# A Smart, Energy Efficient Home

## Stage 2 Progress Report

F29SO - Group Project
2019 - 2020

**Group 3 |** Sinderet Green | ***Uplink***

| | |
|---|---|
| Benjamin Milne | <bm56@hw.ac.uk> |
| Samuel Barnett | <sb95@hw.ac.uk> |
| Adam Sterling | <as317@hw.ac.uk> |
| Andrew Sime | <ass8@hw.ac.uk> |
| Euan Gordon | <ejg9@hw.ac.uk> |
| **Manager:** | Dr. Rob Stewart |

# Table of Contents

**1 Introduction** **2**
- 1.1 Purpose 2
- 1.2 Scope 2
- 1.3 Glossary 3
- 1.4 References 3
- 1.5 Roles 4

**2 Development Overview** **5**
- 2.1 Summary 5
- 2.2 Product Overview 5
  - For the homeowners 6
  - For the property managers 6
- 2.3 HUB Environment 7
- 2.4 Progress per Functional Requirement 8
- 2.4 Non-Functional Requirements 10

**3 In-Depth Overview** **11**
- 3.1 Front-end Development 11
  - 3.1.1 Interface Changes Illustrated 12
  - 3.1.2 Testing for Technical Correctness in the Front-end 14
  - 3.1.3 Front-end Functionality and Next Steps 14
- 3.2 Back-end Development 15
  - 3.2.1 Module Architecture 15
  - 3.2.2 Major Changes 16

**4 Next Steps** **18**

# 1 Introduction

## 1.1 Purpose

The purpose of the Stage 2 Progress Report ( ‘the Report’ ) is to detail the development progress of the Uplink smart-home control software made between the 28th of November 2019 and the 6th of February 2020.

The Report provides a high level overview of the product as it is currently planned and a detailed breakdown of the progress towards each requirement as laid out in the Stage 1 Report.

The Report is a confidential document intended to be viewed only by those listed below:

| | |
|---|---|
| The Customer | **Team Esteem** |
| The Project Manager | **Dr. Rob Stewart** |
| The Developers | **Sinderet Green** ( ‘Team Uplink’ ) |

## 1.2 Scope

The Uplink project aims to produce a smart-home control system ( ‘the System’ ) capable of interacting with a multitude of smart devices, record readings from devices & sensors and use recorded data to advise the Customer on eco-efficient energy practices.

## 1.3 Glossary

| | |
|---|---|
| MQTT | Message Queue Telemetry Transport for inter-device communication |
| Mosquitto | MQTT relayer |
| JavaScript | Primarily web-based programming language |
| Node.js | JavaScript-based back-end language |
| Vue.js | JavaScript-based front-end library |
| SQLite | Lightweight database controller |
| HTML | HyperText Markup Language |
| CSS | Cascading Style Sheets |
| Raspberry Pi | Low-power single-board computer |
| Uplink HUB | The device where the Uplink software will be based |
| Linux | Operating System alternative to Windows / OS X |
| JSON | JavaScript Object Notation for standardised data transfers |
| IDE | Integrated Development Environment |
| VCS | Version Control System |
| API | Application Programming Interface |

## 1.4 References

- Uplink ( 2019 ). Stage 1 Report

## 1.5 Roles

We have assigned the following roles for stage 2 based upon availability and experience:

| Role | Dept. |
|---|---|
| **Branding (.COM)** | |
| Organisational Manager, web dev. | Samual Bernett |
| Group Reporter, web dev. | Andrew Sime |
| **Application** | |
| Technical Manager | Benjamin Milne |
| Back-end | |
| Front-end | Euan Gordon |
| | Adam Sterling |

Unfortunately one of our team, Robert Hardiment <rh187@hw.ac.uk> has had to withdraw from the project. This has placed us slightly behind schedule, but we do not foresee this having a notable impact on the process.

# 2 Development Overview

## 2.1 Summary

In summary, our focus throughout Stage 2 has been developing the majority of the back-end API, building the general layout and building blocks of the interface and the company website.

