Programming Project Databases - Final Report

Groep 4

Jens Bijtebier
David Danssaert
Thanh Danh Le
Filip Smets
Nisse Strauven

17 mei 2016

1 Status

1.1 Homepage

1.1.1 Appealing to the end user

The Homepage is where the end user will be when he or she first visits SmartHome. Since first impressions are important, we made sure to make the homepage look appealing, and added some example graphs to showcase the features of our project.

(Thanh, Jens)

1.2 Translation

1.2.1 Dutch - English

The footer features a Dutch and English flag, used to switch languages. Everything on the website is translated in both languages, and they can be switched on the spot, at any time.
(Filip, Jens)

1.3 Login and Registration

1.3.1 Simple and elegant

We have added both the register and login button to the top-right of the screen. Both buttons open a pop-over dialog, which makes for a smooth transition. This is one of the reasons we went for React in combination with Material-UI. Everything happens on one page. No refreshing and unnecessary loading times needed.

(Filip)

1.4 Households and Sensors

1.4.1 Creation

Although the simulation software doesn't really support this, the user can create both households and sensors. Creating a household opens a new dialog where the user can fill in the name, country, city, zip code, street, and housenumber. Likewise, creating a sensor opens another dialog where the user can provide a name, household, description, power unit, and tags.

(Thanh, Jens)

1.4.2 Viewing

The "View Households"menu changes the layout of the page so that it now displays a card for every household. The graph type can be chosen by the user (Line, Bar, Radar). It is also possible to change how the usage is shown (price in euros, or the original format).

(Filip)

1.4.3 Filtering

Within the Household cards, the user can select which sensor should be shown (a combination of sensors is also possible). It is also possible to specify a time period and graph type, allowing for maximum flexibility.

(Filip)

1.4.4 Tags and Editing

Household cards also contain a button that can be used to edit a sensor's tags. This button opens a new dialog where tags can be added and removed to the user's liking.

(Jens)

1.5 Social Aspect

1.5.1 Friends

Using the friend search feature, it is possible to find other people by name and send them a friend request. Users need to be friends before they can see each other's posts and shared graphs.

(Thanh)

1.5.2 Notifications

Users can find any pending friend requests in their notifications menu, where they can choose to approve said request. Once the request is approved, they will be able to see posts and compare their energy consumption using the Ranking feature.

(Jens)

1.5.3 Wall

The Wall is where all the user's friends' posts and shared graphs will be displayed, much like the activity feed on the Facebook homepage. Shared graphs can still be viewed using a graph type chosen by the end user.

There is also a button that opens a dialog for posting a new message.

(Jens)

1.5.4 Sharing Graphs

Every card in the "View Household"menu has a "Share Graph"button that opens up a new dialog where the graph can be shared, along with a textfield for any additional info the user might want provide.
(Jens)

1.6 Back end

1.6.1 Django

The backend is entirely written in python using the Django web framework. This framework uses the Model View Controller design pattern. In our application, the View part exists of the REST Framework.
(David)

1.6.2 Django Rest Framework

The 'Django Rest Framework' is used to implement the REST Server. This is implemented in the form of Django Views.
(David)

1.7 Extra Functionalities

1.7.1 Ranking

The first extra feature we have decided to add is the possibility to see a ranking among friends. This ranking would show which one of your friends on SmartHome has the lowest energy consumption. We feel that this would provide an extra incentive to try and decrease your usage.

It is possible to rank either by total usage, or average usage per household.

(Nisse)

1.7.2 Clustering

Clustering is used to divide sensors in groups according to power usage. Their are 3 categories right now: low-, medium- and high usage. The clustering is implemented using the k-means algorithm.

At first, the clusters were placed at the minimum, half and maximum of the usage domain at the initial step. The results were, however, not very useful. The groups for low- and medium usage were usually small and their means were often very close to each other compared to the high usage group. Therefore we introduced a logarithmic weight function:

$$w(val, max) = val * \log(\frac{val}{max} \cdot (e-1) + 1)$$

This makes sure that difference between small values are equally important as those between large values. Furthermore, the clusters are initialy placed at 2/4, 3/4 and 4/4 of the domain maximum to make sure that the cluster for high usage doesn't take all datapoints. This gave useful clusters for data generated with ElecSim. The low usage cluster usually contains appliances that consume very little all the time (answer machine, clock, ...), the medium usage cluster contains appliances like fax machines, cooking gear, scanners, ... The last cluster usually contains heavy appliances like washmachines, electrical showers, ... We concluded that this division is the most useful for end users.

(David)

2 Design

2.1 Client

2.1.1 React

We have chosen to use React for our client's UI. React describes itself as "the V in MVC", which is exactly what we intend to use it for.

React's component based design allows us to construct new components (such as the graphs, household cards, notifications, and so on), and then use these components to create so called PageComponents. These allow us to switch pages without the need to refresh the browser, ensuring maximum smoothness.

2.1.2 Material-UI

Google's widely used design standard, Material, is used everywhere. Android, along with many smartphone apps and websites use Material. This means the user will recognize the interface, making the UI feel familiar. Furthermore, the interface will react exactly the way the end user expects it to, making Material the best possible option.

The Material-UI Library for react incorporates these design standards in simple components such as appbars, sidebars, and more. This made it possible for us to create a responsive and recognizable design that will do exactly what the user wants it to.

2.1.3 ChartJs

Graphs are the core of SmartHome, so their responsiveness and readability were top priority for us. We have opted to use ChartJs, a React library that provides us with easiliy customizable chart components, which we have incorporated in the household cards.

