```
import os

import logging

import warnings

import concurrent.futures

from dotenv import load_dotenv

from loguru import logger

from swarms.utils.disable_logging import disable_logging




def bootup():

    """Initialize swarms environment and configuration


    Handles environment setup, logging configuration, telemetry,

    and workspace initialization.

    """

    try:

        # Load environment variables

        load_dotenv()


        # Configure logging

        if (

            os.getenv("SWARMS_VERBOSE_GLOBAL", "False").lower()

            == "false"

        ):

            logger.disable("")

            logging.disable(logging.CRITICAL)
```

```
# Silent wandb
os.environ["WANDB_SILENT"] = "true"

# Configure workspace
workspace_dir = os.path.join(os.getcwd(), "agent_workspace")
os.makedirs(workspace_dir, exist_ok=True)
os.environ["WORKSPACE_DIR"] = workspace_dir

# Suppress warnings
warnings.filterwarnings("ignore", category=DeprecationWarning)

# Run telemetry functions concurrently
try:
    with concurrent.futures.ThreadPoolExecutor(
        max_workers=2
    ) as executor:
        from swarms.telemetry.sentry_active import (
            activate_sentry,
        )

        future_disable_logging = executor.submit(
            disable_logging
        )
        future_sentry = executor.submit(activate_sentry)
```

```
                # Wait for completion and check for exceptions
                future_disable_logging.result()
                future_sentry.result()
        except Exception as e:
            logger.error(f"Error running telemetry functions: {e}")

    except Exception as e:
        logger.error(f"Error during bootup: {str(e)}")
        raise


# Run bootup
bootup()
```