## 2.2 Product Overview

Our product will be targeted towards two groups:

1. Owners of eco-efficient smart homes
2. Property managers overseeing a portfolio of smart homes

Our product will be designed to link up with a vast range of smart devices, expanding the smart eco-system to every corner of the home - and beyond. Our system will be able to give smart suggestions and make decisions to aid the owners or property managers in reducing their carbon footprint and living a healthier, more economical life.

### For the homeowners

We will partner with home builders to provide our system as standard when purchasing a new, eco-efficient home. The system will be based around an ‘uplink-hub’ which will communicate with the ecosystem, record smart data and provide the homeowner with an interface to view and configure their smart home.

We will also provide these smart devices and our uplink-hub aftermarket for homeowners looking to introduce our smart ecosystem to their home.

The system will provide the homeowner with a one-stop-shop to control devices and view reports from around their smart home. It will allow them to enable and disable devices, set triggers, alarms, timers, and view a plethora of reports on data such as temperature, humidity and energy.

The smart home will allow the homeowner to divide their devices and reports into rooms, making it easier to get the data they need, when they need it. It will also allow the triggering of groups of devices.

### For the property managers

Property managers employing our system will be able to fetch data from multiple uplink-hubs and view the same data as the homeowners themselves through one central view. They will be able to configure varying levels of control the homeowner has over the system, and set similar triggers and alarms as the homeowner might through their own panel and the homeowners’.

## 2.3 HUB Environment

The HUB System will be hosted upon a small single-board computer provided to the Customer to connect within their home network. The device will run a light-weight version of Linux such as Raspbian, although the System will also run in Windows and OS X environments. The device will have a small screen to aid in its initial setup.

The Interface provided by the System will be available to any devices on the network provided their Internet Browser supports JavaScript ECMAScript 5 (2009) or above.

The technologies used to build the system are as follows:

| | |
|---|---|
| **Device** | |
| Linux | Operating System |
| **Back-end** | |
| Node.js | Primary server language |
| SQLite | Data storage |
| Mosquitto | Broker for relaying MQTT messages sent between devices and the HUB |
| JSON | For database and front-end / API communication |
| **Front-end** | |
| Vue.js | Responsive JavaScript library |
| JSON | For back-end / API communication |
| **Development Tools** | |
| Visual Studio Code | Light-weight Microsoft source-code editor |
| GitHub | Version Control System |
| BSS | Bootstrap Studio for the company website |

## 2.4 Progress per Functional Requirement

| Req. Code | Req. Explanation | Status |
|---|---|---|
| **F-M-1** | Friendly and graphical user interface | 🟩 |
| The general style of the interface has been solidified, as have the majority of the blocks ( ‘cards’ ) used to display information. We believe we have taken into account the usability study of the Stage 1 Report and created an inoffensive, universally accessible interface for the System. | | |
| **F-M-2** | Display and record electrical usage (in/out) | 🟨 |
| Currently we have the ability to show the details of electrical usage via our interface. We are currently in the process of collecting mock sample data that can be shown to reflect and simulate power usage within a house. To be completed by the end of sprint 6. | | |
| **F-M-3** | Display and record heat levels | 🟨 |
| Currently we have the ability to show the details of temperature levels via our interface. We are currently in the process of collecting mock sample data that can be shown to reflect and simulate heat levels within a house. To be completed by the end of sprint 6. | | |
| **F-M-4** | Enable and disable devices | 🟩 |
| The ability to turn on or off devices has been completed. This includes the ability to set timers to devices that can be added to a schedule. These timers will trigger a device action at a set time that has been set out to the user. | | |
| **F-M-5** | Reminder and warning notifications | 🟥 |

