MDPI

Article

Enabling End-User Development in Smart Homes: A Machine Learning-Powered Digital Twin for Energy Efficient Management

Luca Cotti, Davide Guizzardi, Barbara Rita Barricelli and Daniela Fogli *

Department of Information Engineering, University of Brescia, Via Branze 38, 25123 Brescia, Italy; davide.guizzardi@unibs.it (D.G.); barbara.barricelli@unibs.it (B.R.B.)
* Correspondence: daniela.fogli@unibs.it

Abstract: End-User Development has been proposed over the years to allow end users to control and manage their Internet of Things-based environments, such as smart homes. With End-User Development, end users are able to create trigger-action rules or routines to tailor the behavior of their smart homes. However, the scientific research proposed to date does not encompass methods that evaluate the suitability of user-created routines in terms of energy consumption. This paper proposes using Machine Learning to build a Digital Twin of a smart home that can predict the energy consumption of smart appliances. The Digital Twin will allow end users to simulate possible scenarios related to the creation of routines. Simulations will be used to assess the effects of the activation of appliances involved in the routines under creation and possibly modify them to save energy consumption according to the Digital Twin's suggestions.

Keywords: End-User Development; digital twin; internet of things; routine; smart home

check for updates

Citation: Cotti, L.; Guizzardi, D.; Barricelli, B.R.; Fogli, D. Enabling End-User Development in Smart Homes: A Machine Learning-Powered Digital Twin for Energy Efficient Management. *Future Internet* **2024**, *16*, 208. https://doi.org/10.3390/fi16060208

Academic Editors: Mario Casillo, Marco Lombardi and Domenico Santaniello

Received: 9 May 2024
Revised: 11 June 2024
Accepted: 12 June 2024
Published: 14 June 2024

1. Introduction

The Internet of Things is a paradigm that was originally proposed at the Massachusetts Institute of Technology [1]. The underlying idea is the presence of a variety of things and devices (e.g., sensors, actuators, smartphones, etc.) that are able to communicate over the internet and cooperate to achieve common goals [2]. This concept has been developed over the years as an interdisciplinary field and has been applied to a variety of contexts, like smart homes, smart cities, and smart factories [3].

In line with the definition provided in [4], designing an Internet of Things application with a sociotechnical approach [5] means creating an Internet of Things ecosystem that encompasses smart sensors and devices, social networks, software applications, recommender systems, and users. The present paper focuses on the design of Internet of Things ecosystems in smart homes. The availability of inexpensive low-power sensors, radio frequency chips, and embedded processors allows smart homes to be fitted with a large number of networked sensors that collaboratively process and make inferences from the acquired data on the state of the home as well as the activities and behaviors of its occupants [6]. Computers or devices with computing power, such as micro-controllers, analyze these data to recognize the actions of residents or the occurrence of events. They then react to these actions and events by operating on connected devices available in the home [7]. Examples of such smart behaviors include switching the lights on when a person enters a room or detecting if an elderly resident has fallen in the corridor. Users are part of the Internet of Things ecosystem in that they not only trigger smart home behaviors but can actively define and modify such behaviors. Indeed, smart devices embed electronic components that support direct interaction and remote control [8,9]. In this way, users can directly, remotely, or automatically adjust the lights, televisions, thermostats, shutters, locks, and other smart objects using domotic dashboards, mobile applications, or virtual assistants (e.g., Google Home, Amazon Alexa, and Apple Siri).

Defining smart home behaviors means creating *routines* (namely, trigger-action rules) that are automatically executed when some event or condition occurs. The user is guided through the steps required to set up the name of the routine, the trigger, and the actions. A given number of parameters are requested for the trigger and each action of a routine. For instance, if the user would like to create a routine that starts at 7:00 a.m. every working day, time and day parameters must be set; and, if the user would like the light of the bathroom to be switched on with brightness 40%, both these parameters ("on" and "40%") must be set for the light-switching action. IFTTT (*IF This Then That*) [10] is one of the most widely known examples of this kind: it allows the user to combine a huge number of services, apps, and devices in rules with a single trigger and a single action. Scientific research (e.g., refs. [11–14]) has addressed the problem of allowing the creation of more complex rules through easy-to-use graphical user interfaces, augmented reality [15], voice-based interfaces [16–18], and multi-modal interfaces combining visual, voice, and touch interaction [19].

This approach can address end users' diverse and changing needs, where the term *end user* refers to people who are not experts in programming or computer technologies. Therefore, in the case at hand, smart home inhabitants are end users who require End-User Development techniques to facilitate the creation of smart home behaviors tailored to their needs. Specifically, End-User Development has been defined as "the set of methods, techniques, tools, and sociotechnical environments that allow end users to act as professionals in those information and communication technology-related domains in which they are not professionals, by creating, modifying, extending and testing digital artifacts without requiring knowledge in traditional software engineering techniques" [20].

In the current era of the Internet of Things, the creation of automatic behaviors like routines may have significant potential to develop effective energy management systems for homes. This will contribute to achieving the United Nations' sustainability development goals (https://sdgs.un.org/goals accessed on 1 May 2024) by reducing energy consumption and environmental impact while lowering residents' energy bills. The Internet of Things can be combined with Machine Learning algorithms to obtain a so-called Digital Twin, a powerful tool for modeling and optimizing complex systems in various domains, such as manufacturing, healthcare, and energy saving [21–23]. Digital Twins can effectively be used in buildings to enable the more efficient and intelligent management of energy, security, and comfort [24–26] or to improve the healthcare of older people through monitoring and prediction [27,28].

Monitoring and prediction can be applied to the creation of routines in smart homes so that the Digital Twin can notify users when a conflict between routines occurs. For example, the user might have defined a routine to switch on the light of the bedroom lampshade every day at 10:30 p.m., and then another routine to switch off all the lights at 10:30 p.m. without realizing that the second routine will also turn off the lampshade. Obviously, more complex situations may occur when several routines are defined to work synergistically in the same smart home. Some initial work in the Human–Computer Interaction field has been developed to study how to specify automation rules and permit their debugging, including conflict detection [29,30].

However, in the literature on Digital Twins, home inhabitants usually play a passive role: they contribute to generating data but do not intervene in actively modifying smart home behaviors by creating routines. Moreover, there is a lack of scientific work addressing the issues related to the interaction between users and Digital Twins [31]. This paper explores the design and implementation of a Digital Twin for a green smart home to forecast and optimize the energy consumption of various appliances. Particularly, the Digital Twin can evaluate in advance the routines that home inhabitants would like to create and suggest possible modifications that enhance energy consumption or avoid energy supply interruption. The novel contribution of this research is applying a sociotechnical approach [5] to designing Digital Twins for smart homes, focusing both on humans through End-User Development techniques and on technical issues related to appliance automation with Machine Learning algorithms. The final goal is to better support users in controlling their smart home and preventing undesired situations.

More precisely, this paper's contributions are the following:

- creation of the Digital Twin of a hypothetical smart home based on energy consumption open data;
- a method for mining appliances' operation modes with unsupervised machine learning;
- implementation of a Digital Twin web service for energy consumption prediction;
- design of the Digital Twin interface for energy consumption simulation of routines created through End-User Development.

The paper is organized as follows: Section 2 presents the hypothetical smart home and the Machine Learning approach adopted in the Digital Twin to recognize the operation modes of appliances; Section 3 describes the developed features to evaluate user-created routines and the Digital Twin operation through a proof-of-concept web-based application; Section 4 presents the most important findings, design implications, and open issues of the research.

