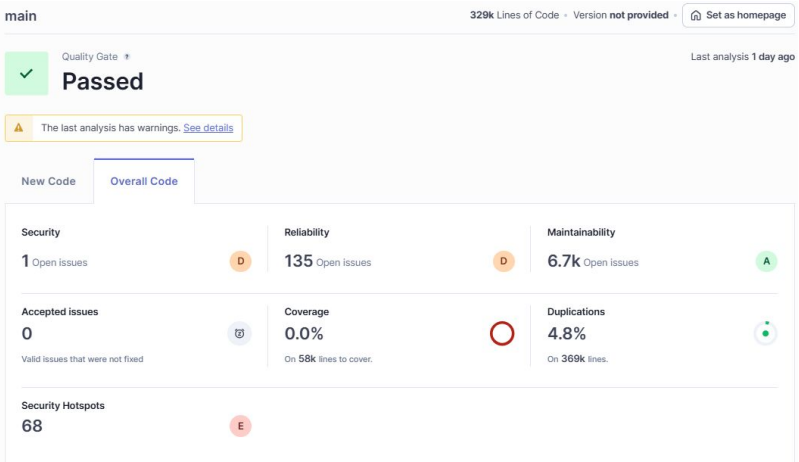
Group 3 Assignment 1 Presentation

Shiqi Wu, Wenkang Gong, Zihan Kuang, Yanqiu Mei, Ruixuan Li

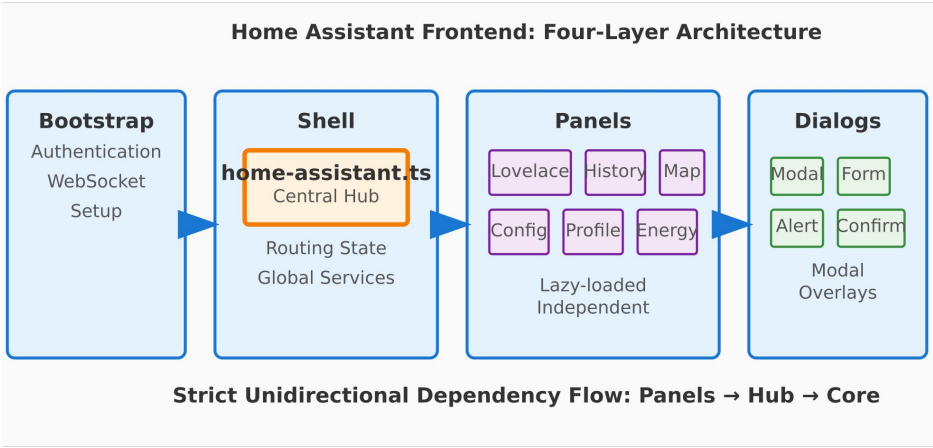
HA Frontend: Scalable Hub-Based Architecture

Multi-Technique Analysis



- **Quality:** Only 0.4% technical debt ratio
- **Hub-based design:** home-assistant.ts as central architectural coordinator

Scalable Architecture Pattern



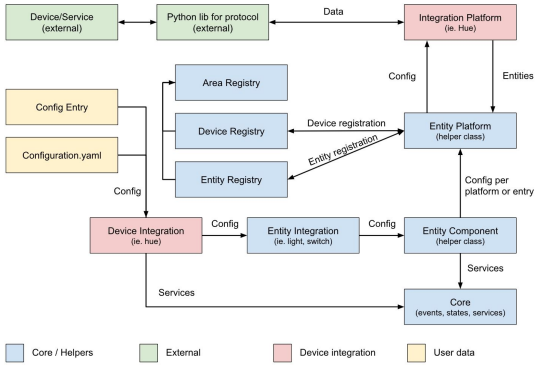
- **Four-layer design:** Bootstrap → Shell → Panels → Dialogs
- **Decoupled Panels:** src/panels/ with independent, lazy-loaded components

HA Core: A Disciplined "Micro-OS" Architecture at Scale

High-Level View from Analysis Tools

- **Excellent Code Health:** 'A' rating for maintainability with high modularity.
- **"Micro-OS" Architecture:** A central "Kernel" with a decoupled plugin ecosystem.

Official Entity Architecture



- **Event-Driven Core:** Built on an Event Bus for loose coupling.
- **The "Entity" Abstraction:** A universal model that standardizes all devices
- **Design Philosophy:** The Entity model is the practical design behind the "Micro-OS" structure

Hue Integration

Hue System Structure

- State-based architecture, where all devices has their states and they are being managed via *Hue Bridge*
- Communication through Hue APIs (aiohue), exposing lights, sensors, remotes, scenes, etc.

“The hue system is built around the idea of everything in your system having a unique URL served by the bridge. Interacting with these URLs lets you modify them or find out their current state.”

— Hue core concepts

https://developers.meethue.com/develop/hue-api-v2/core-concepts/

Static Code Analysis

- High overall test coverage (90%) means safer refactoring & feature additions
- A few modules do not have high coverage (services.py 65%; device_trigger.py 68%): provides opportunity to increase the amount of unit tests for higher robustness
- Hue integration supports 2 versions of Hue: Legacy V1 and current V2. They have similar structure and metrics.

Group Reflections

- Reflections on the code comprehension
  - Educational and necessary
  - Find potential defects or improvements + learn good practices in design and development
- Tool Exploration for Architecture Visualization

Tool Category	Tool Name	Decision	Rationale for Decision
Call Graph Generators	**PyReveng3 & pycg**	Abandon	Complex setup and python compatibility issues.
Dependency Visualizer	**pydeps**	Chosen	Practical and direct for visualizing Python imports.

Also: Switch tools and discuss your findings before it’s too late!

Thank you for listening!

Questions?