| | | |
|---|---|---|
| **F-M-6** | The service should be available 24 / 7 | 🟩 |
| The service is active 24 / 7 due to the service not needing access to the internet and due to the system functioning entirely locally. The System has an inbuilt database to store details relating to devices, energy usage and users; this is connected directly to our front-end via our api. | | |
| **F-M-7** | Available on multiple devices | 🟩 |
| The web app works directly with all devices due to the ability to be viewed on any web browser. | | |
| **F-S-8** | Property manager's view | 🟥 |
| **F-S-9** | User must be able to delete data | 🟥 |
| **F-S-10** | Configurable users and permissions | 🟥 |
| **F-C-11** | Calculate energy savings and difference over time | 🟨 |
| Currently we have the ability to show the details of energy savings via our interface. We are currently in the process of collecting mock sample data that can be shown to reflect and simulate energy savings and cost per minute or hour to keep devices on. To be completed by the end of sprint 7. | | |
| **F-C-12** | Historical data control | 🟨 |
| Currently starting to develop the interface and pair with the API to delete historical data. | | |
| **F-W-13** | Add / configure any number of devices | 🟩 |
| Any number of devices can be stored in and used via the System. | | |

## 2.4 Non-Functional Requirements

| Req. Code | Req. Explanation | Status |
|---|---|---|
| **NF-M-1** | Logging service must record data to SQL database | 🟩 |
| This has been accomplished using Node.js and the SQLite database manager. | | |
| **NF-M-2** | Devices should communicate over MQTT | 🟨 |
| This has been completed, however we are looking to expand the ways in which we communicate with devices to accomodate for other manufacturers. | | |
| **NF-M-3** | Configuration saved to persistent storage | 🟩 |
| This has been accomplished. | | |
| **NF-M-4** | API must retrieve data from SQL database | 🟨 |
| In progress! The API can retrieve most data by ID, devices and sensors by room, and filter data by timescales provided in UNIX time. | | |
| **NF-S-5** | MQTT protocol should use WebSockets | 🟥 |
| This is an extra layer of security which however hinders development, so it is planned to be enabled at the end of the development process. We do not foresee any difficulties with this. | | |
| **NF-S-6** | Minimal / instant response time for configuring devices | 🟨 |
| As we have not yet finished the API, we have not optimised the way in which devices and the HUB communicate. | | |
| **NF-S-7** | HUB should be able to report data upstream | 🟥 |
| The HUB is not currently able to report data upstream, as the upstream system does not yet exist. | | |

# 3 In-Depth Overview

## 3.1 Front-end Development

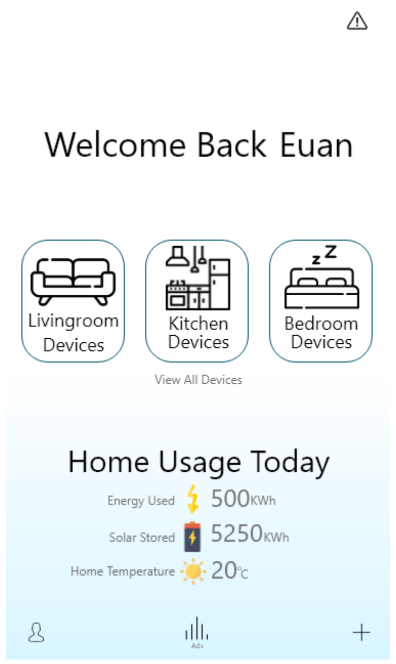
The systems front end was changed to reflect the results of the Stage 1 usability test and to reflect modern design patterns in the industry - mobile design. The interface was redesigned to feature cards that would contain important information relating to devices, rooms and details about energy usage and temperature per room. These cards make the interface more blocky and overall making it simple and easy to use - especially for mobile devices.

The use of cards to store and present data relating to devices and rooms has made generating the interface quick and simple. By grouping data taken from the API in cards, dynamically generating them via vue.js has been very simple thus speeding up the implementation of the frontend designs. By having cards of grouped information we have made it easier for the user to use the system while on a mobile device. Due to the cards size and style we made it a clickable item to which it would lead them to another page or to additional information related to the card they had clicked on.

