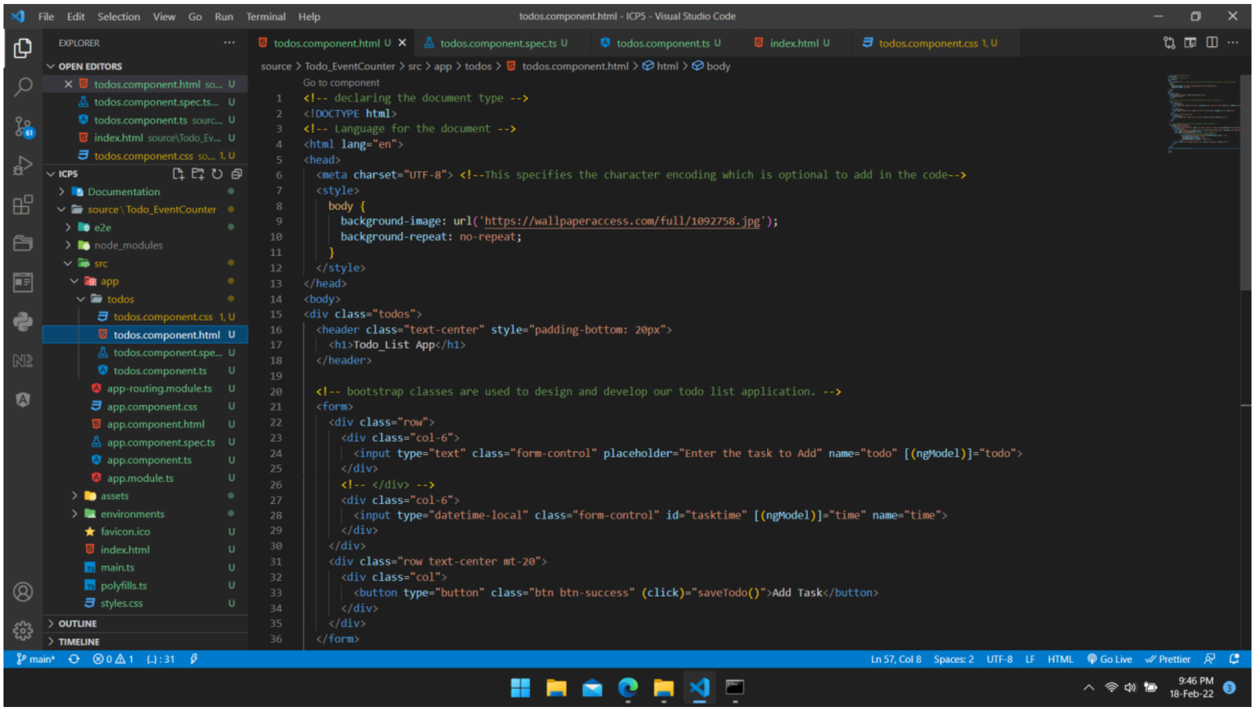
cmd
Search for "cmd"

	Date modified	Type	Size
e2e	18-Feb-22 8:20 PM	File folder	
src	18-Feb-22 8:20 PM	File folder	
.editorconfig	07-Feb-22 7:29 PM	EDITORCONFIG File	1 KB
.gitignore	07-Feb-22 7:29 PM	txtfile	1 KB
angular.json	07-Feb-22 7:29 PM	JSON File	4 KB
browserslist	07-Feb-22 7:29 PM	File	1 KB
karma.conf	07-Feb-22 7:29 PM	JavaScript File	1 KB
package.json	07-Feb-22 7:29 PM	JSON File	2 KB
package-lock.json	07-Feb-22 7:29 PM	JSON File	511 KB
README	07-Feb-22 7:29 PM	MD File	1 KB
tsconfig.app.json	07-Feb-22 7:29 PM	JSON File	1 KB
tsconfig.json	07-Feb-22 7:29 PM	JSON File	1 KB
tsconfig.spec.json	07-Feb-22 7:29 PM	JSON File	1 KB
tslint.json	07-Feb-22 7:29 PM	JSON File	4 KB