2. Materials and Methods

This section presents a hypothetical smart home to develop and validate the Digital Twin functionalities; it has been created considering publicly available datasets. Then, the machine learning approach to recognizing appliance operation modes is illustrated. The hypothetical smart home and the machine learning approach are the building blocks of the developed Digital Twin, which can inform the user of possible conflicts between routines and provide suggestions on how to solve them.

2.1. A Hypothetical Smart Home

A hypothetical home was designed as a starting point for creating the Digital Twin. Different types of appliances were chosen to be introduced into the home based on their commonality in European households and their availability in the GREen ENergy Dataset (GREEND) [32] (https://www.andreatonello.com/greend-energy-metering-data-set/ accessed on 15 May 2024) and the UK Domestic Appliance-Level Electricity (UK-DALE) [33] (https://jack-kelly.com/data/ accessed on 15 May 2024) datasets. These two datasets were selected among a variety of open-source datasets made available as additional materials of scientific papers because they provide a large number of appliances, and minimize the geographic variations in appliance usage patterns. Our hypothetical home is composed of 6 rooms, each hosting several smart appliances. The details are presented in Table 1.

Table 1. In this table, for each room of the hypothetical home, the relative smart appliances are listed.

Room	Appliances
Corridor	Lamp
Living room	Air conditioning split, lamp, radio, television
Kitchen	Dishwasher, fridge with a combined freezer, lamp, microwave
Bedroom #1	Air conditioning split, desktop computer, lamp
Bedroom #2	Air conditioning split, lamp, television
Bathroom	Boiler, lamp, washing machine

For each appliance type, a specific appliance available in one of the buildings included in the GREEND or UK-DALE datasets was chosen as a representative sample. Table 2 shows the details of the selected appliances, including the source dataset, the building, the manufacturer, and the model. The preferred sources were buildings 3 and 5 of the GREEND dataset, as they had a large number of appliance data and reasonable consumption values. Other buildings in GREEND were discarded, as they either contained insufficient usage data or the consumption data were highly unrealistic, indicating measurement errors. The appliance types that were either unavailable or unsuitable in GREEND were instead chosen from the UK-DALE dataset. Because no information about individual air conditioning splits was available, the whole-home air conditioning consumption was used instead, which corresponded to the three splits being controlled collectively.

Table 2. Details of the selected appliances. Empty cells indicate that the information was not provided.

Appliance Type	Dataset	Building	Manufacturer	Model
Radio	GREEND	3	Denon	DRA-275RD
Dishwasher	GREEND	5	Whirlpool	ADG 555 IX
Fridge w/freezer	GREEND	5	Siemens	
Television	GREEND	5	Samsung	LE32C650
Microwave	GREEND	3	Whirlpool	AMW 494/IX
Lamp	UK-DALE	1		
Washing machine	GREEND	3	Zanussi	F1215
Desktop	GREEND	5	Dimotion	QuietONE Q2
Boiler	UK-DALE	4		
Air conditioning	UK-DALE	5		

A limitation of the GREEND and UK-DALE datasets is that they do not record the operation mode of appliances at a given time. This information is required to assign a specific consumption value to each operation mode and to estimate its duration. A possible solution to this problem is identifying the operation mode of appliances from the power consumption data using unsupervised learning methods, such as applying clustering techniques to the raw energy values. However, this approach can work only for simple appliances, such as lamps or televisions, but not for complex ones, such as fridges or washing machines, which may have different *operation cycles*. For example, a washing machine might draw no power for a short period while the clothes soak, and then use a lot of power for a spin cycle. Both of these cycles are part of the same operation mode, but there is no way to recognize this when clustering the raw data. Furthermore, methods based on raw data are also susceptible to noisy points and outliers; thus, they often lead to poor clustering results [34]. For this reason, the first step of our work was to identify a method for operation mode recognition. This is illustrated in the following subsection.

2.2. Recognition of Appliance Operation Modes

The recognition of the operation modes of appliances is necessary to estimate the consumption value and the duration of the appliances' modes, thus keeping track of the status of smart devices at any given time. To address this problem, an approach similar to the one presented by Castangia et al. [34] is used. Specifically, unsupervised deep learning clustering techniques are applied to a learned, latent-state representation of the raw data to produce the list of appliances' modes of operation. The high-level procedure is divided into the 4 steps displayed in Figure 1. Firstly, segmentation methods are applied to the raw data to extract the power signatures of the device (called *appliance activations*). Subsequently, the extracted signatures undergo standardization before becoming the input of an autoencoder model. This model learns to reconstruct the operation cycles and encode them into a latent representation. Next, a K-means clustering algorithm is applied to the latent representation, being responsible for grouping them into potential appliance modes. Finally, the clusters are manually assigned human-intelligible labels.

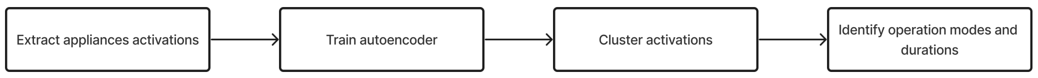

Extract appliances activations
Train autoencoder
Cluster activations
Identify operation modes and durations

Figure 1. High-level scheme of the procedure for operation mode extraction.

For the extraction of appliances' activations *the Non-Intrusive Load Monitoring Toolkit* (NILMTK) (https://github.com/nilmtk/nilmtk accessed on 1 May 2024) has been used. This Python toolkit, designed by Batra et al. [35], provides a way to use and compare different energy disaggregation algorithms, producing the corresponding power signature. The optimal values for NILMTK's parameters have been selected empirically for each appliance by preferring combinations that produce a balance between the number of activations and their mean length.

The extracted data have been then used to train an autoencoder network. This particular type of unsupervised method can learn to compress the inputs into a low-dimensional representation, called a *latent state* [36]. Figure 2 shows the adopted architecture. The *Encoder* is the part responsible for the conversion of the initial input into a suitable latent state, while the *Decoder* takes the latent states and converts them back to the original input. The Long Short-Term Memory (LSTM) included in the Encoder and Decoder is a recurrent neural network that can learn long-term temporal dependencies in sequential data [37]. In the case at hand, it is useful to learn the dependencies among the values of the activations of an appliance. The autoencoder is implemented using Tensorflow (https://www.tensorflow.org accessed on 15 May 2024), in its version 2.15.0, along with the Keras (https://keras.io accessed on 15 May 2024) deep learning library, and the code is executed inside a *Google Colab* notebook environment.

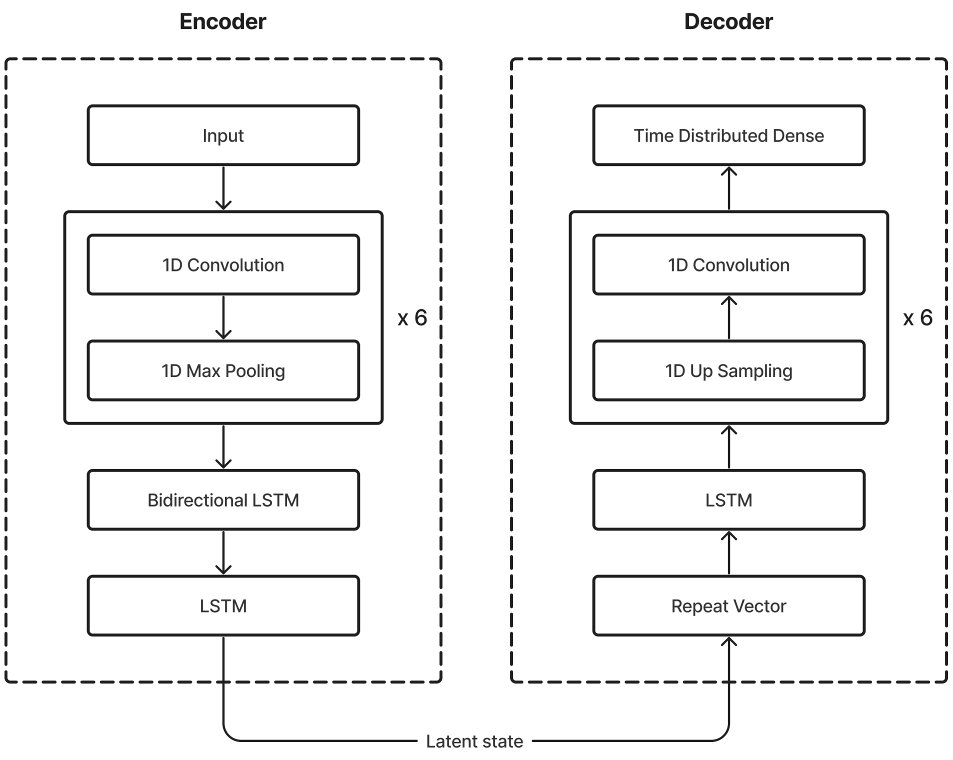


Figure 2. Architecture of the implemented autoencoder.