2.2 Server

2.2.1 Django

The server side implementation is written in Django python. This was chosen because of it's scalabilty, code-efficiency and cross-platform capabilities. The interface is implemented as a REST API Server.

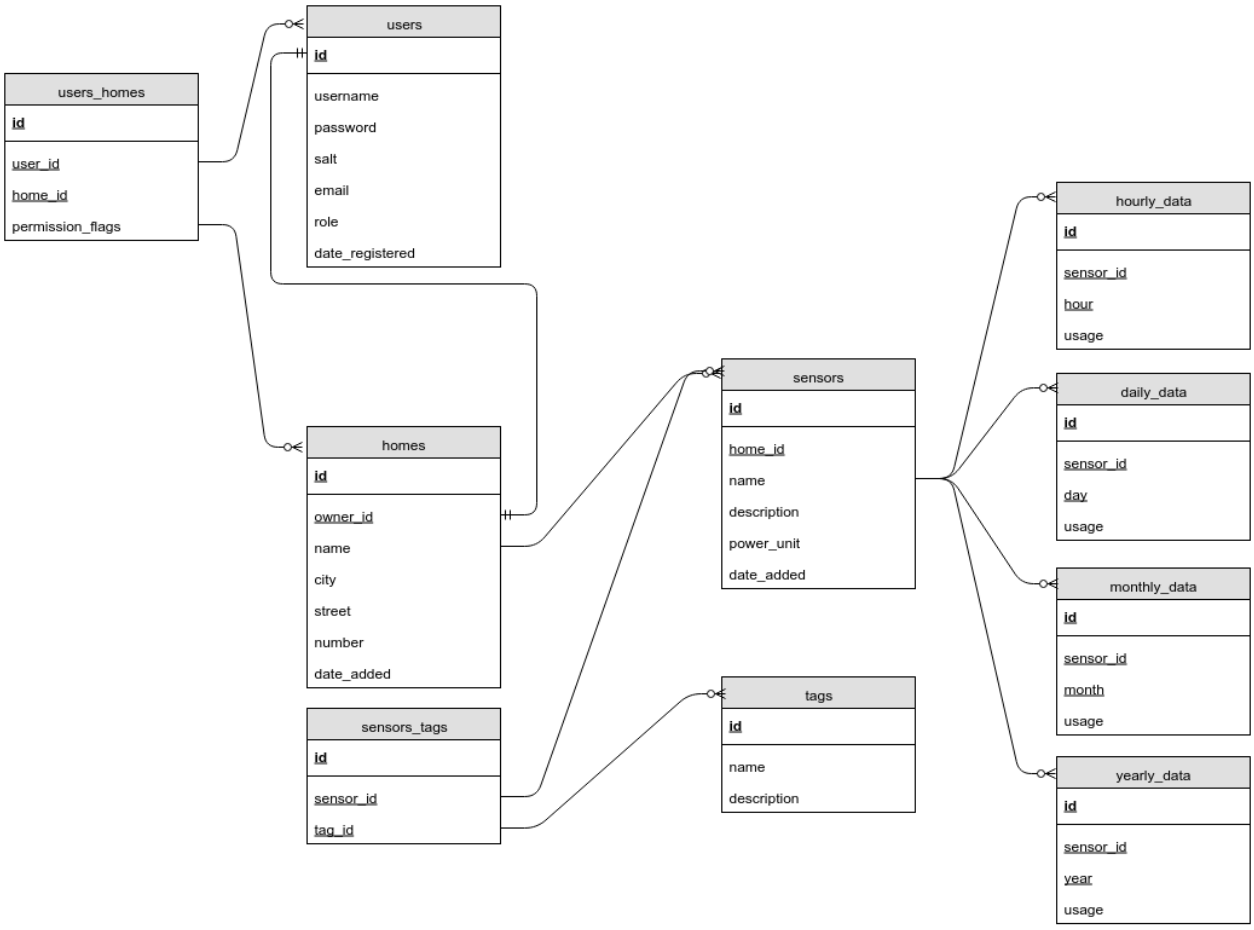
(David)