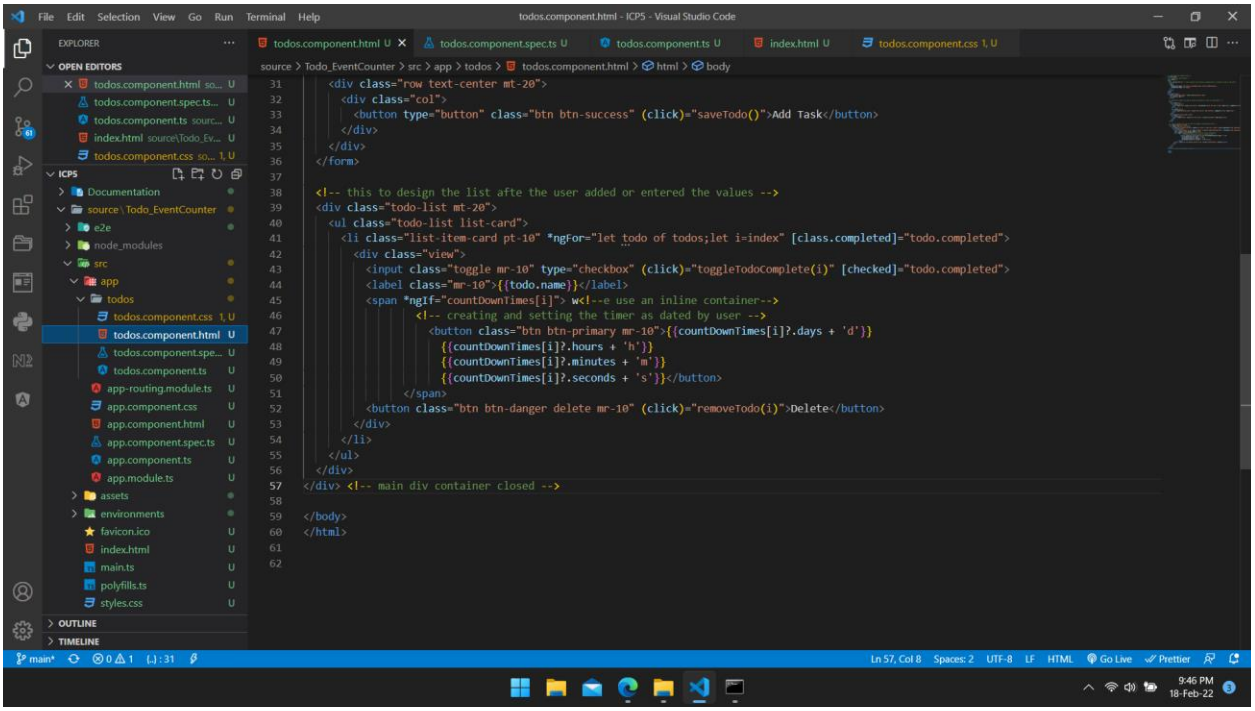
14 items

HTML code:

Below are the screenshots of html code developed to build our application.



```html
<!-- declaring the document type -->
<!DOCTYPE html>
<!-- Language for the document -->
<html lang="en">
<head>
  <meta charset="UTF-8"> <!--This specifies the character encoding which is optional to add in the code-->
  <style>
    body {
      background-image: url('https://wallpaperaccess.com/full/1092758.jpg');
      background-repeat: no-repeat;
    }
  </style>
</head>
<body>
<div class="todos">
  <header class="text-center" style="padding-bottom: 20px">
    <h1>Todo_List App</h1>
  </header>

  <!-- bootstrap classes are used to design and develop our todo list application. -->
  <form>
    <div class="row">
      <div class="col-6">
        <input type="text" class="form-control" placeholder="Enter the task to Add" name="todo" [(ngModel)]="todo">
      </div>
      <!-- </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>
```