Following the training of the autoencoder network, the subsequent step involves applying a clustering procedure to partition the latent states produced by the Encoder. A detailed scheme of the procedure can be found in Figure 3. Firstly, the activations of each appliance are standardized, thus helping with the consequent training process by enhancing its convergence speed and stability and by setting the same scale for each activation. The standardized activation is then fed to the Encoder in order to produce the latent states. Next, the K-means algorithm is used to cluster the latent states into K different groups. The optimal value of K for each appliance category has been obtained by running a grid search procedure. The grid search's solutions have been evaluated using the *silhouette score* proposed in [38], choosing the one with the highest value.

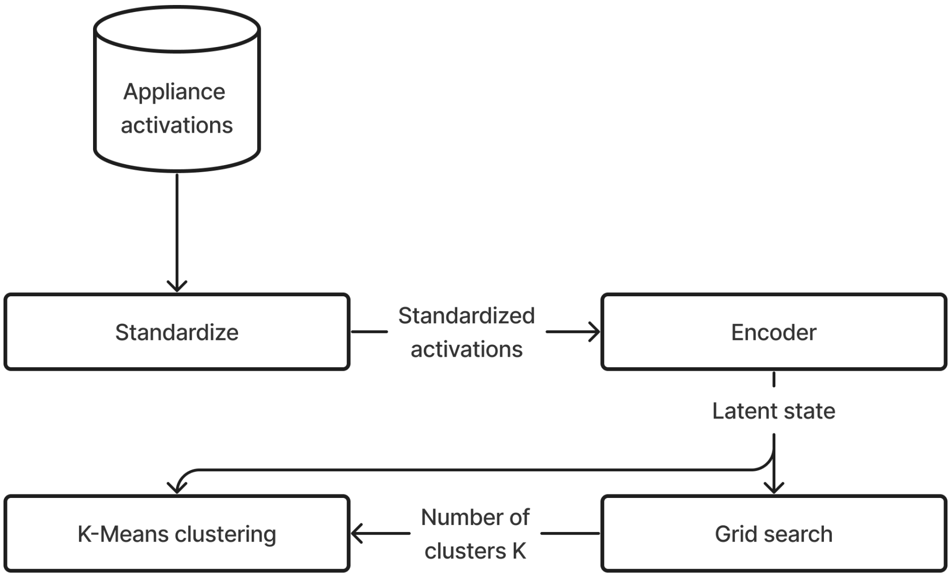


Figure 3. Procedure for clustering the operation modes of appliances.

The last step of the approach requires identifying the appliances' modes of operation, given the cluster labels, thus computing their mean energy consumption and duration. This procedure is presented in Figure 4. All the appliance activations are assigned to the closest cluster centroid. Once the assignment is done, it is possible to obtain the estimated energy consumption of each cluster by computing the average energy consumption of all the activations belonging to that cluster and by rounding it to the nearest integer. The same can be done for the activations' duration, producing a table that specifies the average consumption and duration for each cluster representing one mode of operation. Finally, names are assigned to each mode of operation by comparing the extracted data with the information provided by the appliances' manuals. Table 3 summarizes the results of the identification procedure for all the appliances included in the hypothetical smart home.

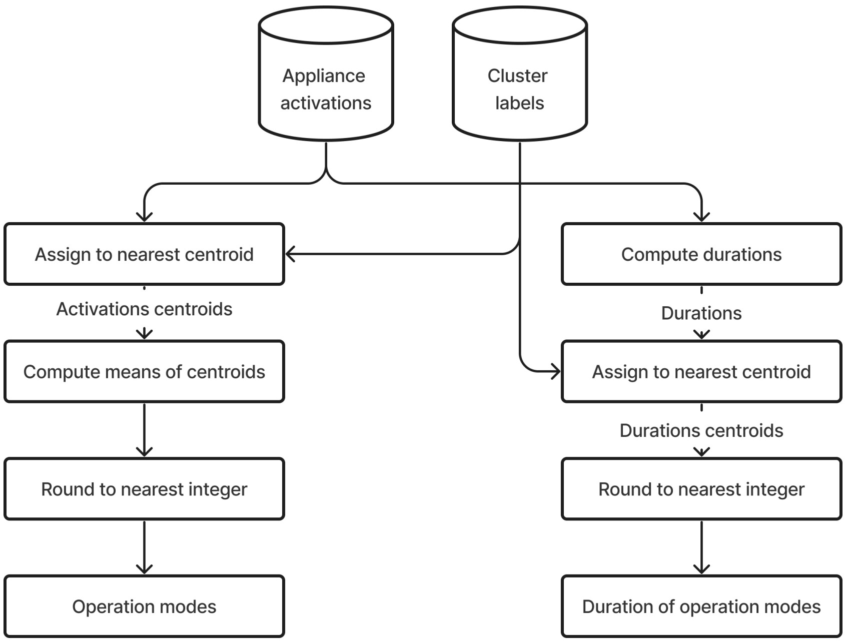


Figure 4. Operation modes identification procedure.

Table 3. Results of operation modes identification. The *off* mode is not reported, as it is assumed to have 0 Watt consumption and indeterminate duration for all appliances.

Appliance Type	Mode Name	Power (W)	Duration (s)
Radio	Stand-by	1	∞
	On	37	4435
Dishwasher	Intensive	1400	10,380
	Daily	810	10,260
	Eco	700	6660
	Express	350	1680
	Delicate	800	7080
	Pre-wash	10	660
Fridge w/freezer	On	208	6304
Television	Stand-by	18	2514
	On	110	1833
Microwave	Stand-by	5	∞
	Microwave	750	300
	Crisp	500	300
	Grill	500	300
	Grill + microwave	500	300
	Steam	350	300
	Jet defrost	160	300
Lamp	On	19	587
Washing machine	Cotton 90°	2000	8700
	Cotton 60 ECO	950	7680
	Cotton 60°	1200	7200
	Cotton 30°	550	6900
	Synthetic 30°	900	5400
	Delicate 30°	500	3600
	Wool	450	3300
Desktop	Stand-by	4	∞
	On	41	12,412
Boiler	Holiday	33	300
	Comfort	57	525
	Auto	101	2187
Air conditioning	Cool	336	3337
	Heat	656	30,176

3. Results

In this section, we present the routine simulation feature used by the Digital Twin to identify conflicts and provide users with recommendations and suggestions. Then, the user–Digital Twin interaction is illustrated by means of a proof-of-concept web application.

3.1. Simulation of New Routines

Routine evaluation and simulation features are implemented through a REST API. The Python *FastAPI* (https://fastapi.tiangolo.com/ accessed on 1 May 2024) library is the selected tool for the development.

