# To-do list web application

**Design:**

Front-End Design:
User Interface (UI):
Clean and minimalistic design.
Input field for adding tasks.
List to display tasks with checkboxes and delete buttons.
Task count display.

**Test Cases:**

Back-End Tests:

1. Fetching Tasks:

Send a GET request to /api/tasks.
Verify the response contains an array of tasks.

2. Adding Task:

Send a POST request to /api/tasks with a new task.
Verify the task is added to the database.

3. Updating Task:

Send a PUT request to /api/tasks/:id with an updated task.
Verify the task is updated in the database.

4. Deleting Task:

Send a DELETE request to /api/tasks/:id.
Verify the task is removed from the database.

**Pseudo-Code:**

Back-End Pseudo-Code (Spring Boot):

```
// Entity Class
@Entity
public class Task {
```

```
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private String text;
    private boolean completed;

    // Getters and setters
}

// Repository Interface
public interface TaskRepository extends JpaRepository<Task, Long> {
}

// Service Class
@Service
public class TaskService {
    @Autowired
    private TaskRepository taskRepository;

    public List<Task> getAllTasks() {
        return taskRepository.findAll();
    }

    public Task addTask(Task task) {
        return taskRepository.save(task);
    }

    public Task updateTask(Long id, Task updatedTask) {
        // Implement logic to update task in the database
    }

    public void deleteTask(Long id) {
        taskRepository.deleteById(id);
    }
}

// Controller Class
@RestController
@RequestMapping("/api/tasks")
```

```
public class TaskController {
   @Autowired
   private TaskService taskService;

   @GetMapping
   public List<Task> getAllTasks() {
      return taskService.getAllTasks();
   }

   @PostMapping
   public Task addTask(@RequestBody Task task) {
      return taskService.addTask(task);
   }

   @PutMapping("/{id}")
   public Task updateTask(@PathVariable Long id, @RequestBody Task
updatedTask) {
      return taskService.updateTask(id, updatedTask);
   }

   @DeleteMapping("/{id}")
   public void deleteTask(@PathVariable Long id) {
      taskService.deleteTask(id);
   }
}
```

**Business Logic:**

Adding Task:

User adds a task through the UI.
Front-end sends a POST request to /api/tasks.
Spring Boot adds the task to the database.
Deleting Task:

User deletes a task through the UI.
Front-end sends a DELETE request to /api/tasks/:id.
Spring Boot removes the task from the database.
Completing Task:

User marks a task as completed through the UI.
Front-end sends a PUT request to /api/tasks/:id.
Spring Boot updates the task status in the database.
Fetching Tasks:

User opens the application.
Front-end sends a GET request to /api/tasks.
Spring Boot retrieves all tasks from the database.

**UI:**

Homepage:
Header: "To-Do List"
Input field for adding tasks.
List to display tasks with checkboxes and delete buttons.
Task count display.

**Wireframe:**