2.3 REST API

2.3.1 Retrieving data from the server

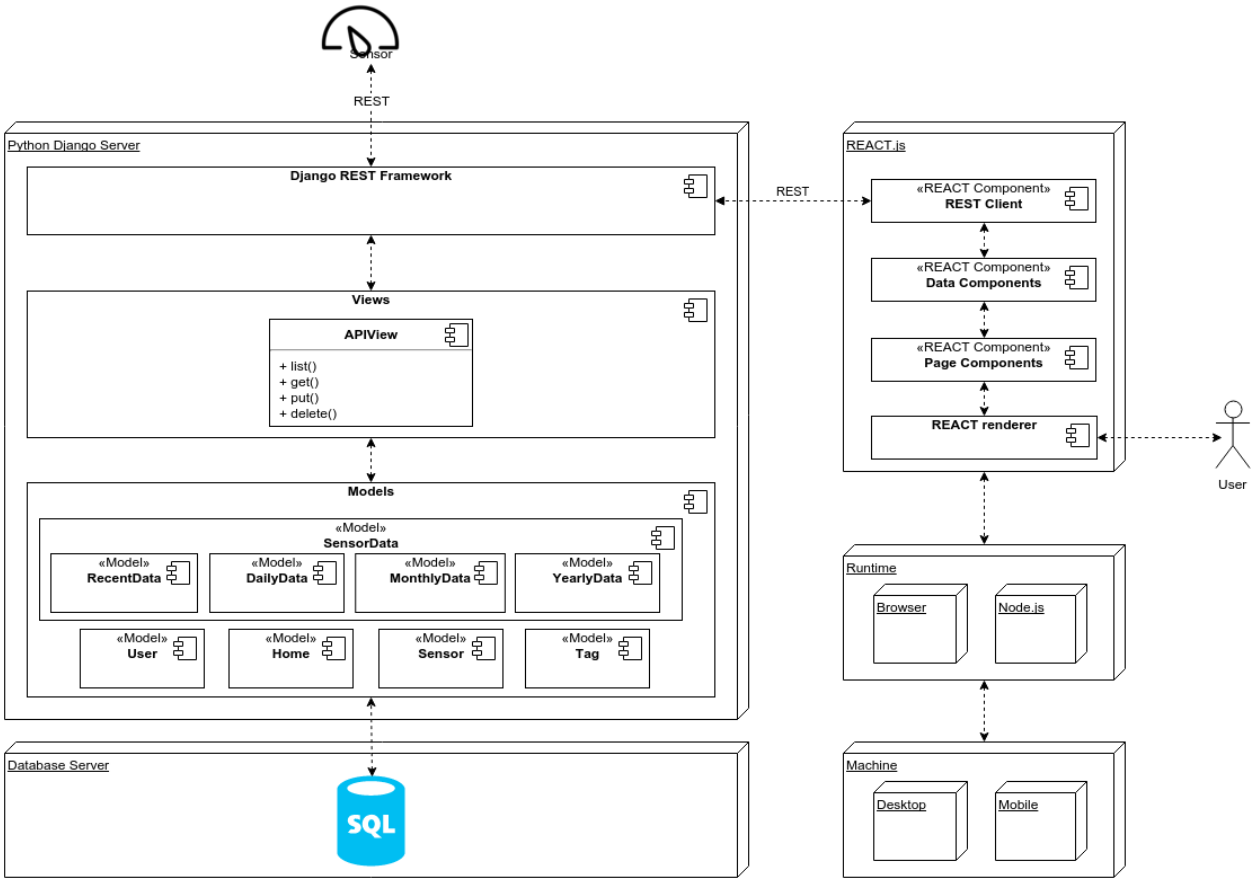
The communication between client and server happens using a well defined REST api. This splits the logic well, and ensures seperation of concerns. Client side, we use plain XMLHttpRequests wrapped in a small homebrew library. Server side, Django provides an interface to make it easy to handle the requests.

2.4 Database Schema



Figuur 1: Het 'Entity Relation Diagram'.

2.5 UML Schema



Figuur 2: Deployment diagram voor server (links) en client (rechts).

3 Product

This section contains a short synopsis of all the required and extra features that we have implemented. For more information about a certain feature, please refer to the Status section.

3.1 Basisvereisten

All requirements have been met. We have implemented a REST interface, to retrieve the sensor data from the server. This data can then be shown in the "View Household"menu, and the user can select one or more sensors and a time period to see exactly what he or she wants. The user can register, log in, create sensors and households. Sensors have editable tags, a name, a power unit, a description, and a household. Households also have all required attributes.

Translations are part of the interface, so that languages can be switched at any time.

On the social side, it is possible to search for other users, and send them a friendrequest. Once they have accepted the request in their notifications menu, it will be possible to see each other's posts. Posts can either be posted using the wall, resulting in a plain text post, or graphs can be shared. When the end users shares a graph, it is possible to add a message to the graph in order to provide some extra information.

An admin interface is accessible for users that are marked as staff.

3.2 Extra functionaliteit

A clustering algorithm has been implemented that divides sensors of each household in groups for low-, medium- and high power usage. The results seem consistent and useful.

We have also added a ranking feature that allows the user to compare energy usage amongst their friends. Users that have a lower average consumption, will rank higher. This promotes saving energy, which is essentially what SmartHome is all about.

4 Appendix

4.1 SQL Queries

Here are some of the most important queries in our application=

```
/* filter users by name */
SELECT id, username, first_name, last_name, email, is_staff,
   is_active, date_joined FROM auth_user
WHERE username LIKE <username> OR first_name LIKE <username> OR
   last_name LIKE <username>;


/* Friends are selected based on approved friend requests */
SELECT auth_user.id, username, first_name, last_name, email,
   is_staff, is_active, date_joined FROM auth_user
INNER JOIN friend_requests
ON auth_user.id = friend_requests.sender_id
WHERE (auth_user.id = <user_id>) OR (friend_requests.status = 1 AND
   friend_requests.receiver_id = <user_id>) OR (friend_requests.
   status = 1 AND friend_requests.sender_id = <user_id>);


/* list the sum total usage of all sensors for each minute of today
   for a given user */
SELECT recent_data.timestamp, sensors.home_id AS home_id, homes.name
    AS home_name, SUM(recent_data.usage) AS usage
FROM recent_data
INNER JOIN sensors
```

```
ON (recent_data.sensor_id = sensors.id)
INNER JOIN homes ON (sensors.home_id = homes.id)
WHERE (homes.owner_id = <user_id> AND recent_data.timestamp >=
   DATE_SUB(NOW(), INTERVAL 1 DAY) AND recent_data.timestamp < NOW()
    )
GROUP BY recent_data.timestamp, sensors.home_id, homes.name
ORDER BY home_id ASC, recent_data.timestamp ASC;


/* Aggregate all minutely data from today into one record for all
   sensors */
INSERT INTO daily_data (sensor_id, timestamp, usage, n_measurements)
SELECT recent_data.sensor_id, CONCAT(DATE(NOW()), ' 00:00:00') AS
   new_date, AVG(recent_data.usage), SUM(recent_data.n_measurements)
FROM recent_data
WHERE ( recent_data.timestamp >= new_date AND recent_data.timestamp
   < DATE_ADD(new_date, INTERVAL 1 DAY) )
GROUP BY recent_data.sensor_id
```