Table 4 shows all the available API's endpoints. Four groups of functionalities can be identified:

- */consumption*. The endpoints belonging to this group provide insight into the appliances' energy consumption. It is possible to ask for the energy consumption of a specific time range and/or a specific appliance;

- */appliance*. This group provides two methods: one to obtain data from all the appliances of the house and one to obtain information on a specific appliance;
- */routine*. There are two endpoints for this group as well. One lists all the user-created routines saved in the system, and the other provides the user with data on a specific routine;
- */simulate*. The endpoints of this group are the ones responsible for the simulation of routine addition. The procedure will be discussed in more detail in the following section.

Table 4. Endpoints of the Digital Twin's REST API.

Endpoint	Method	Path Parameters	Query Parameters	Body Parameters	Description
/	GET				Displays the API documentation using Swagger UI.
/consumption	GET	Date and time			Returns a list of energy consumption values of all appliances at the given date and time.
/consumption	GET	Appliance identifier, date, and time			Returns the energy consumption value of the given appliance at the given date and time.
/consumption /total	GET		List of dates and times		Returns the list of total energy consumption of all appliances at the given dates and times.
/consumption /total	GET	Date and time			Returns the total energy consumption of all appliances at the given date and time.
/appliance	GET				Returns the list of all appliances.
/appliance	GET	Appliance identifier			Returns a specific appliance.
/routine	GET				Returns the list of all routines
/routine	GET	Routine identifier			Returns a specific routine
/simulate	POST			Routine to simulate	Simulates the addition of a routine and returns a list of recommendations and errors (if any).
/simulate /consumption	POST	Date and time		Routine to simulate	Simulates the addition of a routine and returns a list of energy consumption values of all appliances at the given date and time.
/simulate /consumption	POST	Appliance identifier, date, and time		Routine to simulate	Simulates the addition of a routine and returns the energy consumption value of the given appliance at the given date and time.
/simulate /consumption /total	POST	Date and time		Routine to simulate	Simulates the addition of a routine and returns the list of total energy consumption of all appliances at the given date and time.
/simulate /consumption /total	POST		List of dates and times	Routine to simulate	Simulates the addition of a routine and returns the total energy consumption of all appliances at the given dates and times.

To simulate the addition to the Digital Twin of a new routine created at runtime by the user, it is necessary to have a data structure that represents the state of the appliances throughout the day. Therefore, the *State Matrix* has been designed. Each row of the matrix represents one specific minute of the day, while each column refers to a household appliance. Each cell contains a unique identifier corresponding to the operation mode in which the appliance is configured for that particular minute.

Before adding a new routine, the system checks if any conflict will arise by comparing the current State Matrix with a hypothetical one, in which the routine has already been saved. Specifically, two types of conflicts are taken into account:

C1. *An appliance has conflicting operation modes in the same time interval.* This conflict emerges when two routines want to use the same appliance simultaneously but with two different modes. Such a scenario happens when a cell in the original matrix, holding an operation mode other than "stand-by", is replaced by a distinct value in the new matrix. Figure 5 shows an example in which this type of conflict occurs and is correctly identified;

C2. *The power consumption of the appliances in a time interval exceed a maximum limit.* In this case, the user asks their smart devices to use a quantity of energy that is above the maximum load capacity of the house (commonly set to 3 kW for domestic use in Italian households; the Italian case is representative of those countries that share an issue with constrained energy availability). Such a case is identified by computing the sum of the consumption values for each row of the matrix and comparing it with the maximum allowed value.

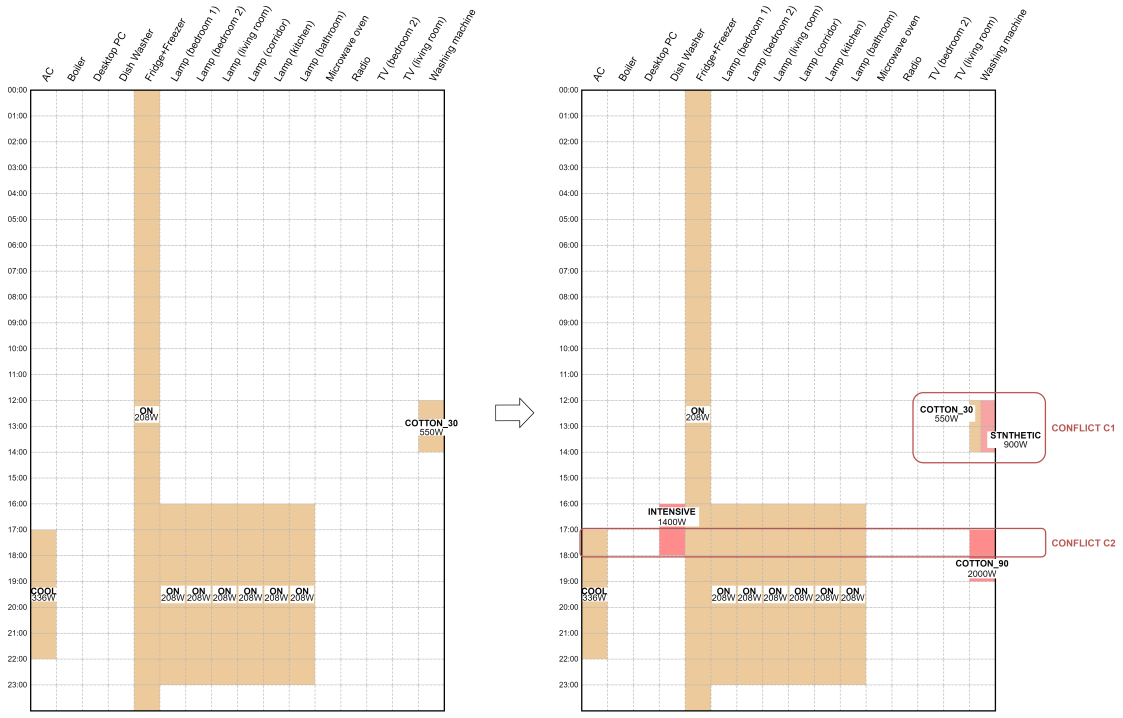


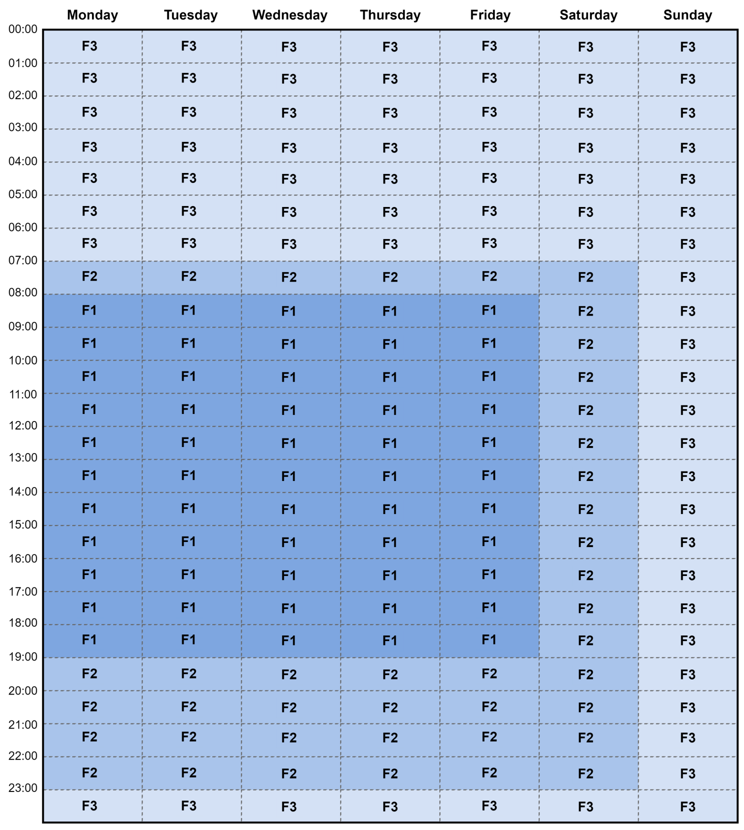
Figure 5. Example of routine addition simulation and consequent conflict identification. On the left is the current State Matrix, and on the right is the simulated one. A conflict of type C1 emerges due to the washing machine being set to *"Wool"* and *"Cotton"* at the same time. One C2-type conflict emerges due to the overall consumption of the house exceeding 3 kW between 17:00 and 18:00. AC stands for air conditioning, PC for personal computer, and TV for television.

