**Лабораторная работа № 3**

**Выбор технологий, используемых для реализации системы “Умный дом”.**

**Язык программирования Python:**

* Простота и читаемость кода, а это быстрая разработка и легкая поддержка.
* Большое количество библиотек для работы.
* Поддержка асинхронного программирования для обработки событий от множества устройств.
* Широкая поддержка сообщества и обширная документация.

**Прикладная платформа VS Code:**

* Легковесная, но мощная среда разработки.
* Отличная поддержка Python через расширения.
* Интеграция с Git и GitHub, а также с инструментами для работы с базами данных.
* Поддержка PlantUML через расширение для построения диаграмм.

**Репозиторий Git + GitHub:**

* Контроль версий для кода и конфигураций.
* Возможность командной работы и отслеживания изменений.
* GitHub предоставляет платформу для CI/CD, что упрощает развертывание и тестирование.

**Диаграммы PlantUML:**

* Текстовое описание диаграмм, которое можно хранить в Git.
* Легкое создание и изменение диаграмм без ручной расстановки элементов.
* Поддержка различных типов диаграмм (архитектура, последовательность, состояния и др.), что полезно для документирования системы.

**База данных PostgreSQL:**

* Надежная, объектно-реляционная СУБД.
* Поддержка JSONB для хранения данных с изменяемой структурой (полезно для хранения показаний датчиков и состояний устройств).
* Возможность использования временных рядов (timescale) для эффективного хранения данных с временными метками.
* Высокая производительность и масштабируемость.

**Декомпозиция разрабатываемой системы**

**Модули:**

1. **Ядро системы (Core)**

* Отвечает за запуск, конфигурацию, управление жизненным циклом и координацию всех модулей.
* Взаимодействие с базой данных для хранения состояний и истории.
* Обработка событий и сервисов.

1. **Модуль интеграции устройств (Device Integration)**

* Поддержка различных протоколов (Zigbee, Z-Wave, MQTT, HTTP)
* Перевод команд и состояний между устройствами и единым форматом внутри системы.

1. **Модуль автоматизации (Automation)**

* Движок для выполнения правил "если-то" (if-then-else).
* Поддержка триггеров, условий и действий.

1. **Модуль сцен (Scenes)**

* Позволяет сохранять и восстанавливать состояния устройств (например, сцена "Кино": свет гаснет).

1. **Модуль скриптов (Scripts)**

* Последовательности действий, которые могут быть вызваны автоматизациями или вручную.

1. **Модуль истории и логирования (History and Logging)**

* Сохранение состояний и событий в базу данных (PostgreSQL).
* Предоставление API для извлечения исторических данных.

1. **Модуль пользовательского интерфейса (Frontend)**

* Веб-интерфейс для управления и просмотра состояния системы.
* Панели управления, дашборды.

1. **Модуль API (API)**

* REST API для взаимодействия с внешними системами.
* WebSocket API для реального времени.

1. **Модуль голосового управления (Voice Assistant)**

* Интеграция с голосовыми помощниками (Алиса, Google Assistant, Siri).

1. **Модуль мобильного приложения (Mobile App)**

* Отправка push-уведомлений, удаленное управление.

1. **Модуль безопасности (Security)**

* Управление доступом, аутентификация, авторизация.
* Шифрование данных.

1. **Модуль резервного копирования и восстановления (Backup and Restore)**

* Автоматическое создание резервных копий конфигурации и данных.

1. **Модуль анализа и машинного обучения (Analytics and ML)**

* Анализ данных для выявления паттернов и оптимизации работы.
* Прогнозирование (например, энергопотребления).

1. **Модуль уведомлений (Notifications)**

* Отправка уведомлений через различные каналы (email, SMS, push).

1. **Модуль системного мониторинга (System Monitoring)**

* Мониторинг здоровья системы (доступность устройств, загрузка CPU, память).

1. **Модуль управления энергопотреблением (Energy Management)**

* Мониторинг и оптимизация энергопотребления.

1. **Модуль геолокации (Geolocation)**

* Отслеживание местоположения пользователей для автоматизаций (например, включение отопления при приближении).

1. **Модуль работы с расписаниями (Scheduler)**

* Планирование действий по времени (расписания, таймеры).

1. **Модуль интеграции с внешними сервисами (External Services)**

* Интеграция с погодными сервисами, календарями.

1. **Модуль тестирования и отладки (Testing and Debugging)**

* Тестирование автоматизаций, симуляция событий, логирование для отладки.

**Набор задач необходимых для реализации ПО.**

**Сформулируем список задач с приоритетами (где 1 - самый важный, базовый функционал):**

1. ﻿﻿﻿Настройка Git-репозитория (обязательно, чтобы фиксировать прогресс). - Приоритет 1

2. ﻿﻿﻿Создание UML-моделей (классов, вариантов использования). - Приоритет 1

3. ﻿﻿﻿Реализация классов-заглушек для User, SmartHome и Device. - Приоритет 1

4. ﻿﻿﻿Реализация базовых устройств (Light, AC, Camera, DoorLock). - Приоритет 2

5. ﻿﻿﻿Добавление датчиков (Temperature, Motion, Smoke, WaterLeak). - Приоритет 2

6. ﻿﻿﻿Реализация SecuritySystem (сигнализация). - Приоритет 2

7. ﻿﻿﻿Реализация EnergyManager (отключение ненужных устройств). - Приоритет 3

8. ﻿﻿﻿Реализация ClimateControl (поддержание температуры/режимов). - Приоритет 3

9. ﻿﻿﻿Создание Notification и Log + подключение к NotificationDB/LogDB. - Приоритет 3

10. ﻿﻿﻿﻿Добавление контроллера связи (Controller). - Приоритет 4

11. ﻿﻿﻿﻿Тестовые сценарии (например, включение света, сработка датчика движения). - Приоритет 4