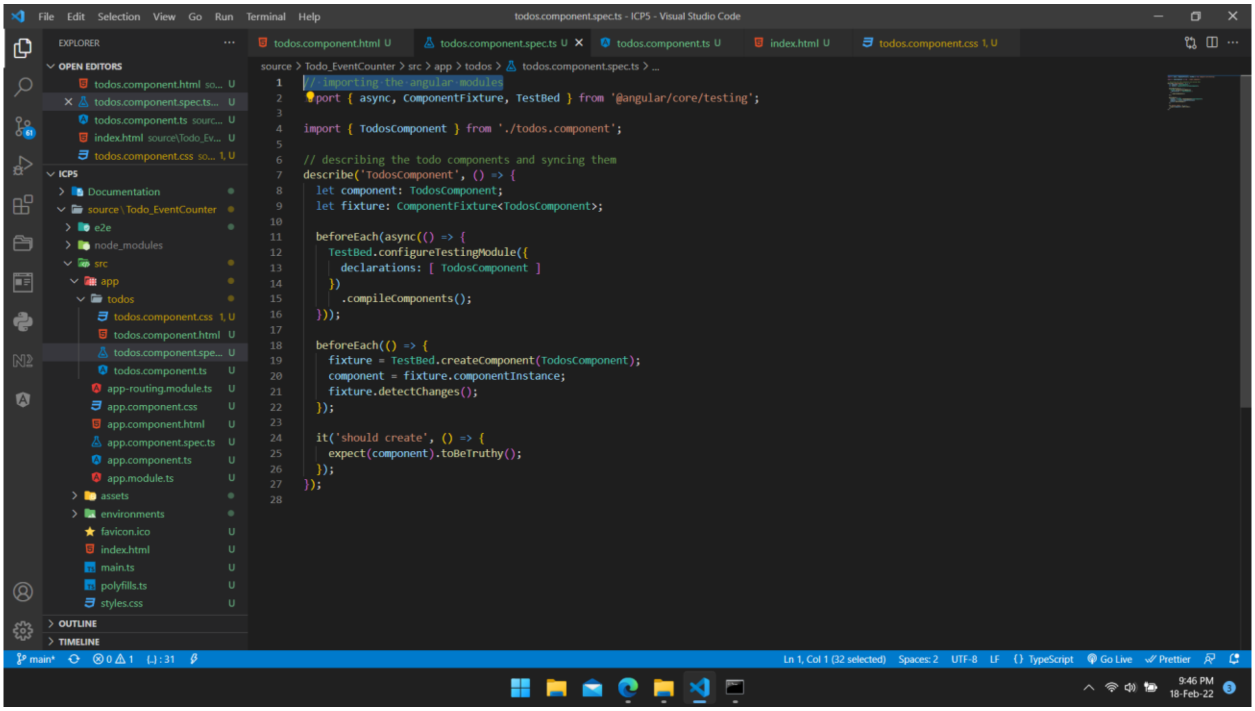


File Edit Selection View Go Run Terminal Help
todos.component.html - ICP5 - Visual Studio Code
EXPLORER
OPEN EDITORS
todos.component.html U
todos.component.spec.ts U
todos.component.ts U
index.html U
todos.component.css 1, U
source > Todo_EventCounter > src > app > todos > todos.component.html > html > body
main*
Ln 57, Col 8 Spaces: 2 UTF-8 LF HTML Go Live Prettier
9:46 PM
18-Feb-22

```html
    <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>

  <!-- this to design the list afte the user added or entered the values -->
  <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" [class.completed]="todo.completed">
        <div class="view">
          <input class="toggle mr-10" type="checkbox" (click)="toggleTodoComplete(i)" [checked]="todo.completed">
          <label class="mr-10">{{todo.name}}</label>
          <span *ngIf="countDownTimes[i]"> w<!--e use an inline container-->
                  <!-- creating and setting the timer as dated by user -->
                    <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>
    </ul>
  </div>
</div> <!-- main div container closed -->

</body>
</html>


```

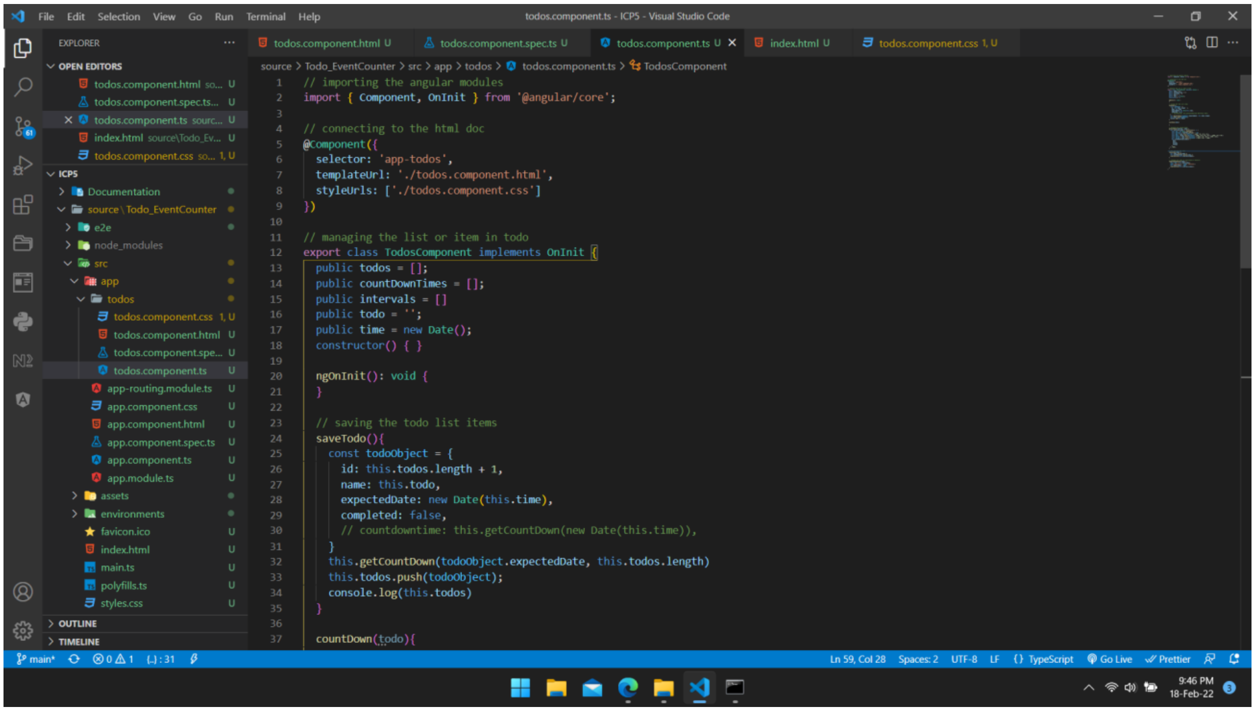
Below is the snap of angular spec,

Unit tests for your source files are contained in the spec files. For Angular apps, it is standard to have a .spec.ts file for each .ts file. When you use the ng test command, they are run using the Jasmine JavaScript test framework using the Karma test runner.

