```
import datetime
from typing import Dict, List, Optional

class WeeklyPlanner:
    def __init__(self):
        self.week_days = ["Monday", "Tuesday", "Wednesday",
"Thursday", "Friday", "Saturday", "Sunday"]
        self.schedule: Dict[str, List[Dict]] = {day: [] for day in
self.week_days}

    def add_task(self, day: str, task: str, time: str = "", priority: str =
"Medium"):
        """Add a task to a specific day"""
        if day not in self.week_days:
            return f"Invalid day. Please choose from: {',
'.join(self.week_days)}"

        task_entry = {
            "task": task,
            "time": time,
            "priority": priority,
            "completed": False
        }
        self.schedule[day].append(task_entry)
        return f"Added '{task}' to {day} at {time if time else 'no specific
time'}"

    def view_day(self, day: str):
        """View tasks for a specific day"""
        if day not in self.week_days:
            return f"Invalid day. Please choose from: {',
'.join(self.week_days)}"
```

```
    tasks = self.schedule[day]
    if not tasks:
        return f"No tasks scheduled for {day}"

    result = f"Schedule for {day}:\n"
    result += "-" * 30 + "\n"

    for i, task in enumerate(tasks, 1):
        status = "" if task["completed"] else "
        result += f"{i}. {status} {task['task']}\n"
        if task['time']:
            result += f"   Time: {task['time']}\n"
        result += f"   Priority: {task['priority']}\n"

    return result

def view_week(self):
    """View the entire week's schedule"""
    result = "WEEKLY SCHEDULE OVERVIEW\n"
    result += "=" * 40 + "\n"

    for day in self.week_days:
        tasks = self.schedule[day]
        completed = sum(1 for t in tasks if t["completed"])
        total = len(tasks)

        result += f"\n{day}:\n"
        result += f"  Tasks: {completed}/{total} completed\n"
        for task in tasks:
            status = "✓" if task["completed"] else "○"
            result += f"    {status} {task['task'][:30]}{'...' if
```

```
len(task['task']) > 30 else "}\n"

        return result

    def mark_complete(self, day: str, task_index: int):
        """Mark a task as completed"""
        if day not in self.week_days:
            return f"Invalid day."

        tasks = self.schedule[day]
        if task_index < 1 or task_index > len(tasks):
            return f"Invalid task number. Please choose between 1 and
{len(tasks)}"

        tasks[task_index - 1]["completed"] = True
        return f"Marked '{tasks[task_index - 1]['task']}' as completed!"

    def get_today_schedule(self):
        """Get schedule for today"""
        today = datetime.datetime.now().strftime("%A")
        return self.view_day(today)

    def clear_day(self, day: str):
        """Clear all tasks for a specific day"""
        if day not in self.week_days:
            return f"Invalid day."

        self.schedule[day] = []
        return f"Cleared all tasks for {day}"

    def get_upcoming_tasks(self):
        """Get all upcoming (incomplete) tasks"""
```

```
        result = "UPCOMING TASKS:\n"
        result += "-" * 30 + "\n"

        today = datetime.datetime.now().strftime("%A")
        today_index = self.week_days.index(today)

        for i in range(7):
            day_index = (today_index + i) % 7
            day = self.week_days[day_index]
            tasks = [t for t in self.schedule[day] if not t["completed"]]

            if tasks:
                day_label = "TODAY" if i == 0 else f"In {i} day{'s' if i > 1 else
"}" if i > 0 else day
                result += f"\n{day_label} ({day}):\n"
                for task in tasks:
                    result += f"  • {task['task']}\n"
                    if task['time']:
                        result += f"    [Time: {task['time']}]\n"

        return result

# Google Assistant Style Interface
def assistant_interface():
    planner = WeeklyPlanner()

    print Google Assistant Weekly Planner Activated!")