In addition to identifying conflicts, the developed API can also guide users in optimizing their routines by providing them with recommendations. Two families of recommendations can be produced:

R1. *Change routine's start time.* The system can suggest to users a better start time for their routines, according to the energy cost of running them at different times during the day. This type of recommendation is based on the peak hours and off-peak hours defined by the user during the Digital Twin configuration, which depends on the electricity supplier. Figure 6 shows an example of such a configuration;

R2. *Disable routine.* When one of the conflict scenarios described above takes place, the system will suggest to the user to disable one of the responsible routines.

The high-level scheme of the simulation procedure is displayed in Figure 7. Here, the API will return either recommendations or data on routine consumption depending on the endpoint: `/simulate` will produce the former, while `/simulate/consumption` will provide the latter.



	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
00:00	F3	F3	F3	F3	F3	F3	F3
01:00	F3	F3	F3	F3	F3	F3	F3
02:00	F3	F3	F3	F3	F3	F3	F3
03:00	F3	F3	F3	F3	F3	F3	F3
04:00	F3	F3	F3	F3	F3	F3	F3
05:00	F3	F3	F3	F3	F3	F3	F3
06:00	F3	F3	F3	F3	F3	F3	F3
07:00	F2	F2	F2	F2	F2	F2	F3
08:00	F1	F1	F1	F1	F1	F2	F3
09:00	F1	F1	F1	F1	F1	F2	F3
10:00	F1	F1	F1	F1	F1	F2	F3
11:00	F1	F1	F1	F1	F1	F2	F3
12:00	F1	F1	F1	F1	F1	F2	F3
13:00	F1	F1	F1	F1	F1	F2	F3
14:00	F1	F1	F1	F1	F1	F2	F3
15:00	F1	F1	F1	F1	F1	F2	F3
16:00	F1	F1	F1	F1	F1	F2	F3
17:00	F1	F1	F1	F1	F1	F2	F3
18:00	F1	F1	F1	F1	F1	F2	F3
19:00	F2	F2	F2	F2	F2	F2	F3
20:00	F2	F2	F2	F2	F2	F2	F3
21:00	F2	F2	F2	F2	F2	F2	F3
22:00	F2	F2	F2	F2	F2	F2	F3
23:00	F3	F3	F3	F3	F3	F3	F3

Figure 6. Electricity tariffs applied in Italy for every day of the week. F1 is the peak rate, F2 is the mid-level, and F3 is off-peak.

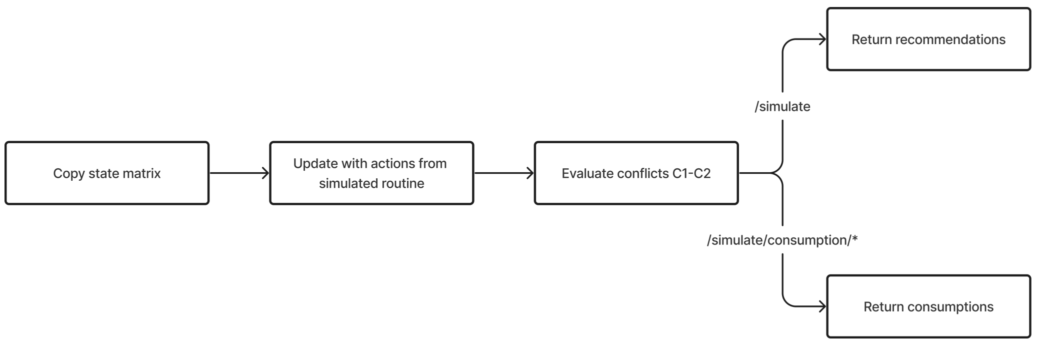


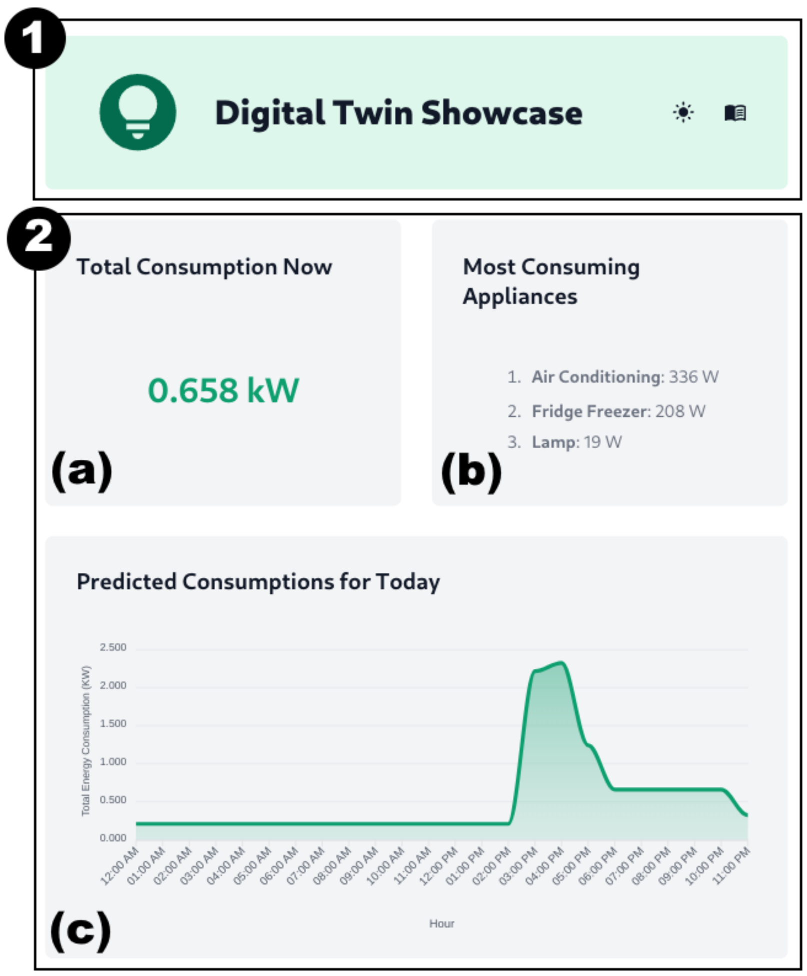
Figure 7. New routine addition simulation procedure.

3.2. Interaction with the Digital Twin

To demonstrate the features offered by the Digital Twin, a responsive Web-based application has been developed. It is not meant to be a fully functional interactive system but a first prototype to show the potential of our work. The application was built using the React (https://react.dev/ accessed on 15 May 2024) framework and has a one-page interface styled using the TailwindCSS (https://tailwindcss.com/ accessed on 15 May 2024) and the FlowBite (https://flowbite.com/ accessed on 15 May 2024) libraries.