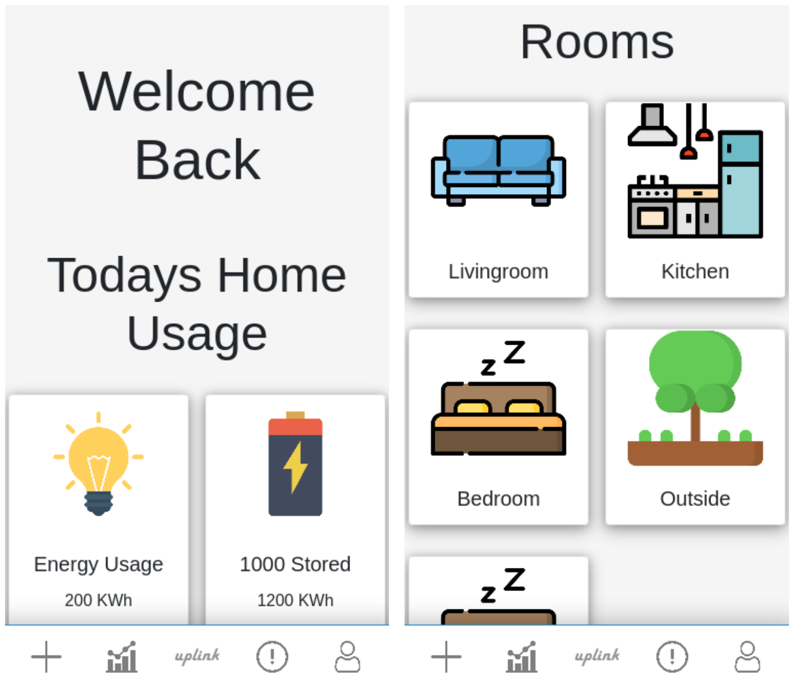
Completed sprints - 1 - 7 that focused on login security, adding devices, adding rooms, adding timers for schedules, ability to turn on and off devices, solidify UI and for the application to dynamically generate pages with data that relates to the page the user is currently on.

Currently on sprint 8 for front-end focusing on charting and management of data along with device and user profile settings.

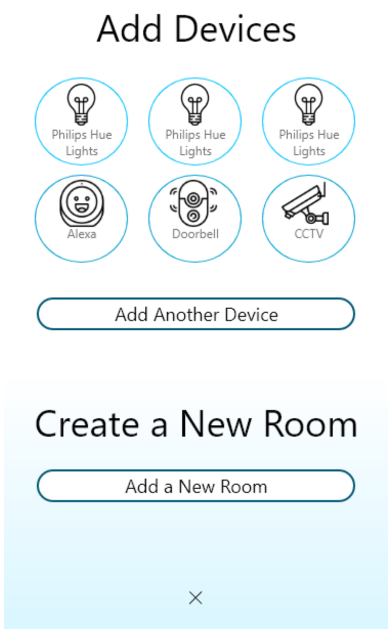
### 3.1.1 Interface Changes Illustrated



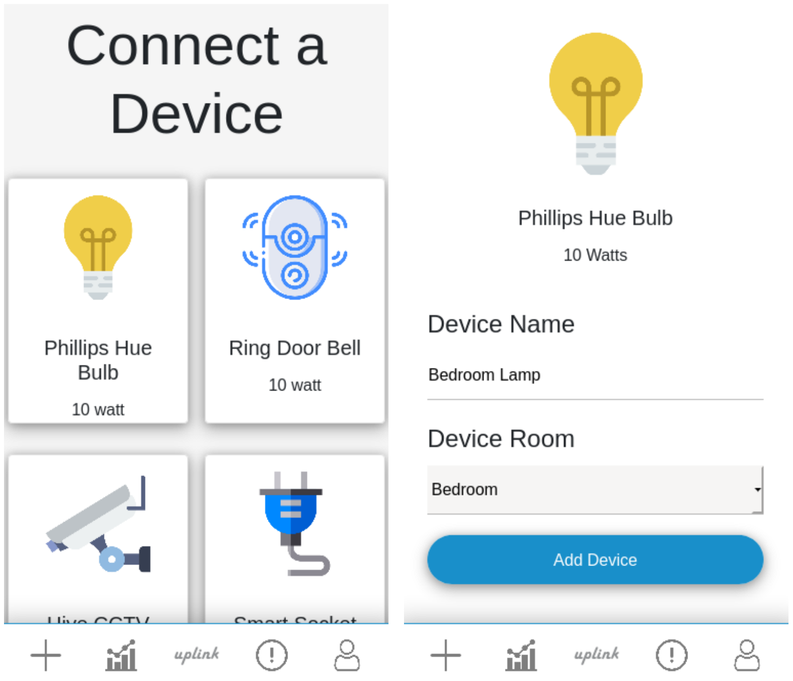
[ Fig. 1: Original Dashboard ]