4.2 SQL Database Structure

This is the database structure. All tables that are custom defined, and not part of Django, are included. With exception for the 'Users' table, because the authentication functionality from Django is used.

```
--
-- Tabelstructuur voor tabel `account_emailaddress`
--

CREATE TABLE IF NOT EXISTS `account_emailaddress` (
  `id` int(11) NOT NULL,
  `email` varchar(254) NOT NULL,
  `verified` tinyint(1) NOT NULL,
  `primary` tinyint(1) NOT NULL,
  `user_id` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Tabelstructuur voor tabel `account_emailconfirmation`
--

CREATE TABLE IF NOT EXISTS `account_emailconfirmation` (
  `id` int(11) NOT NULL,
  `created` datetime(6) NOT NULL,
```

```
  `sent` datetime(6) DEFAULT NULL,
  `key` varchar(64) NOT NULL,
  `email_address_id` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Tabelstructuur voor tabel `authtoken_token`
--

CREATE TABLE IF NOT EXISTS `authtoken_token` (
  `key` varchar(40) NOT NULL,
  `created` datetime(6) NOT NULL,
  `user_id` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Tabelstructuur voor tabel `auth_group`
--

CREATE TABLE IF NOT EXISTS `auth_group` (
  `id` int(11) NOT NULL,
  `name` varchar(80) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Tabelstructuur voor tabel `auth_group_permissions`
--

CREATE TABLE IF NOT EXISTS `auth_group_permissions` (
  `id` int(11) NOT NULL,
  `group_id` int(11) NOT NULL,
  `permission_id` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Tabelstructuur voor tabel `auth_permission`
--
```

```
CREATE TABLE IF NOT EXISTS `auth_permission` (
  `id` int(11) NOT NULL,
  `name` varchar(255) NOT NULL,
  `content_type_id` int(11) NOT NULL,
  `codename` varchar(100) NOT NULL
) ENGINE=InnoDB AUTO_INCREMENT=76 DEFAULT CHARSET=latin1;

--
-- Gegevens worden gexporteerd voor tabel `auth_permission`
--

INSERT INTO `auth_permission` (`id`, `name`, `content_type_id`, `
   codename`) VALUES
(1, 'Can add log entry', 1, 'add_logentry'),
(2, 'Can change log entry', 1, 'change_logentry'),
(3, 'Can delete log entry', 1, 'delete_logentry'),
(4, 'Can add permission', 2, 'add_permission'),
(5, 'Can change permission', 2, 'change_permission'),
(6, 'Can delete permission', 2, 'delete_permission'),
(7, 'Can add group', 3, 'add_group'),
(8, 'Can change group', 3, 'change_group'),
(9, 'Can delete group', 3, 'delete_group'),
(10, 'Can add user', 4, 'add_user'),
(11, 'Can change user', 4, 'change_user'),
(12, 'Can delete user', 4, 'delete_user'),
(13, 'Can add content type', 5, 'add_contenttype'),
(14, 'Can change content type', 5, 'change_contenttype'),
(15, 'Can delete content type', 5, 'delete_contenttype'),
(16, 'Can add session', 6, 'add_session'),
(17, 'Can change session', 6, 'change_session'),
(18, 'Can delete session', 6, 'delete_session'),
(19, 'Can add site', 7, 'add_site'),
(20, 'Can change site', 7, 'change_site'),
(21, 'Can delete site', 7, 'delete_site'),
(22, 'Can add token', 8, 'add_token'),
(23, 'Can change token', 8, 'change_token'),
(24, 'Can delete token', 8, 'delete_token'),
(25, 'Can add email address', 9, 'add_emailaddress'),
(26, 'Can change email address', 9, 'change_emailaddress'),
(27, 'Can delete email address', 9, 'delete_emailaddress'),
(28, 'Can add email confirmation', 10, 'add_emailconfirmation'),
(29, 'Can change email confirmation', 10, 'change_emailconfirmation'
   ),
```

```
(30, 'Can␣delete␣email␣confirmation', 10, 'delete_emailconfirmation'
   ),
(31, 'Can␣add␣social␣application', 11, 'add_socialapp'),
(32, 'Can␣change␣social␣application', 11, 'change_socialapp'),
(33, 'Can␣delete␣social␣application', 11, 'delete_socialapp'),
(34, 'Can␣add␣social␣account', 12, 'add_socialaccount'),
(35, 'Can␣change␣social␣account', 12, 'change_socialaccount'),
(36, 'Can␣delete␣social␣account', 12, 'delete_socialaccount'),
(37, 'Can␣add␣social␣application␣token', 13, 'add_socialtoken'),
(38, 'Can␣change␣social␣application␣token', 13, 'change_socialtoken'
   ),
(39, 'Can␣delete␣social␣application␣token', 13, 'delete_socialtoken'
   ),
(40, 'Can␣add␣cors␣model', 14, 'add_corsmodel'),
(41, 'Can␣change␣cors␣model', 14, 'change_corsmodel'),
(42, 'Can␣delete␣cors␣model', 14, 'delete_corsmodel'),
(43, 'Can␣add␣home', 15, 'add_home'),
(44, 'Can␣change␣home', 15, 'change_home'),
(45, 'Can␣delete␣home', 15, 'delete_home'),