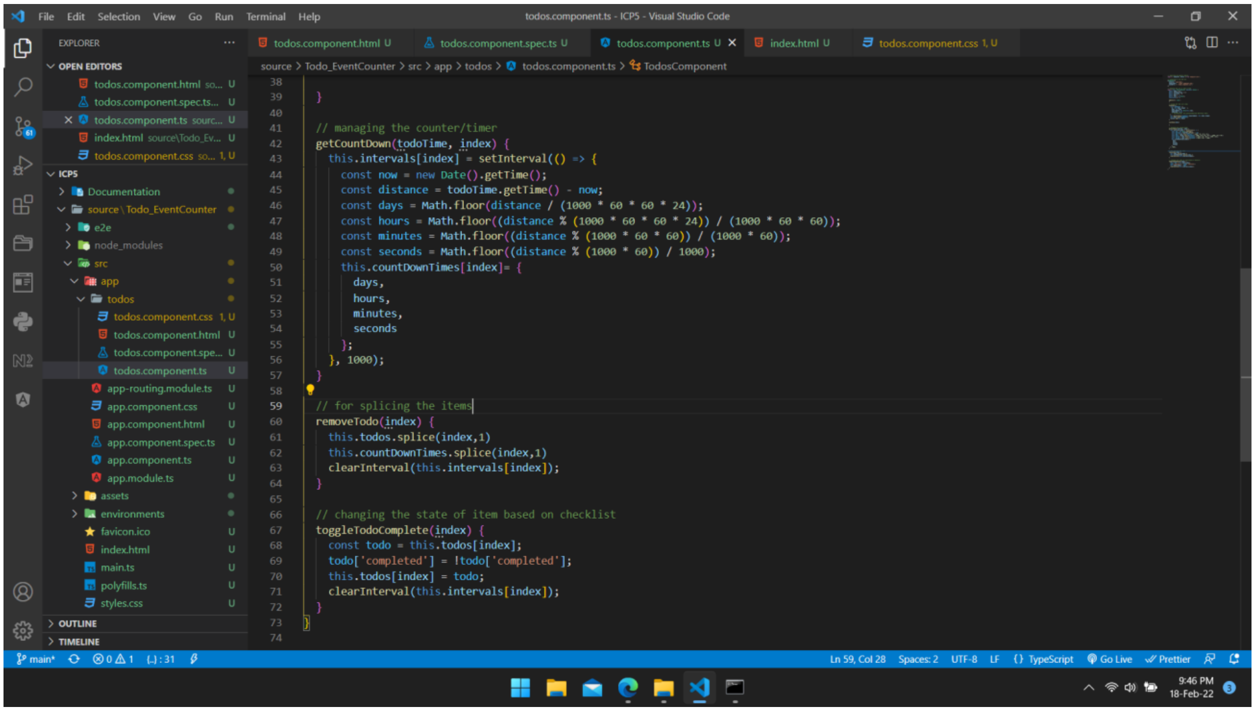

todos.component.spec.ts - ICP5 - Visual Studio Code
todos.component.html U
todos.component.spec.ts U
todos.component.ts U
index.html U
todos.component.css 1, U
source > Todo_EventCounter > src > app > todos > todos.component.spec.ts > ...
Ln 1, Col 1 (32 selected) Spaces: 2 UTF-8 LF TypeScript Go Live Prettier
9:46 PM
18-Feb-22

```
// importing the angular modules
port { async, ComponentFixture, TestBed } from '@angular/core/testing';

import { TodosComponent } from './todos.component';

// describing the todo components and syncing them
describe('TodosComponent', () => {
  let component: TodosComponent;
  let fixture: ComponentFixture<TodosComponent>;

  beforeEach(async(() => {
    TestBed.configureTestingModule({
      declarations: [ TodosComponent ]
    })
      .compileComponents();
  }));

  beforeEach(() => {
    fixture = TestBed.createComponent(TodosComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
  });
});
```

ts code:


todos.component.ts - ICP5 - Visual Studio Code
todos.component.html U
todos.component.spec.ts U
todos.component.ts U
index.html U
todos.component.css 1, U
source > Todo_EventCounter > src > app > todos > todos.component.ts > TodosComponent
Ln 59, Col 28 Spaces: 2 UTF-8 LF TypeScript Go Live Prettier
9:46 PM
18-Feb-22