[ Fig. 2: Current Dashboard ]



[ Fig. 3: Original Add Devices Page ]



[ Fig. 4: Current Add Devices Page ]

### 3.1.2 Testing for Technical Correctness in the Front-end

Due to continuous integration with the backend API and database the front-end is technically correct due to the correct calls to the API. Simple tests were used on functions created in the frontend to either parse or sort data collected by an API call.

### 3.1.3 Front-end Functionality and Next Steps

Currently the frontend has the ability to add devices and rooms to the database via our API. Add schedules for devices using repeat timers and also turn on and off devices. We currently have a login system that generates an authentication token that's used to gain access to the API. The token is also saved in a cookie for the ability to stay logged in. The frontend dynamically sorts and displays data for each room page and device card.

We currently have the ability to display **current energy usage** and **temperature levels** on each page but we are yet to supply this with data that reflects the devices that are connected to the system. We are also yet to supply data to the frontend that reflects solar panel intake energy and usage of that energy. From the data that is yet to be supplied we are planning to incorporate comprehensive graphs of this information that the user can use to get details of their energy usage. We aim to incorporate filters that can be used to get a more indepth look at the data collected.

Stage 3 will incorporate **push notifications** and **warnings** that will be triggered by events that happen when the app is running. These should be displayed in the warnings section and should display a suggestion to fix the issue that triggered the notification. These issues could be the home temperature being too hot or the home humidity being too high. The suggestions should be related to the issue like turn off the heating or open a window. For clarity for the user the warnings section should group warnings and notifications by room of the issue and by severity of the issue.

We also plan to have some level of automation that the user can add and control. This automation service should be used to let the Uplink HUB control certain aspects of the home when a trigger is triggered. This could be when the house is cold the system will automatically trigger the heating to come on. The frontend for this service should be simple to use and also allowing the user maximum control of the automation.

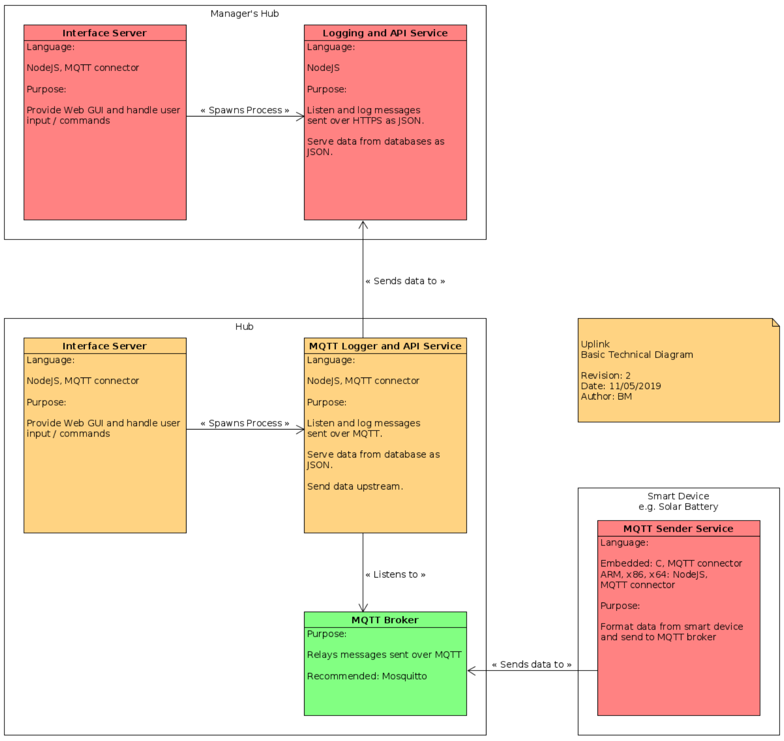
## 3.2 Back-end Development

Much of the back-end functionality has been implemented, including but not limited to;