(46, 'Can␣add␣sensor', 16, 'add_sensor'),
(47, 'Can␣change␣sensor', 16, 'change_sensor'),
(48, 'Can␣delete␣sensor', 16, 'delete_sensor'),
(49, 'Can␣add␣sensors␣tags', 17, 'add_sensorstags'),
(50, 'Can␣change␣sensors␣tags', 17, 'change_sensorstags'),
(51, 'Can␣delete␣sensors␣tags', 17, 'delete_sensorstags'),
(52, 'Can␣add␣tag', 18, 'add_tag'),
(53, 'Can␣change␣tag', 18, 'change_tag'),
(54, 'Can␣delete␣tag', 18, 'delete_tag'),
(55, 'Can␣add␣users␣homes', 19, 'add_usershomes'),
(56, 'Can␣change␣users␣homes', 19, 'change_usershomes'),
(57, 'Can␣delete␣users␣homes', 19, 'delete_usershomes'),
(58, 'Can␣add␣yearly␣data', 20, 'add_yearlydata'),
(59, 'Can␣change␣yearly␣data', 20, 'change_yearlydata'),
(60, 'Can␣delete␣yearly␣data', 20, 'delete_yearlydata'),
(61, 'Can␣add␣monthly␣data', 21, 'add_monthlydata'),
(62, 'Can␣change␣monthly␣data', 21, 'change_monthlydata'),
(63, 'Can␣delete␣monthly␣data', 21, 'delete_monthlydata'),
(64, 'Can␣add␣daily␣data', 22, 'add_dailydata'),
(65, 'Can␣change␣daily␣data', 22, 'change_dailydata'),
(66, 'Can␣delete␣daily␣data', 22, 'delete_dailydata'),
(67, 'Can␣add␣recent␣data', 23, 'add_recentdata'),
(68, 'Can␣change␣recent␣data', 23, 'change_recentdata'),
(69, 'Can␣delete␣recent␣data', 23, 'delete_recentdata'),
(70, 'Can␣add␣friend␣request', 24, 'add_friendrequest'),
(71, 'Can␣change␣friend␣request', 24, 'change_friendrequest'),
```

```
(72, 'Can␣delete␣friend␣request', 24, 'delete_friendrequest'),
(73, 'Can␣add␣post', 25, 'add_post'),
(74, 'Can␣change␣post', 25, 'change_post'),
(75, 'Can␣delete␣post', 25, 'delete_post');

-- --------------------------------------------------------

--
-- Tabelstructuur voor tabel `auth_user`
--

CREATE TABLE IF NOT EXISTS `auth_user` (
  `id` int(11) NOT NULL,
  `password` varchar(128) NOT NULL,
  `last_login` datetime(6) DEFAULT NULL,
  `is_superuser` tinyint(1) NOT NULL,
  `username` varchar(30) NOT NULL,
  `first_name` varchar(30) NOT NULL,
  `last_name` varchar(30) NOT NULL,
  `email` varchar(254) NOT NULL,
  `is_staff` tinyint(1) NOT NULL,
  `is_active` tinyint(1) NOT NULL,
  `date_joined` datetime(6) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Tabelstructuur voor tabel `auth_user_groups`
--

CREATE TABLE IF NOT EXISTS `auth_user_groups` (
  `id` int(11) NOT NULL,
  `user_id` int(11) NOT NULL,
  `group_id` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Tabelstructuur voor tabel `auth_user_user_permissions`
--

CREATE TABLE IF NOT EXISTS `auth_user_user_permissions` (
  `id` int(11) NOT NULL,
```

```
  `user_id` int(11) NOT NULL,
  `permission_id` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Tabelstructuur voor tabel `daily_data`
--

CREATE TABLE IF NOT EXISTS `daily_data` (
  `id` int(11) NOT NULL,
  `timestamp` datetime(6) NOT NULL,
  `usage` int(11) NOT NULL,
  `n_measurements` int(11) NOT NULL,
  `sensor_id` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Tabelstructuur voor tabel `friend_requests`
--

CREATE TABLE IF NOT EXISTS `friend_requests` (
  `id` int(11) NOT NULL,
  `status` int(11) NOT NULL,
  `date_sent` datetime(6) NOT NULL,
  `receiver_id` int(11) NOT NULL,
  `sender_id` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Tabelstructuur voor tabel `homes`
--

CREATE TABLE IF NOT EXISTS `homes` (
  `id` int(11) NOT NULL,
  `name` varchar(64) NOT NULL,
  `country` varchar(2) NOT NULL,
  `city` varchar(64) NOT NULL,
  `zipcode` varchar(16) NOT NULL,
  `street` varchar(64) NOT NULL,
```

```
  `house_number` varchar(8) NOT NULL,
  `date_added` datetime(6) NOT NULL,
  `owner_id` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Tabelstructuur voor tabel `messages`
--

CREATE TABLE IF NOT EXISTS `messages` (
  `id` int(11) NOT NULL,
  `content` longtext NOT NULL,
  `plot` longtext NOT NULL,
  `date_sent` datetime(6) NOT NULL,
  `receiver_id` int(11) NOT NULL,
  `sender_id` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Tabelstructuur voor tabel `monthly_data`
--

CREATE TABLE IF NOT EXISTS `monthly_data` (
  `id` int(11) NOT NULL,
  `timestamp` datetime(6) NOT NULL,
  `usage` int(11) NOT NULL,
  `n_measurements` int(11) NOT NULL,
  `sensor_id` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Tabelstructuur voor tabel `recent_data`
--

CREATE TABLE IF NOT EXISTS `recent_data` (
  `id` int(11) NOT NULL,
  `timestamp` datetime(6) NOT NULL,