The application page is organized into the following sections:

- **Header**: Contains the title and logo of the application, which is derived from Material Icons (https://fonts.google.com/icons accessed on 1 May 2024), and two buttons for switching between light and dark modes and to access the API documentation (see (1) in Figure 8).
- **Statistics**: Provides various items of information about the home's energy consumption (see (2) in Figure 8 and the corresponding letters used in the following list):
    a. The total instant power consumption of all appliances.
    b. The three appliances that consume the most power at the current time and their respective power consumption values.
    c. A chart that predicts the total power consumption of all appliances throughout the day based on historical data. The chart is implemented using the ApexCharts (https://apexcharts.com/ accessed on 15 May 2024) library and offers features such as zooming, panning, and tooltips.



1. **Air Conditioning**: 336 W
2. **Fridge Freezer**: 208 W
3. **Lamp**: 19 W

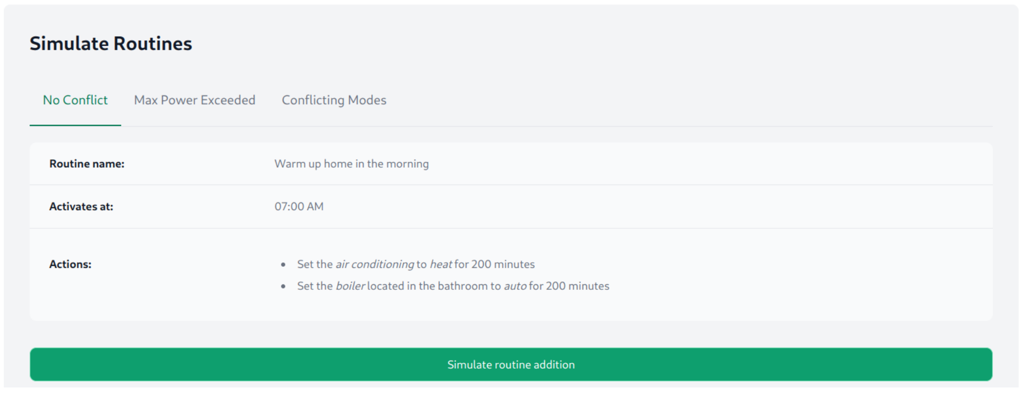
Figure 8. This is the top part of the application page made of the Header section (**1**) and the Statistics section (**2**), in which three elements are included: (**a**) total instant power consumption; (**b**) the three appliances that are using the most power; (**c**) a prediction chart of daily total power consumption.

- **Appliances**: Displays a table with the device name and type, the manufacturer and model, the location in the home, and the names of the supported operation modes. The table also automatically adds an appropriate icon for each appliance, taken from Material Icons.

- **Routines**: Displays the name, time of activation of the routines, whether the routine is enabled or not, and the list of actions of the routines.
- **Simulation**: Allows the user to simulate the effect of adding a new routine to the home and to detect any potential conflicts with the existing routines. In what follows, we will describe in detail how this last section works.

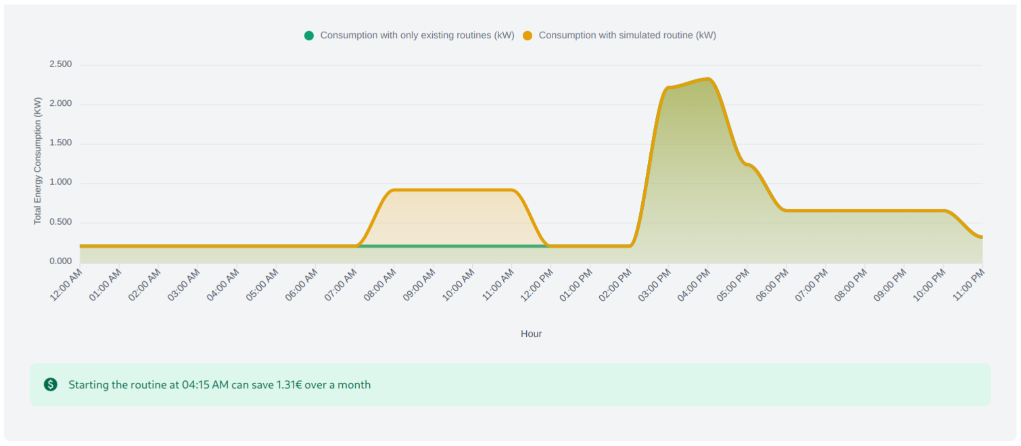
Given the demonstrative nature of this application, the application allows the user to choose only from three predefined routines, which are shown in three different tabs: *No conflict*, *Party time*, and *Heavy-duty wash*.

The "No Conflict" tab, shown in Figure 9, displays the details of the *Warm up home in the morning* routine, which activates the air conditioning and the boiler at 7:00 a.m. By clicking on the "Simulate routine addition" button, the user can send a `POST` request to the `/simulate` endpoint of the API and see the result of the simulation, as shown in Figure 10. The simulation indicates that the routine is accepted without any conflicts. A graph, which uses the data from the `/simulate/consumption/total` endpoint of the API, compares the power consumption with and without the simulated routine. A recommendation to change the start time of the routine to a more optimal one is also provided, along with the projected savings over 30 days.



Routine name:	Warm up home in the morning
Activates at:	07:00 AM
Actions:	• Set the *air conditioning* to *heat* for 200 minutes • Set the *boiler* located in the bathroom to *auto* for 200 minutes

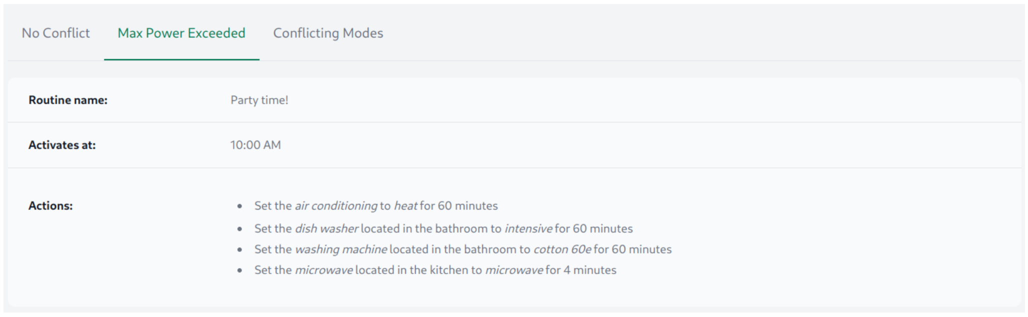
Figure 9. Details of the *No Conflict* routine.



Starting the routine at 04:15 AM can save 1.31€ over a month

Figure 10. Results of simulating the *Warm up home in the morning* routine.

The *Party time* routine, shown in Figure 11, is designed to test the scenario where the power consumption surpasses the maximum limit of 3 kW. It turns on the air conditioning, the dishwasher, the washing machine, and the microwave at 10:00 a.m. The simulation results are displayed in Figure 12. The simulation shows an error due to conflict C2 and suggests disabling either the simulated routine or an existing routine that consumes much power. No recommendation to change the start time of the routine is given because the routine would always exceed the maximum limit regardless of the time.

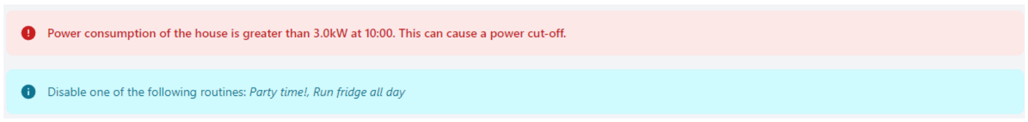


Routine name: Party time!