- Listening for and logging device and sensor messages
- Around 60% of database queries
- Around 60% of API calls
    - Majority of INSERT and SELECT API calls
- Full authentication support
    - Secure token generation and storage
    - Authorization header and token-based authentication
- Timer support
    - Repeat / regular timers
    - 'Oneshot' or one-off timers

### 3.2.1 Module Architecture

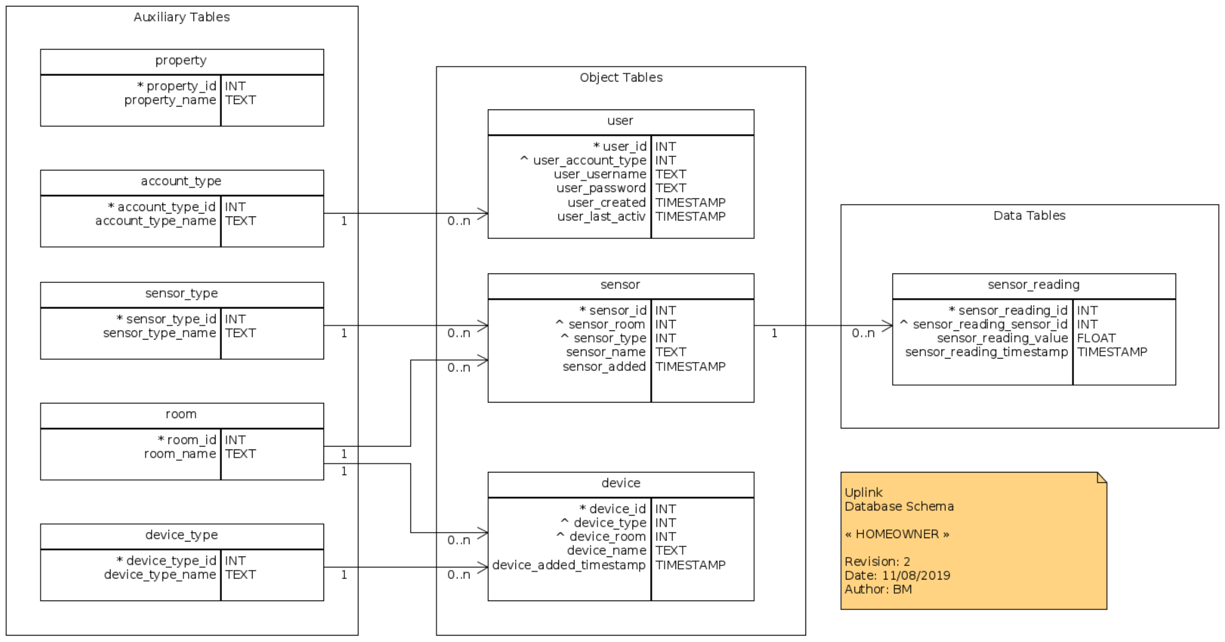
The below diagram illustrates the way in which the different modules within the Uplink HUB System interact and what role they play. The general layout has not changed. The Uplink HUB will provide the storage and interface for the Customer's System.



[ Fig. 5: High level module overview ]