  `usage` int(11) NOT NULL,
  `n_measurements` int(11) NOT NULL,
```

```
  `sensor_id` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Tabelstructuur voor tabel `sensors`
--

CREATE TABLE IF NOT EXISTS `sensors` (
  `id` int(11) NOT NULL,
  `name` varchar(64) NOT NULL,
  `description` varchar(256) NOT NULL,
  `power_unit` varchar(3) NOT NULL,
  `date_created` datetime(6) NOT NULL,
  `home_id` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Tabelstructuur voor tabel `sensors_tags`
--

CREATE TABLE IF NOT EXISTS `sensors_tags` (
  `id` int(11) NOT NULL,
  `date_created` datetime(6) NOT NULL,
  `sensor_id` int(11) NOT NULL,
  `tag_id` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Tabelstructuur voor tabel `tags`
--

CREATE TABLE IF NOT EXISTS `tags` (
  `id` int(11) NOT NULL,
  `name` varchar(64) NOT NULL,
  `description` varchar(256) NOT NULL,
  `date_created` datetime(6) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------
```

```
--
-- Tabelstructuur voor tabel `users_homes`
--

CREATE TABLE IF NOT EXISTS `users_homes` (
  `id` int(11) NOT NULL,
  `permission_flags` int(11) NOT NULL,
  `date_created` datetime(6) NOT NULL,
  `home_id` int(11) NOT NULL,
  `user_id` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Tabelstructuur voor tabel `yearly_data`
--

CREATE TABLE IF NOT EXISTS `yearly_data` (
  `id` int(11) NOT NULL,
  `timestamp` datetime(6) NOT NULL,
  `usage` int(11) NOT NULL,
  `n_measurements` int(11) NOT NULL,
  `sensor_id` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Indexen voor ge xporteerde tabellen
--

--
-- Indexen voor tabel `account_emailaddress`
--
ALTER TABLE `account_emailaddress`
  ADD PRIMARY KEY (`id`),
  ADD UNIQUE KEY `email` (`email`),
  ADD KEY `account_emailaddress_user_id_2c513194_fk_auth_user_id` (`
     user_id`);

--
-- Indexen voor tabel `account_emailconfirmation`
--
ALTER TABLE `account_emailconfirmation`
  ADD PRIMARY KEY (`id`),
```

```
  ADD UNIQUE KEY `key` (`key`),
  ADD KEY `
     account_ema_email_address_id_5b7f8c58_fk_account_emailaddress_id
     ` (`email_address_id`);

--
-- Indexen voor tabel `authtoken_token`
--
ALTER TABLE `authtoken_token`
  ADD PRIMARY KEY (`key`),
  ADD UNIQUE KEY `user_id` (`user_id`);

--
-- Indexen voor tabel `auth_group`
--
ALTER TABLE `auth_group`
  ADD PRIMARY KEY (`id`),
  ADD UNIQUE KEY `name` (`name`);

--
-- Indexen voor tabel `auth_group_permissions`
--
ALTER TABLE `auth_group_permissions`
  ADD PRIMARY KEY (`id`),
  ADD UNIQUE KEY `auth_group_permissions_group_id_0cd325b0_uniq` (`
     group_id`,`permission_id`),
  ADD KEY `
     auth_group_permissi_permission_id_84c5c92e_fk_auth_permission_id
     ` (`permission_id`);

--
-- Indexen voor tabel `auth_permission`
--
ALTER TABLE `auth_permission`
  ADD PRIMARY KEY (`id`),
  ADD UNIQUE KEY `auth_permission_content_type_id_01ab375a_uniq` (`
     content_type_id`,`codename`);

--
-- Indexen voor tabel `auth_user`
--
ALTER TABLE `auth_user`
  ADD PRIMARY KEY (`id`),
  ADD UNIQUE KEY `username` (`username`);
```

```
--
-- Indexen voor tabel `auth_user_groups`
--
ALTER TABLE `auth_user_groups`
  ADD PRIMARY KEY (`id`),
  ADD UNIQUE KEY `auth_user_groups_user_id_94350c0c_uniq` (`user_id
     `,`group_id`),
  ADD KEY `auth_user_groups_group_id_97559544_fk_auth_group_id` (`
     group_id`);

--
-- Indexen voor tabel `auth_user_user_permissions`
--
ALTER TABLE `auth_user_user_permissions`
  ADD PRIMARY KEY (`id`),
  ADD UNIQUE KEY `auth_user_user_permissions_user_id_14a6b632_uniq`
     (`user_id`,`permission_id`),
  ADD KEY `
     auth_user_user_perm_permission_id_1fbb5f2c_fk_auth_permission_id
     ` (`permission_id`);

--
-- Indexen voor tabel `daily_data`
--
ALTER TABLE `daily_data`
  ADD PRIMARY KEY (`id`),
  ADD KEY `daily_data_d96d866a` (`sensor_id`);

--
-- Indexen voor tabel `friend_requests`
--
ALTER TABLE `friend_requests`
  ADD PRIMARY KEY (`id`),
  ADD KEY `friend_requests_receiver_id_6e370850_fk_auth_user_id` (`
     receiver_id`),
  ADD KEY `friend_requests_sender_id_e02b64d5_fk_auth_user_id` (`
     sender_id`);

--
-- Indexen voor tabel `homes`
--
ALTER TABLE `homes`
  ADD PRIMARY KEY (`id`),
  ADD KEY `homes_owner_id_32e87e52_fk_auth_user_id` (`owner_id`);
```

```
--
-- Indexen voor tabel `messages`
--
ALTER TABLE `messages`
  ADD PRIMARY KEY (`id`),
