Testing Scripts

Before you start check all directories under Claude/Tasks. Each directory is one task done by you. Completed tasks have report markdown file: TASK_REPORT.md. Go through each report to learn what you have done so far with the project.

After you are done with this, extend the project with additional bash script `test.sh` which will test every single installed AI model that we support by sending to it the request. Then,
the result of its work will be asserted. If assertion of certain model work fails exit the test script with proper error and the details.

Based on obtained error information apply the fix on the project codebase. After fix is done verify it and re-run the test. Repeat the routine until the test finishes without failure and all discovered problem have been solved.

Follow the `install.sh` and `install_ollama.sh` scripts to get to the information about supported models. Since the scripts dynamically determine which models can run on local host machine, you will test only the models that are possible to runn on the current machine.

Pay attention on the generative audio models that we support and exposed scripts to "play" with them. Same principle shall be applied for every other model so model can accept the request, do its job and return us result of work in some form.

Note: Running the install with proper argument passed to it to install models for particular category can take some time (in minutes or stronger) since missing models will be downloaded from the remote backend(s).

Note: All produced materials bu the test - especially generated materials by generative models shall be located under the `Tests` directory under the following directory structure: `Tests/{YYYY-MM-DD}/${MODEL}/Generated`. Discovered issues with all the details will be documented in the separate markdown files under: `Tests/{YYYY-MM-DD}/${MODEL}/Issues` so you can read them sometimes again. Final report for the whole run for particula model will be located under: `Tests/{YYYY-MM-DD}/${MODEL}/Report.md`. Report must contain all relevant indirmation about the testing and fixes applied for partiular model.

Note: During the testing pay attention about issues that could come out as the result of external factor - for example our internet does not work. If you determine such obstacle abort the testing and write proper details about the proble which has prevented us to finish the testing and fixing session.

Note: Add flag into the `test.sh` script so it could just run and fail on error or, if error is detected it will do exactly the same repairing loop like you are going to do. In your case since you are already running the whole "show" you will need the test script to execute with this flag set to off - which will be the default (no Claude AI will fix discovered issues in a loop). Claude (you) shall run fixing using the storngest available model, with all interactive questions auto-approved.

Once you have extended the project verify that no bugs in the scripts are intorduced and that everything works as expected. For this task write its TASK_REPORT.md so you can continue next time with next / upcoming task.