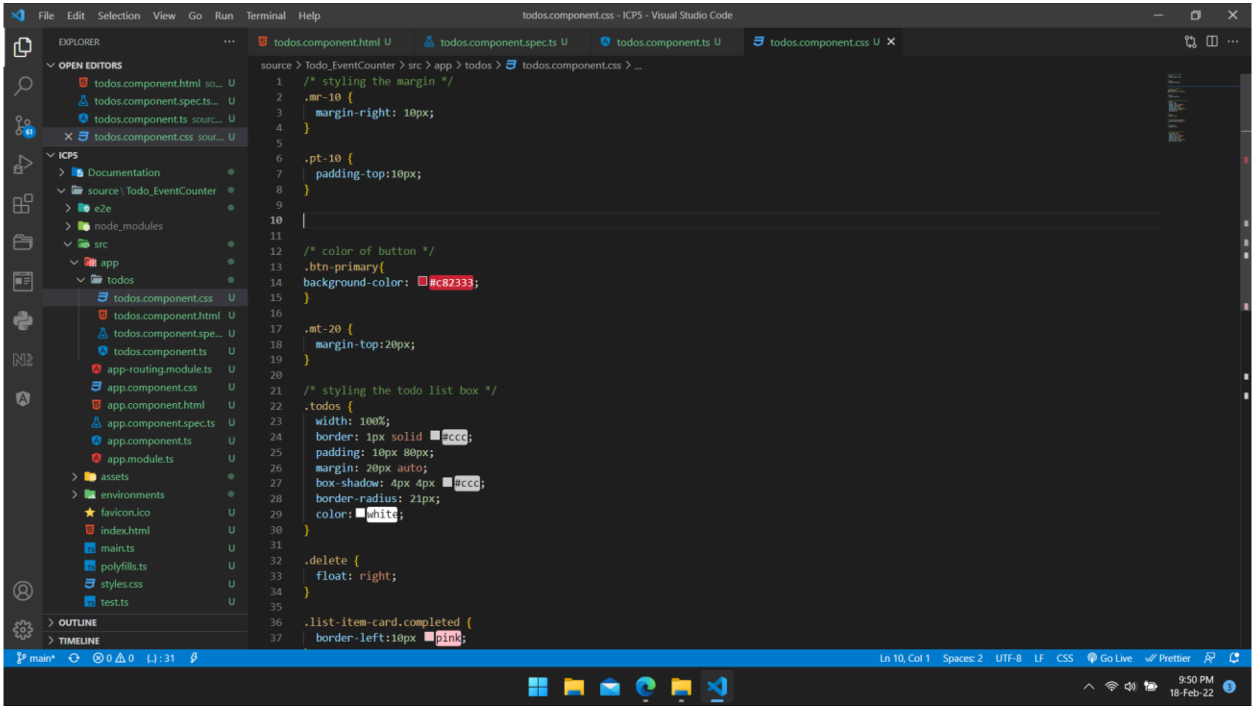
```
// importing the angular modules
import { Component, OnInit } from '@angular/core';

// connecting to the html doc
@Component({
  selector: 'app-todos',
  templateUrl: './todos.component.html',
  styleUrls: ['./todos.component.css']
})

// managing the list or item in todo
export class TodosComponent implements OnInit {
  public todos = [];
  public countDownTimes = [];
  public intervals = []
  public todo = '';
  public time = new Date();
  constructor() { }

  ngOnInit(): void {
  }

  // saving the todo list items
  saveTodo(){
    const todoObject = {
      id: this.todos.length + 1,
      name: this.todo,
      expectedDate: new Date(this.time),
      completed: false,
      // countdowntime: this.getCountDown(new Date(this.time)),
    }
    this.getCountDown(todoObject.expectedDate, this.todos.length)
    this.todos.push(todoObject);
    console.log(this.todos)
  }

  countDown(todo){
```



```
  }

  // managing the counter/timer
  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);
  }

  // for splicing the items
  removeTodo(index) {
    this.todos.splice(index,1)
    this.countDownTimes.splice(index,1)
    clearInterval(this.intervals[index]);
  }

  // changing the state of item based on checklist
  toggleTodoComplete(index) {
    const todo = this.todos[index];
    todo['completed'] = !todo['completed'];
    this.todos[index] = todo;
    clearInterval(this.intervals[index]);
  }
}

```

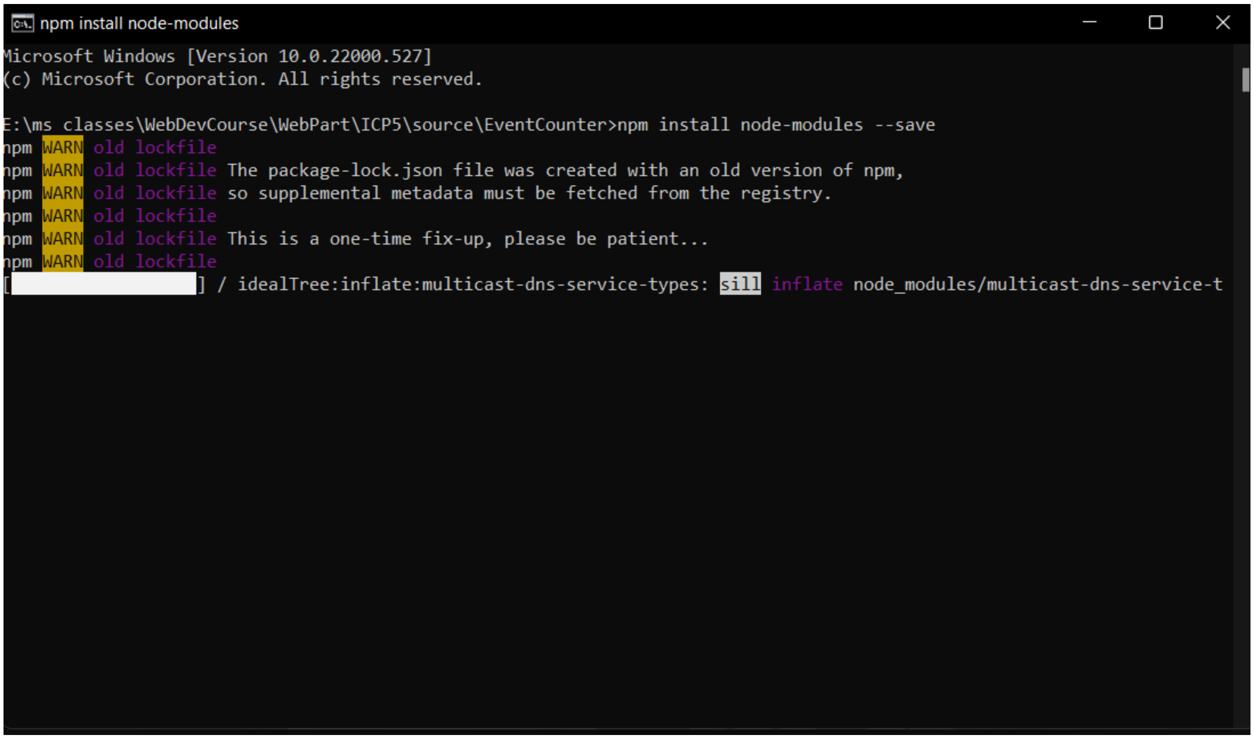
CSS code:

Below is the css code used to design our application.



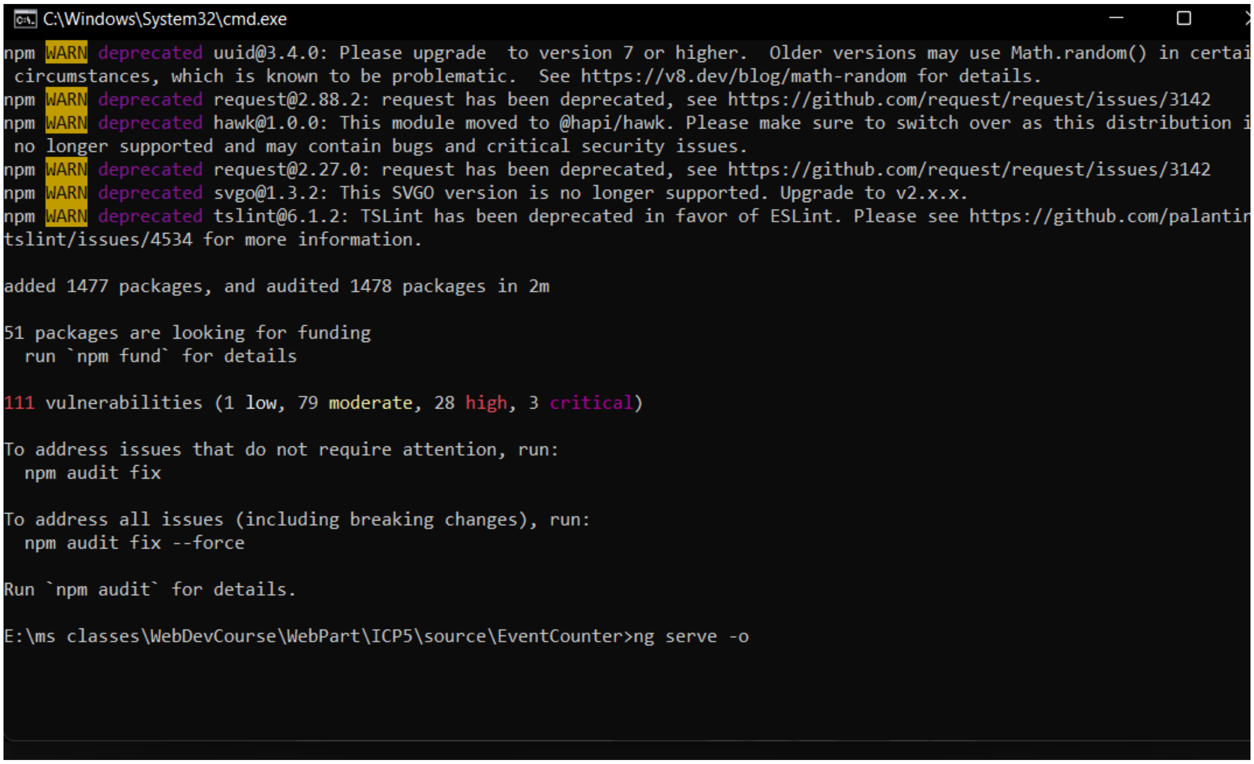
```
/* styling the margin */
.mr-10 {
  margin-right: 10px;
}

.pt-10 {
  padding-top:10px;
}



/* color of button */
.btn-primary{
background-color: #c82333;
}

.mt-20 {
  margin-top:20px;
}

/* styling the todo list box */
.todos {
  width: 100%;
  border: 1px solid #ccc;
  padding: 10px 80px;
  margin: 20px auto;
  box-shadow: 4px 4px #ccc;
  border-radius: 21px;
  color: white;
}

.delete {
  float: right;
}

.list-item-card.completed {
  border-left:10px pink;
```

