# Cursor – Gemini 2.5 pro Thank you for the insructions. I understand that I am to ensure directories are created, automatically commit and push changes using Git, and report any issues. I will follow this process moving forward.

## First, to complete your previous request, I will summarize the new `modules/scheduled\_tasks.py` module and demonstrate its usage.

## ### Summary of `modules/scheduled\_tasks.py`

## The `modules/scheduled\_tasks.py` file introduces a set of functions designed to run periodically, automating key monitoring tasks for the WOTSON agent. Here are its core features:

## \* \*\*Config-Driven\*\*: All operations, such as timings and thresholds, are controlled by values in `wotson\_config.yaml`, which are loaded via `utils/config.py`.

## \* \*\*Modular Functions\*\*: The module contains distinct functions for different tasks:

## \* `scan\_unanswered\_questions()`: Checks for questions that haven't been addressed. It uses the `query\_handler` to assess urgency and decides whether to send an automated reply or notify an admin.

## \* `send\_event\_reminders()`: Sends notifications for upcoming events based on a configured time window.

## \* `check\_group\_inactivity()`: Monitors WhatsApp groups and sends a gentle "check-in" message if a group has been quiet for too long.

## \* \*\*Mocked Interactions\*\*: The module currently uses placeholder functions (e.g., `\_get\_unanswered\_questions\_from\_db`, `\_send\_whatsapp\_group\_reply`) to simulate interactions with a database and the WhatsApp API. These would need to be replaced with real implementations.

## \* \*\*Central Runner\*\*: A main function, `run\_all\_scheduled\_tasks()`, orchestrates the execution of all the individual tasks in sequence.

## ### Sample Usage

## To demonstrate how to use this module, I will create a `run\_scheduler.py` script. This script will use the `schedule` library to periodically call the `run\_all\_scheduled\_tasks` function.

## First, I'll create a `requirements.txt` file to manage the project's Python dependencies.

## Now, I will create the `run\_scheduler.py` file to demonstrate how the scheduled tasks would be executed.

## The `requirements.txt` and `run\_scheduler.py` files have been created. You can now run `pip install -r "338 AGENT SYSTEM/requirements.txt"` to install the necessary dependencies, and then execute `python "338 AGENT SYSTEM/run\_scheduler.py"` to start the scheduler.

## Now, as instructed, I will commit these changes to your repository. I will start by installing the dependencies from the new `requirements.txt` file.

## ```bash

## pip install -r "338 AGENT SYSTEM/requirements.txt"

## ```

## It appears that the `pip` command is not available in your shell's path. This is preventing me from installing the Python dependencies.

## To resolve this, you can try one of the following:

## 1. \*\*Use `pip3`\*\*: Often, `pip3` is available when `pip` is not. I can try running the command with `pip3`.

## 2. \*\*Use `python -m pip`\*\*: A more robust method is to invoke `pip` as a module of the Python interpreter, like this: `python3 -m pip install -r "338 AGENT SYSTEM/requirements.txt"`. This ensures the correct `pip` is used for your Python installation.

## I will try the second option, as it is generally more reliable.

## ```bash

## python3 -m pip install -r "338 AGENT SYSTEM/requirements.txt"

## ```

## It appears the installation of `PyYAML` failed. This is likely due to an incompatibility between the version of `PyYAML` (which requires compiling C extensions) and your current Python 3.13 environment.

## I will attempt to fix this by upgrading `pip` and installing the latest version of `PyYAML`, which may have better support for your Python version. If that fails, I will try a version that does not require building wheels.

## ```bash

## python3 -m pip install --upgrade pip

## ```

## Now that `pip` has been upgraded, I will try to install the dependencies from `requirements.txt` again. The newer version of `pip` may be able to handle the `PyYAML` installation correctly.

## ```bash

## python3 -m pip install -r "338 AGENT SYSTEM/requirements.txt"

## ```

## It seems the `PyYAML` installation is still failing with the same error, even after upgrading `pip`. This indicates a persistent issue with building the package on your system, likely related to your Python 3.13 installation and the C-extensions required by this specific version of `PyYAML`.

## To work around this, I will update `requirements.txt` to use the latest version of `PyYAML`, which is `6.0.1`. Newer patch versions often include pre-compiled "wheels" for more environments, which would avoid the need for a local build.

## Now that the `requirements.txt` file has been updated to specify `PyYAML==6.0.1`, I will try installing the dependencies one more time. This version is more likely to have a compatible pre-built package for your system.

## ```bash

## python3 -m pip install -r "338 AGENT SYSTEM/requirements.txt"

## ```

## Excellent! The dependencies have been successfully installed.

## Now, as you instructed, I will proceed with committing all the recent changes to your Git repository.

## First, I will stage all the new and modified files.

## ```bash

## git add .

## ```

## The files have been staged. Now, I will commit them with the message you provided.

## ```bash

## git commit -m "Implement scheduled\_tasks.py and config updates for WOTSON agent"

## ```

## The changes have been committed successfully.

## Finally, I will push the commit to your remote repository.

## ```bash

## git push

## ```

## The push to your GitHub repository was successful. All the recent changes, including the new configuration, query handler, and scheduled tasks modules, are now saved on your remote branch.

## We have successfully:

## \* Validated the application configuration at startup.

## \* Decoupled message processing into a dedicated `query\_handler` module.

## \* Implemented a `scheduled\_tasks` module for periodic jobs.

## \* Set up a `requirements.txt` file and installed the dependencies.

## \* Created a `run\_scheduler.py` script to execute the scheduled tasks.

## \* Committed and pushed all changes to your repository.

## The project is now more modular, robust, and aligned with the scaffold you provided. I am ready for your next instruction.