    print("=" * 50)
    print("Commands:")
    print("  add [day] [task] [time(optional)] [priority(optional)]")
    print("  view [day] or 'view week'")
    print("  complete [day] [task number]")
```

```
print("  today")
print("  upcoming")
print("  clear [day]")
print("  help")
print("  exit")
print("=" * 50)

while True:
    try:
        user_input = input("\n How can I help you? > ").strip().lower()

        if user_input == "exit":
            print("Goodbye! Have a productive week! ")
            break

        elif user_input == "help":
            print("Available commands:")
            print("  add Monday 'Meeting with team' '10:00 AM' High")
            print("  view Tuesday")
            print("  view week")
            print("  complete Monday 1")
            print("  today")
            print("  upcoming")
            print("  clear Monday")
            print("  exit")

        elif user_input.startswith("add "):
            parts = user_input[4:].split("'")
            # Extract day and task details
            if len(parts) >= 3:
                day = parts[0].strip().title()
                task = parts[1].strip()
```

```
            time = parts[2].strip() if len(parts) > 2 else ""
            priority = parts[3].strip().title() if len(parts) > 3 else
"Medium"
            print(planner.add_task(day, task, time, priority))

        elif user_input.startswith("view "):
            day = user_input[5:].strip().title()
            if day == "Week":
                print(planner.view_week())
            else:
                print(planner.view_day(day))

        elif user_input.startswith("complete "):
            parts = user_input[9:].split()
            if len(parts) >= 2:
                day = parts[0].title()
                try:
                    task_num = int(parts[1])
                    print(planner.mark_complete(day, task_num))
                except ValueError:
                    print("Please provide a valid task number")

        elif user_input == "today":
            print(planner.get_today_schedule())

        elif user_input == "upcoming":
            print(planner.get_upcoming_tasks())

        elif user_input.startswith("clear "):
            day = user_input[6:].strip().title()
            print(planner.clear_day(day))
```

```
            else:
                print("I didn't understand that. Type 'help' to see available
commands.")

        except KeyboardInterrupt:
            print("\n\nGoodbye! ")
            break
        except Exception as e:
            print(f"Oops! Something went wrong: {e}")

# Quick setup function for sample schedule
def setup_sample_schedule():
    planner = WeeklyPlanner()

    # Add sample tasks
    planner.add_task("Monday", "family program ", "4:00 AM", "High")
    planner.add_task("Monday", "Go to work shop", "6:00 AM",
"Medium")
    planner.add_task("Tuesday", "Gym session", "7:00 AM", "Low")
    planner.add_task("Wednesday", "Church team work", "9:00 AM",
"High")
planner.add_task("Thursday ", " work day high ", "2:00 AM", "High")
    planner.add_task("Friday", "Weekly report test ", "4:00 PM", "High")
    planner.add_task("Saturday", " shopping", "10:00 AM", "Medium")
    planner.add_task("Sunday", "assignment summit", "8:00 AM",
"Medium")

    return planner

# If you want a simpler text-based version
def simple_weekly_planner():
    planner = setup_sample_schedule()
```

```
    print("\n" + "="*50)
    print("WEEKLY PLANNER DEMO")
    print("="*50)

    # Show entire week
    print(planner.view_week())

    # Show today's schedule
    print("\n" + "="*50)
    print("TODAY'S SCHEDULE:")
    print("="*50)
    print(planner.get_today_schedule())

    # Show upcoming tasks
    print("\n" + "="*50)
    print("UPCOMING TASKS:")
    print("="*50)
    print(planner.get_upcoming_tasks())

# Run the program
if __name__ == "__main__":
    print("Select mode:")
    print("1. Google Assistant Style Interface")
    print("2. Simple Weekly Planner Demo")

    choice = input("Enter choice (1 or 2): ").strip()

    if choice == "1":
        assistant_interface()
    elif choice == "2":
        simple_weekly_planner()
```

```
    else:
        print("Starting assistant interface by default...")
        assistant_interface()

import speech_recognition as sr
import pyttsx3
import time

class VoicePlanner(WeeklyPlanner):
    def __init__(self):
        super().__init__()
        self.engine = pyttsx3.init()
        self.recognizer = sr.Recognizer()

    def speak(self, text):
        """Convert text to speech"""
        self.engine.say(text)
        self.engine.runAndWait()

    def listen(self):
        """Listen for voice commands"""
        with sr.Microphone() as source:
            print(" Listening...")
            self.recognizer.adjust_for_ambient_noise(source)
            audio = self.recognizer.listen(source)

            try:
                command = self.recognizer.recognize_google(audio)
                print(f"You said: {command}")
                return command.lower()
            except sr.UnknownValueError:
                return "Sorry, I didn't catch that."
```

```
            except sr.RequestError:
                return "Sorry, speech service is down."


Select mode: 1

Google Assistant Weekly Planner Activated!
==================================================
Commands:
  add [day] [task] [time(optional)] [priority(optional)]
  view [day] or 'view week'
  complete [day] [task number]
  today
  upcoming
  clear [day]
  help
  exit
==================================================
1.
How can I help you? > add Monday 'Team Meeting' '9:00 AM' High
Added 'Team Meeting' to Monday at 9:00 AM

How can I help you? > view Monday
 Schedule for Monday:
----------------------------
1.  Team Meeting
   Time: 9:00 AM
   Priority: High
```