ВВЕДЕНИЕ

Данная дипломная работа посвящена проектированию web-интерфейса для удаленного доступа к программно-аппаратной платформе домашней автоматизации.

Задача данного программного проекта заключена в реализации контроля всех подключённых устройств с единого, дружественного пользователю web-интерфейса в реальном времени. Поддержке общения между устройствами и решение ими определенных повседневных задач без участия человека. Предоставление пользователю возможности гибкой настройки системы под свои потребности. Подразумевается, что пользователь выбирает вид управляемых устройств среди поддерживаемых системой в соответствии с задачами, исполняемые ими.

Проект предоставляет возможности по управлению освещением, системой микроклимата, управлению электроприводами, внедрению системы безопасности.

Основные возможности:

* управление различными типами устройств
* веб-доступ с любого устройства
* редактор сценариев работы устройств
* web-интерфейс с обновлением в реальном времени
* push-уведомления
* интеграция со сторонними веб-сервисами (сервис погоды)
* модель безопасности с разграничением доступа между пользователями
* просмотр статистики работы устройств
* пользователю доступны данные всех устройств без их сохранения в облаке
* синхронизация состояния устройств, управляемых непосредственно и через web-интерфейс
* plug and play(PnP) авто определение новых подключённых устройств

Web-сервис представляет собой сервер-клиентское приложение, в котором сервер находится на облачной PaaS (Platform as a Service) платформе Heroku. Клиентская часть выполняется в браузере. Клиент выполняет запросы на сторонние сервисы, такие как сервис погоды openWeatherMap, используя его API а также устанавливает соединение с домашним сервером автоматизации, через который происходит управление устройствами и от которого приходят уведомления о состоянии системы в реальном времени.

Разработка web-приложения производилась с помощью технологий Ruby on Rails, AngularJS, jQuery, Bootstrap, PatternFly, HTML5. Приложение имеет REST (Representation state transfer) архитектурный стиль. Ruby on Rails реализует паттерн MVC (Model-View-Controller), AngularJS – MVW (Model-View-Whatever).