**Task List Application Documentation**

**Overview**

This Task List Application is designed to help users manage a list of tasks. It allows users to create tasks with titles, descriptions, and statuses, mark tasks as complete, add priority tasks, list all tasks, remove tasks, and find tasks by title.

**Classes**

**Task**

Represents a single task in the Task List Application.

**Attributes :**

- title(str): Title of the task.

- description (str): Description of the task (optional).

- status (str): Status of the task, either "incomplete" or "complete".

**- Methods:**

- \_\_init\_\_(self, title, description="", status="incomplete"): Initializes a new task with the given title, description, and status.

- mark\_complete(self): Marks the task as complete.

- `\_\_str\_\_(self)`: Overrides the default string representation to display task details.

**TaskList**

Represents a collection of tasks in the Task List Application.

**Attributes:**

- tasks (list): List to store all tasks.

**Methods:**

- \_\_init\_\_(self): Initializes an empty task list.

- add\_task(self, title, description=""): Adds a new task to the list with the given title and description.

- remove\_task(self, title): Removes a task from the list based on its title.

- list\_tasks(self): Lists all tasks in the task list.

-find\_task(self, title): Finds and returns a task with the specified title.

**PriorityTask**

Represents a priority task, which is a type of task with an additional priority attribute.

**Attributes:**

- Inherits attributes from ‘Task’ class.

- priority (str): Priority level of the task, such as "low", "medium", or "high".

**Methods:**

- \_\_init\_\_(self, title, description="", priority="low"): Initializes a new priority task with the given title, description, and priority.

- \_\_str\_\_(self): Overrides the default string representation to display priority task details.

**OOP Concepts**

**1. Classes and Objects:**

- Classes are used to represent tasks (‘Task’ and ‘PriorityTask’) and task lists (‘TaskList’).

- Objects of these classes are created to perform operations on tasks.

**2. Methods:**

- Various methods are implemented in classes to perform specific tasks such as adding, removing, and listing tasks.

**3. Method Overloading:**

- Python does not support method overloading directly, but default arguments are used to simulate different method signatures, such as adding tasks with or without descriptions.

**4. Method Overriding:**

- The \_\_str\_\_ method is overridden in both ‘Task’ and ‘PriorityTask’ classes to provide custom string representations.

5. Inheritance:

- ‘PriorityTask’ inherits from the ‘Task’ class, gaining its attributes and methods while adding the ‘priority’ attribute.

6. Encapsulation:

- Attributes of classes are encapsulated within the classes, and access to them is controlled through methods.

7. Polymorphisms:

- Polymorphism is demonstrated through method overriding and treating ‘PriorityTask’ objects as ‘Task’ objects when added to the task list.

This documentation provides an overview of the Task List Application, its classes, methods, and how various OOP concepts are applied within the project.