  ADD KEY `messages_receiver_id_874b4e0a_fk_auth_user_id` (`
     receiver_id`),
  ADD KEY `messages_sender_id_dc5a0bbd_fk_auth_user_id` (`sender_id
     `);

--
-- Indexen voor tabel `monthly_data`
--
ALTER TABLE `monthly_data`
  ADD PRIMARY KEY (`id`),
  ADD KEY `monthly_data_d96d866a` (`sensor_id`);

--
-- Indexen voor tabel `recent_data`
--
ALTER TABLE `recent_data`
  ADD PRIMARY KEY (`id`),
  ADD KEY `recent_data_d96d866a` (`sensor_id`);

--
-- Indexen voor tabel `sensors`
--
ALTER TABLE `sensors`
  ADD PRIMARY KEY (`id`),
  ADD KEY `sensors_home_id_1b6b9891_fk_homes_id` (`home_id`);

--
-- Indexen voor tabel `sensors_tags`
--
ALTER TABLE `sensors_tags`
  ADD PRIMARY KEY (`id`),
  ADD KEY `sensors_tags_sensor_id_ec6d5d5a_fk_sensors_id` (`
     sensor_id`),
  ADD KEY `sensors_tags_76f094bc` (`tag_id`);

--
-- Indexen voor tabel `tags`
--
ALTER TABLE `tags`
  ADD PRIMARY KEY (`id`);
```

```
--
-- Indexen voor tabel `users_homes`
--
ALTER TABLE `users_homes`
  ADD PRIMARY KEY (`id`),
  ADD KEY `users_homes_home_id_6d8097a2_fk_homes_id` (`home_id`),
  ADD KEY `users_homes_user_id_c35d6efe_fk_auth_user_id` (`user_id`)
    ;

--
-- Indexen voor tabel `yearly_data`
--
ALTER TABLE `yearly_data`
  ADD PRIMARY KEY (`id`),
  ADD KEY `yearly_data_sensor_id_349bf5e9_fk_sensors_id` (`sensor_id
    `);

--
-- AUTO_INCREMENT voor ge xporteerde tabellen
--


--
-- AUTO_INCREMENT voor een tabel `account_emailaddress`
--
ALTER TABLE `account_emailaddress`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
--
-- AUTO_INCREMENT voor een tabel `account_emailconfirmation`
--
ALTER TABLE `account_emailconfirmation`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
--
-- AUTO_INCREMENT voor een tabel `auth_group`
--
ALTER TABLE `auth_group`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
--
-- AUTO_INCREMENT voor een tabel `auth_group_permissions`
--
ALTER TABLE `auth_group_permissions`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
--
-- AUTO_INCREMENT voor een tabel `auth_permission`
--
```

```
ALTER TABLE `auth_permission`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT,AUTO_INCREMENT=76;
--
-- AUTO_INCREMENT voor een tabel `auth_user`
--
ALTER TABLE `auth_user`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
--
-- AUTO_INCREMENT voor een tabel `auth_user_groups`
--
ALTER TABLE `auth_user_groups`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
--
-- AUTO_INCREMENT voor een tabel `auth_user_user_permissions`
--
ALTER TABLE `auth_user_user_permissions`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
--
-- AUTO_INCREMENT voor een tabel `daily_data`
--
ALTER TABLE `daily_data`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
--
-- AUTO_INCREMENT voor een tabel `friend_requests`
--
ALTER TABLE `friend_requests`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
--
-- AUTO_INCREMENT voor een tabel `homes`
--
ALTER TABLE `homes`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
--
-- AUTO_INCREMENT voor een tabel `messages`
--
ALTER TABLE `messages`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
--
-- AUTO_INCREMENT voor een tabel `monthly_data`
--
ALTER TABLE `monthly_data`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
--
-- AUTO_INCREMENT voor een tabel `recent_data`
--
```

```
ALTER TABLE `recent_data`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
--
-- AUTO_INCREMENT voor een tabel `sensors`
--
ALTER TABLE `sensors`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
--
-- AUTO_INCREMENT voor een tabel `sensors_tags`
--
ALTER TABLE `sensors_tags`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
--
-- AUTO_INCREMENT voor een tabel `tags`
--
ALTER TABLE `tags`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
--
-- AUTO_INCREMENT voor een tabel `users_homes`
--
ALTER TABLE `users_homes`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
--
-- AUTO_INCREMENT voor een tabel `yearly_data`
--
ALTER TABLE `yearly_data`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
--
-- Beperkingen voor ge xporteerde tabellen
--


--
-- Beperkingen voor tabel `account_emailaddress`
--
ALTER TABLE `account_emailaddress`
  ADD CONSTRAINT `
     account_emailaddress_user_id_2c513194_fk_auth_user_id` FOREIGN
     KEY (`user_id`) REFERENCES `auth_user` (`id`);

--
-- Beperkingen voor tabel `account_emailconfirmation`
