Assignment: Building a Simple To-Do List Application and Software Design

Objective:

In this assignment, you will design and implement a **To-Do List application** using Python. You will also write a report explaining your design choices and reflecting on any challenges you faced during development.

Instructions:

1. Design the Application

- Plan how your To-Do List application will work. Consider:
  - **Input/Output**: How will users interact with your application?
  - **Data Storage**: Will the tasks be stored in memory or saved in a file?
  - **Functions**: Break the program into small functions (e.g., adding a task, removing a task, viewing all tasks).

Hint: You can use a list of dictionaries to store tasks, where each task is represented as a dictionary with a name and a status (e.g., incomplete/complete).

Example structure of your application:

- `add_task()`: Adds a task to the list.
- `view_tasks()`: Displays all tasks.
- `complete_task()`: Marks a task as complete.
- `save_tasks()`: Saves the tasks to a file.
- `load_tasks()`: Loads the tasks from a file.

2. Implement the Application in Python

- Write the Python code for your To-Do List application based on your design.
- Your code should allow the user to:
  - Add new tasks.
  - View existing tasks.
  - Mark tasks as complete.
  - Save tasks to a file and load them back into the program.

Example code for adding and viewing tasks:

```
tasks = []

def add_task(name):
    task = {'name': name, 'status': 'incomplete'}
    tasks.append(task)
    print(f"Added task: {name}")
```

```
def view_tasks():
    if len(tasks) == 0:
        print("No tasks to show.")
    else:
        for index, task in enumerate(tasks, start=1):
            status = task['status']
            print(f"{index}. {task['name']} - {status}")
```

Add additional functions like marking tasks as complete and saving/loading tasks from a file.

3. Write a Report

- Write a short report (about 500 words) explaining:
    1. **Software Design**: How did you design the To-Do List application? (Explain the functions you created and how the program is structured.)
    2. **Challenges**: Describe any challenges you faced while implementing the application and how you overcame them.
    3. **Future Improvements**: Suggest features you could add to improve the application in the future (e.g., prioritizing tasks, adding deadlines).