Now, install the node modules using npm command “npm node-modules --save” as shown in the snap below.



```
Microsoft Windows [Version 10.0.22000.527]
(c) Microsoft Corporation. All rights reserved.

E:\ms classes\WebDevCourse\WebPart\ICP5\source\EventCounter>npm install node-modules --save
npm WARN old lockfile
npm WARN old lockfile The package-lock.json file was created with an old version of npm,
npm WARN old lockfile so supplemental metadata must be fetched from the registry.
npm WARN old lockfile
npm WARN old lockfile This is a one-time fix-up, please be patient...
npm WARN old lockfile
[                  ] / idealTree:inflate:multicast-dns-service-types: sill inflate node_modules/multicast-dns-service-t
```

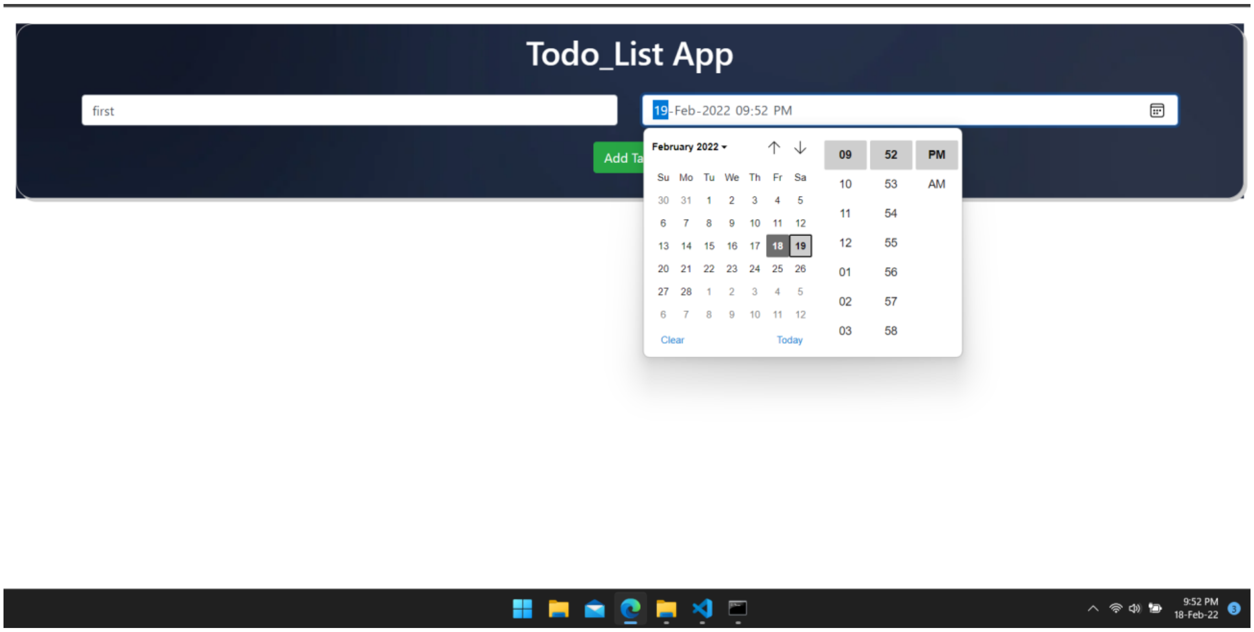
After getting the node files into your project destination, run the application as shown in the screenshot below.



```
npm WARN deprecated uuid@3.4.0: Please upgrade  to version 7 or higher.  Older versions may use Math.random() in certai
 circumstances, which is known to be problematic.  See https://v8.dev/blog/math-random for details.
npm WARN deprecated request@2.88.2: request has been deprecated, see https://github.com/request/request/issues/3142
npm WARN deprecated hawk@1.0.0: This module moved to @hapi/hawk. Please make sure to switch over as this distribution i
 no longer supported and may contain bugs and critical security issues.
npm WARN deprecated request@2.27.0: request has been deprecated, see https://github.com/request/request/issues/3142
npm WARN deprecated svgo@1.3.2: This SVGO version is no longer supported. Upgrade to v2.x.x.
npm WARN deprecated tslint@6.1.2: TSLint has been deprecated in favor of ESLint. Please see https://github.com/palantir
tslint/issues/4534 for more information.

added 1477 packages, and audited 1478 packages in 2m

51 packages are looking for funding
  run `npm fund` for details

111 vulnerabilities (1 low, 79 moderate, 28 high, 3 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.

E:\ms classes\WebDevCourse\WebPart\ICP5\source\EventCounter>ng serve -o
```

OUTPUT:

Enter the value and set the timer as needed, below is the reference to the same.