--
ALTER TABLE `account_emailconfirmation`
  ADD CONSTRAINT `
     account_ema_email_address_id_5b7f8c58_fk_account_emailaddress_id
```

```
    ` FOREIGN KEY (`email_address_id`) REFERENCES `
    account_emailaddress` (`id`);

--
-- Beperkingen voor tabel `authtoken_token`
--
ALTER TABLE `authtoken_token`
  ADD CONSTRAINT `authtoken_token_user_id_35299eff_fk_auth_user_id`
    FOREIGN KEY (`user_id`) REFERENCES `auth_user` (`id`);

--
-- Beperkingen voor tabel `auth_group_permissions`
--
ALTER TABLE `auth_group_permissions`
  ADD CONSTRAINT `
    auth_group_permissi_permission_id_84c5c92e_fk_auth_permission_id
    ` FOREIGN KEY (`permission_id`) REFERENCES `auth_permission` (`
    id`),
  ADD CONSTRAINT `
    auth_group_permissions_group_id_b120cbf9_fk_auth_group_id`
    FOREIGN KEY (`group_id`) REFERENCES `auth_group` (`id`);

--
-- Beperkingen voor tabel `auth_permission`
--
ALTER TABLE `auth_permission`
  ADD CONSTRAINT `
    auth_permissi_content_type_id_2f476e4b_fk_django_content_type_id
    ` FOREIGN KEY (`content_type_id`) REFERENCES `
    django_content_type` (`id`);

--
-- Beperkingen voor tabel `auth_user_groups`
--
ALTER TABLE `auth_user_groups`
  ADD CONSTRAINT `
    auth_user_groups_group_id_97559544_fk_auth_group_id` FOREIGN
    KEY (`group_id`) REFERENCES `auth_group` (`id`),
  ADD CONSTRAINT `auth_user_groups_user_id_6a12ed8b_fk_auth_user_id`
     FOREIGN KEY (`user_id`) REFERENCES `auth_user` (`id`);

--
-- Beperkingen voor tabel `auth_user_user_permissions`
--
ALTER TABLE `auth_user_user_permissions`
```

```
  ADD CONSTRAINT `
     auth_user_user_perm_permission_id_1fbb5f2c_fk_auth_permission_id
     ` FOREIGN KEY (`permission_id`) REFERENCES `auth_permission` (`
     id`),
  ADD CONSTRAINT `
     auth_user_user_permissions_user_id_a95ead1b_fk_auth_user_id`
     FOREIGN KEY (`user_id`) REFERENCES `auth_user` (`id`);

--
-- Beperkingen voor tabel `daily_data`
--
ALTER TABLE `daily_data`
  ADD CONSTRAINT `daily_data_sensor_id_c70d4260_fk_sensors_id`
     FOREIGN KEY (`sensor_id`) REFERENCES `sensors` (`id`);

--
-- Beperkingen voor tabel `friend_requests`
--
ALTER TABLE `friend_requests`
  ADD CONSTRAINT `
     friend_requests_receiver_id_6e370850_fk_auth_user_id` FOREIGN
     KEY (`receiver_id`) REFERENCES `auth_user` (`id`),
  ADD CONSTRAINT `friend_requests_sender_id_e02b64d5_fk_auth_user_id
     ` FOREIGN KEY (`sender_id`) REFERENCES `auth_user` (`id`);

--
-- Beperkingen voor tabel `homes`
--
ALTER TABLE `homes`
  ADD CONSTRAINT `homes_owner_id_32e87e52_fk_auth_user_id` FOREIGN
     KEY (`owner_id`) REFERENCES `auth_user` (`id`);

--
-- Beperkingen voor tabel `messages`
--
ALTER TABLE `messages`
  ADD CONSTRAINT `messages_receiver_id_874b4e0a_fk_auth_user_id`
     FOREIGN KEY (`receiver_id`) REFERENCES `auth_user` (`id`),
  ADD CONSTRAINT `messages_sender_id_dc5a0bbd_fk_auth_user_id`
     FOREIGN KEY (`sender_id`) REFERENCES `auth_user` (`id`);

--
-- Beperkingen voor tabel `monthly_data`
--
ALTER TABLE `monthly_data`
```

```
  ADD CONSTRAINT `monthly_data_sensor_id_5f3ceaa9_fk_sensors_id`
    FOREIGN KEY (`sensor_id`) REFERENCES `sensors` (`id`);

--
-- Beperkingen voor tabel `recent_data`
--
ALTER TABLE `recent_data`
  ADD CONSTRAINT `recent_data_sensor_id_4cb30b35_fk_sensors_id`
    FOREIGN KEY (`sensor_id`) REFERENCES `sensors` (`id`);

--
-- Beperkingen voor tabel `sensors`
--
ALTER TABLE `sensors`
  ADD CONSTRAINT `sensors_home_id_1b6b9891_fk_homes_id` FOREIGN KEY
    (`home_id`) REFERENCES `homes` (`id`);

--
-- Beperkingen voor tabel `sensors_tags`
--
ALTER TABLE `sensors_tags`
  ADD CONSTRAINT `sensors_tags_sensor_id_ec6d5d5a_fk_sensors_id`
    FOREIGN KEY (`sensor_id`) REFERENCES `sensors` (`id`),
  ADD CONSTRAINT `sensors_tags_tag_id_7ab85af9_fk_tags_id` FOREIGN
    KEY (`tag_id`) REFERENCES `tags` (`id`);

--
-- Beperkingen voor tabel `users_homes`
--
ALTER TABLE `users_homes`
  ADD CONSTRAINT `users_homes_home_id_6d8097a2_fk_homes_id` FOREIGN
    KEY (`home_id`) REFERENCES `homes` (`id`),
  ADD CONSTRAINT `users_homes_user_id_c35d6efe_fk_auth_user_id`
    FOREIGN KEY (`user_id`) REFERENCES `auth_user` (`id`);

--
-- Beperkingen voor tabel `yearly_data`
--
ALTER TABLE `yearly_data`
  ADD CONSTRAINT `yearly_data_sensor_id_349bf5e9_fk_sensors_id`
    FOREIGN KEY (`sensor_id`) REFERENCES `sensors` (`id`);
```