Activates at: 10:00 AM

Actions:
- Set the *air conditioning* to *heat* for 60 minutes
- Set the *dish washer* located in the bathroom to *intensive* for 60 minutes
- Set the *washing machine* located in the bathroom to *cotton 60e* for 60 minutes
- Set the *microwave* located in the kitchen to *microwave* for 4 minutes

Figure 11. Details of the *Party time* routine, which causes the power consumption to exceed the maximum limit.

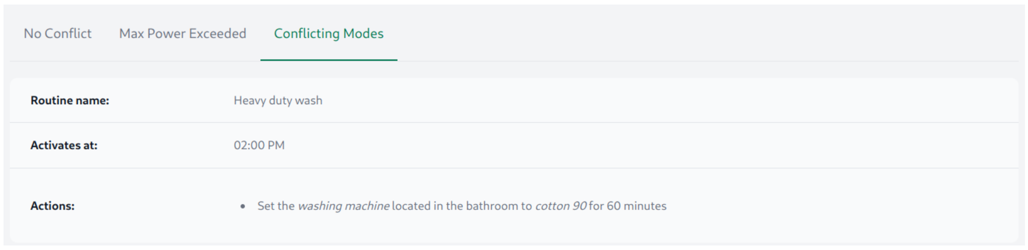


Power consumption of the house is greater than 3.0kW at 10:00. This can cause a power cut-off.

Disable one of the following routines: *Party time!, Run fridge all day*

Figure 12. Results of simulating the *Party time* routine.

The *Heavy-duty wash* routine, shown in Figure 13, is designed to show a conflict with an existing routine. It sets the washing machine to the *cotton 90* mode at 2:00 p.m. However, another routine already sets the washing machine to the *cotton 30* mode at the same time. The simulation results are shown in Figure 14. The simulation reports an error due to conflict C1 and offers two recommendations to disable either of the conflicting routines. A recommendation to change the start time of the routine to a more suitable one is also given, along with the projected savings.

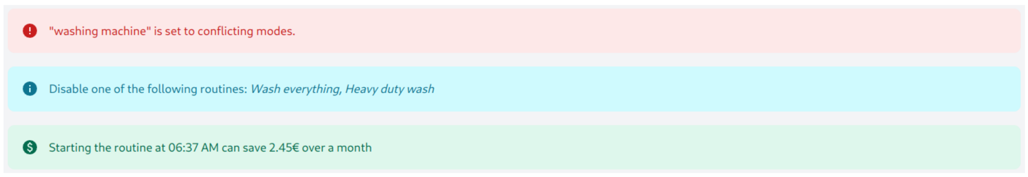


Routine name: Heavy duty wash

Activates at: 02:00 PM

Actions:
- Set the *washing machine* located in the bathroom to *cotton 90* for 60 minutes

Figure 13. Details of the *Heavy duty wash* routine, which causes a conflict with an existing routine.



"washing machine" is set to conflicting modes.

Disable one of the following routines: *Wash everything, Heavy duty wash*

Starting the routine at 06:37 AM can save 2.45€ over a month

Figure 14. Results of simulating the addition of the *Heavy duty wash* routine to the system.

4. Discussion and Conclusions

In this paper, we have shown how to integrate user-created routines involving different appliances' activations in the Digital Twin of a smart home able to monitor and simulate smart home behavior in terms of energy consumption. To this purpose, we designed and implemented an Machine Learning-based solution for the recognition of the operation modes of appliances available in a hypothetical home. Then, functionalities were developed to help users avoid undesired situations and choose more efficient routines or better combinations of them, thus eventually enhancing consumption awareness.

A sociotechnical approach was adopted to developing the Digital Twin. Indeed, the literature on Digital Twins often lacks consideration of the interaction with the user and the needs the user might satisfy with a Digital Twin. According to the classification of maturity levels for Digital Twins proposed in [39], Digital Twins at level 4 foresee users-in-the-loop for the effective and secure operation and control of the Digital Twin, but most Digital Twins discussed in the literature are at levels 0–2 of maturity (modeling and simulation), while only few of them start the integration with real-time data streams (level 3), due to the significant challenge of data gathering, filtering, and processing in real time [40]. Several Digital Twins have been proposed for building automation and smart home control, but the role of the end user in the modification of smart home behavior (and consequent Digital Twin behavior) has not been adequately explored. On the other hand, End-User Development methods to create routines for smart homes have been studied for many years and have been implemented recently in commercial virtual assistants. Integrating these End-User Development methods in a Digital Twin is a contribution towards achieving Digital Twins at level 4: that is, enabling two-way data integration and interaction [39].

Several open challenges will guide our future research in this field and further system development.

For the sake of the experimentation of the simulation modality, a set of routines has been built in the system, and data about appliances' status at different times of the day have been extracted from open-source datasets applying machine learning. In addition, the Digital Twin can only predict energy consumption for the current day. Future versions should enable the system to predict energy consumption for several days ahead, and also to view consumption history for the previous days. This would allow the system to provide more precise recommendations and better handle conflicts among routines. For example, if the user is planning to go on vacation for a week, the Digital Twin could advise to set the boiler to the *holiday* mode and to turn off the fridge.

Moreover, the Digital Twin should be able to handle the manual activation of appliances by end users by automatically detecting their state changes. Indeed, the Digital Twin could monitor the manual activations of appliances and their duration, and use this information to provide suitable recommendations to automate the activations or modify existing routines. For instance, if the user starts the computer daily at 9:00 a.m., the Digital Twin could propose to automate this activation. Suppose the computer stays on for 8 h on average. In that case, the Digital Twin could use this information to anticipate whether this would cause any conflict with existing routines and avoid them before they occur.

To carry out a long-term experimentation of the system operation and collect feedback from users about their experience, we plan as future work to install smart devices and the Digital Twin in a few sample homes. This will entail developing a physical twin similar to the hypothetical home presented in Section 2.1. The Digital Twin should also update the state of the devices and the energy consumption according to the changes in the physical world. Consequently, a quantitative and qualitative comparison with existing approaches to energy management could be performed.

The user interface of the End-User Development feature for routine creation has not been presented in this paper, since this topic has been investigated for many years by the authors, and a proposal exploiting multi-modal interaction with virtual assistants has been recently presented in [19]. The idea is to integrate a similar feature in the presented Digital Twin.

Author Contributions: Conceptualization, L.C., D.G., B.R.B. and D.F.; Methodology, D.G. and B.R.B.; Software, L.C.; Formal analysis, L.C.; Writing—original draft, L.C.; Writing—review and editing, D.G., B.R.B. and D.F.; Supervision, D.F. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Italian MUR PRIN 2022 PNRR Project P2022YR9B7, End-User Development of Automations for Explainable Green Smart Homes, funded by European Union—Next Generation EU.

Data Availability Statement: The raw data supporting the conclusions of this article and the source code of the Web-based application are available at this link: https://github.com/LucaCtt/digital_twin_future_internet accessed on 1 May 2024.