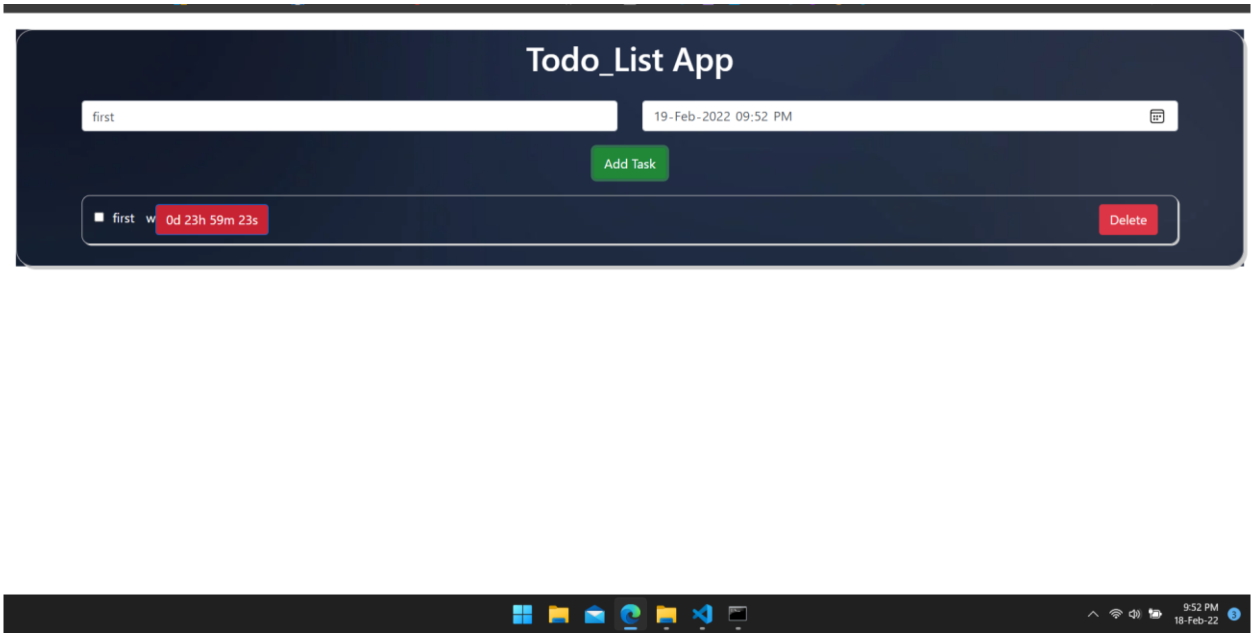


Todo_List App

first

19-Feb-2022 09:52 PM

Next, add the item to the list which is then displayed as below.



Todo_List App

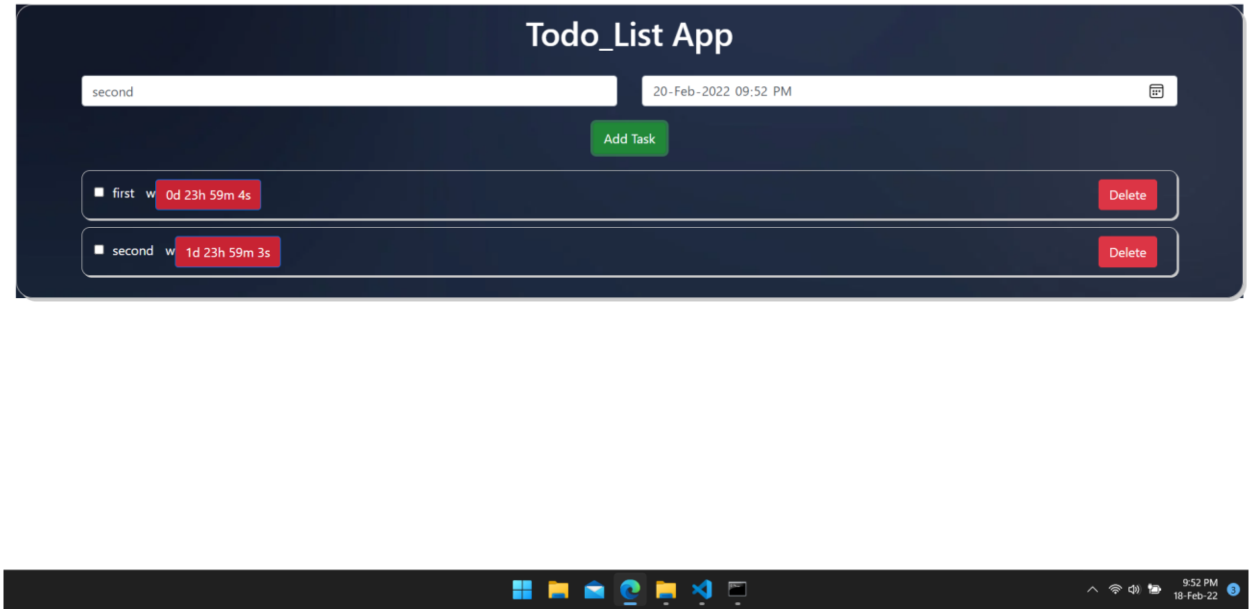
first

19-Feb-2022 09:52 PM

□ first w 0d 23h 59m 23s

Items added can also be deleted when done or when required.

Similarly, multiple list of activities can be added and managed through the application.



Todo_List App

second

20-Feb-2022 09:52 PM

□ first w 0d 23h 59m 4s

□ second w 1d 23h 59m 3s