### 3.2.2 Major Changes

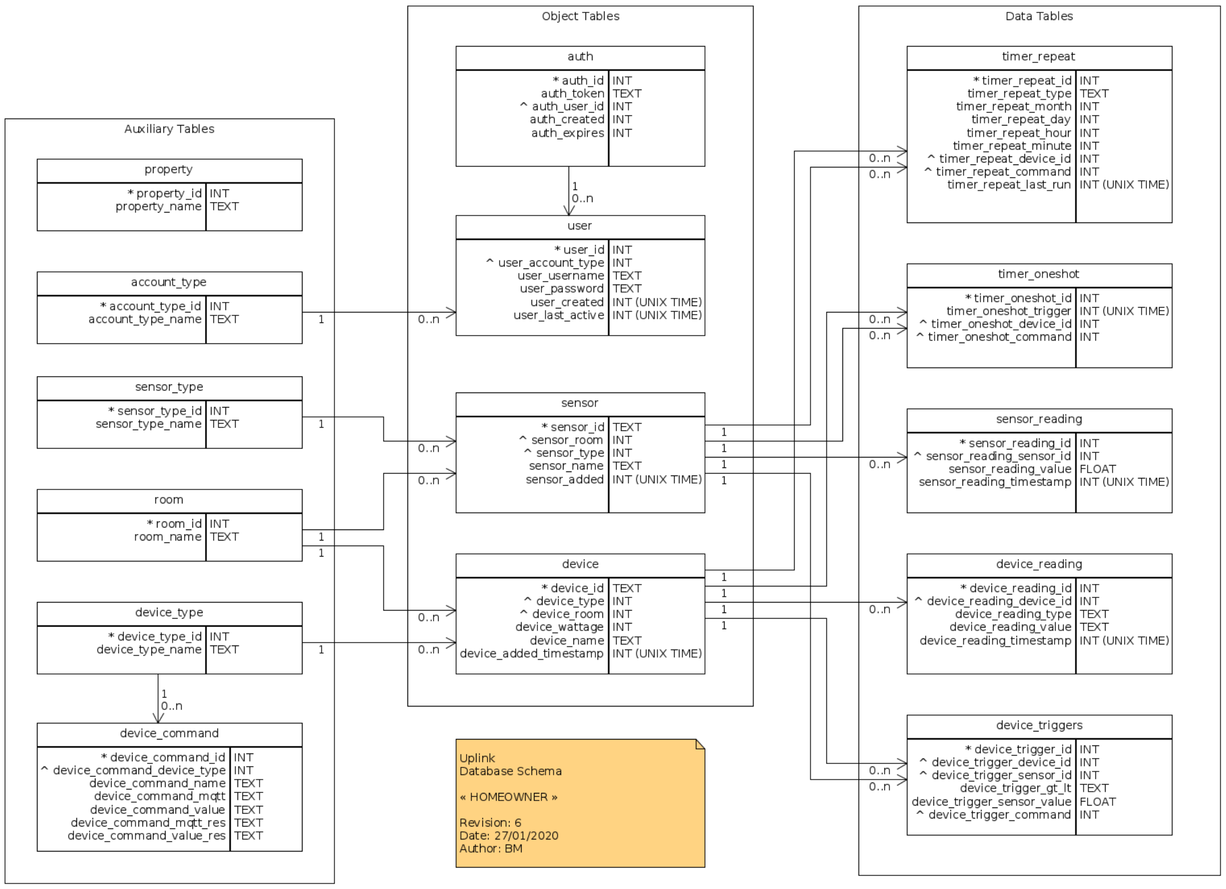
The only major change to the back-end has been to the database schema, to accommodate additional ( but planned ) functionality. Below is the original schema, which stored information on, device / sensor types and instances, sensor data, basic account data and room data:



[ Fig. 6: Original Stage 1 database schema / ER diagram ]

Changes have now been made to accomodate the following features:

- Repeat timers
- Oneshot timers
- Device readings
- Device commands
- Authentication tokens
- Device triggers



[ Fig. 7: Stage 2 database schema / ER diagram]

We plan to make further additions to include user permissions, customizability and compatibility with other brands of devices which do not support the MQTT protocol.

## 4 Next Steps

The project's next steps involve immediate work on the System's **data presentation** in terms of charts, graphs, etc., **push notifications** and **triggers.** We also plan to refine and optimise the Interface for mobile devices and add additional System settings such as data retention policies.

Following a meeting with the team's Project Manager and Proxy Customer, we have also committed to additional market research regarding existing smart-home products and how they may be linked with the Uplink System.

This will involve a re-work of the database schema and the way in which the HUB communicates with devices -- however this should not be too difficult.