Acknowledgments: The authors wish to thank the anonymous reviewers who provided helpful comments and suggestions to improve the paper.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AC	Air Conditioning
API	Application Programming Interface
kW	Kilowatt
LSTM	Long Short-Term Memory
NILMTK	Non-Intrusive Load Monitoring Toolkit
PC	Personal Computer
REST	Representational State Transfer
TV	Television
GREEND	GREen ENergy Dataset
UK-DALE	UK Domestic Appliance-Level Electricity

References

1. Sarma, S.; Brock, D.L.; Ashton, K. The networked physical world. In *Auto-ID Center White Paper MIT-AUTOID-WH-001*; Massachusetts Institute of Technology: Cambridge, MA, USA, 2000; pp. 1–16.
2. Atzori, L.; Iera, A.; Morabito, G. The Internet of Things: A survey. *Comput. Netw.* **2010**, *54*, 2787–2805. [CrossRef]
3. Wang, F.; Hu, L.; Zhou, J.; Zhao, K. A Survey from the Perspective of Evolutionary Process in the Internet of Things. *Int. J. Distrib. Sens. Netw.* **2015**, *11*, 462752. [CrossRef]
4. Barricelli, B.R.; Valtolina, S. Designing for End-User Development in the Internet of Things. In Proceedings of the End-User Development: 5th International Symposium, IS-EUD 2015, Madrid, Spain, 26–29 May 2015; Díaz, P., Pipek, V., Ardito, C., Jensen, C., Aedo, I., Boden, A., Eds.; Springer International Publishing: Cham, Switzerland, 2015; pp. 9–24.
5. Baxter, G.D.; Sommerville, I. Socio-technical systems: From design methods to systems engineering. *Interact. Comput.* **2011**, *23*, 4–17. [CrossRef]
6. Ding, D.; Cooper, R.A.; Pasquina, P.F.; Fici-Pasquina, L. Sensor Technology for Smart Homes. *Maturitas* **2011**, *69*, 131–136. [CrossRef]
7. De Silva, L.C.; Morikawa, C.; Petra, I.M. State of the Art of Smart Homes. *Eng. Appl. Artif. Intell.* **2012**, *25*, 1313–1321. [CrossRef]
8. Kortuem, G.; Kawsar, F.; Sundramoorthy, V.; Fitton, D. Smart Objects as Building Blocks for the Internet of Things. *IEEE Internet Comput.* **2010**, *14*, 44–51. [CrossRef]
9. Wu, Y.; Pillan, M. From Respect to Change User Behaviour. Research on How to Design a next Generation of Smart Home Objects from User Experience and Interaction Design. *Des. J.* **2017**, *20*, S3884–S3898. [CrossRef]
10. Ur, B.; McManus, E.; Pak Yong Ho, M.; Littman, M.L. Practical Trigger-Action Programming in the Smart Home. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'14), New York, NY, USA, 26 April–1 May 2014; pp. 803–812. [CrossRef]
11. Desolda, G.; Ardito, C.; Matera, M. Empowering End Users to Customize Their Smart Environments: Model, Composition Paradigms, and Domain-Specific Tools. *ACM Trans. Comput.-Hum. Interact.* **2017**, *24*, 3057859. [CrossRef]
12. Ghiani, G.; Manca, M.; Paternò, F.; Santoro, C. Personalization of Context-Dependent Applications through Trigger-Action Rules. *ACM Trans. Comput.-Hum. Interact.* **2017**, *24*, 3057861. [CrossRef]
13. Paternò, F.; Santoro, C. End-User Development for personalizing applications, things, and robots. *Int. J.-Hum.-Comput. Stud.* **2019**, *131*, 120–130. [CrossRef]
14. Manca, M.; Paternò, F.; Santoro, C. Remote monitoring of end-user created automations in field trials. *J. Ambient. Intell. Humaniz. Comput.* **2022**, *13*, 5669–5697. [CrossRef]
15. Mattioli, A.; Paternò, F. A Mobile Augmented Reality App for Creating, Controlling, Recommending Automations in Smart Homes. *Proc. ACM Hum.-Comput. Interact.* **2023**, *7*, 3604242. [CrossRef]
16. De Russis, L.; Monge Roffarello, A.; Borsarelli, C. Towards Vocally-Composed Personalization Rules in the IoT. In Proceedings of the EMPATHY: 2nd International Workshop on Empowering People in Dealing with Internet of Things Ecosystems, Aachen, Germany, 17 July 2021; pp. 1–5.
17. Gallo, S.; Paterno, F. A Conversational Agent for Creating Flexible Daily Automation. In Proceedings of the 2022 International Conference on Advanced Visual Interfaces (AVI 2022), New York, NY, USA, 6–10 June 2022. [CrossRef]
18. Lago, A.S.; Dias, J.P.; Ferreira, H.S. Managing non-trivial internet-of-things systems with conversational assistants: A prototype and a feasibility experiment. *J. Comput. Sci.* **2021**, *51*, 1–12. [CrossRef]
19. Barricelli, B.R.; Bondioli, A.; Fogli, D.; Iemmolo, L.; Locoro, A. Creating Routines for IoT Ecosystems through Conversation with Smart Speakers. *Int. J. Hum.-Comput. Interact.* **2023**, 1–19. [CrossRef]

20. Barricelli, B.R.; Cassano, F.; Fogli, D.; Piccinno, A. End-User Development, End-User Programming and End-User Software Engineering: A Systematic Mapping Study. *J. Syst. Softw.* **2019**, *149*, 101–137. [CrossRef]
21. Negri, E.; Fumagalli, L.; Macchi, M. A Review of the Roles of Digital Twin in CPS-based Production Systems. *Procedia Manuf.* **2017**, *11*, 939–948. [CrossRef]
22. Barricelli, B.R.; Casiraghi, E.; Fogli, D. A Survey on Digital Twin: Definitions, Characteristics, Applications, and Design Implications. *IEEE Access* **2019**, *7*, 167653–167671. [CrossRef]
23. Hermann, M.; Pentek, T.; Otto, B. Design Principles for Industrie 4.0 Scenarios. In Proceedings of the 2016 49th Hawaii International Conference on System Sciences (HICSS), Koloa, HI, USA, 5–8 January 2016; pp. 3928–3937. [CrossRef]
24. Tagliabue, L.C.; Cecconi, F.R.; Maltese, S.; Rinaldi, S.; Ciribini, A.L.C.; Flammini, A. Leveraging Digital Twin for Sustainability Assessment of an Educational Building. *Sustainability* **2021**, *13*, 480. [CrossRef]
25. Yang, B.; Lv, Z.; Wang, F. Digital Twins for Intelligent Green Buildings. *Buildings* **2022**, *12*, 856. [CrossRef]
26. Casillo, M.; Cecere, L.; Colace, F.; Lorusso, A.; Marongiu, F.; Santaniello, D. An Internet of Things Approach to Support a High-Tech House. In Proceedings of the 2023 7th IEEE Congress on Information Science and Technology (CiSt), Essaouira, Morocco,16–22 December 2023; pp. 323–328. [CrossRef]
27. Chen, J.; Wang, W.; Fang, B.; Liu, Y.; Yu, K.; Leung, V.C.M.; Hu, X. Digital Twin Empowered Wireless Healthcare Monitoring for Smart Home. *IEEE J. Sel. Areas Commun.* **2023**, *41*, 3662–3676. [CrossRef]
28. Shoukat, M.U.; Yan, L.; Zhang, J.; Cheng, Y.; Raza, M.U.; Niaz, A. Smart home for enhanced healthcare: Exploring human machine interface oriented digital twin model. *Multimed. Tools Appl.* **2024**, *83*, 31297–31315. [CrossRef]
29. Corno, F.; De Russis, L.; Monge Roffarello, A. Empowering End Users in Debugging Trigger-Action Rules. In Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI'19), New York, NY, USA, 4–23 December 2019; pp. 1–13. [CrossRef]
30. Manca, M.; Paternò, F.; Santoro, C.; Corcella, L. Supporting end-user debugging of trigger-action rules for IoT applications. *Int. J.-Hum.-Comput. Stud.* **2019**, *123*, 56–69. [CrossRef]
31. Barricelli, B.R.; Fogli, D. Digital Twins in Human-Computer Interaction: A Systematic Review. *Int. J.-Hum.-Comput. Interact.* **2024**, *40*, 79–97. [CrossRef]
32. Monacchi, A.; Egarter, D.; Elmenreich, W.; D'Alessandro, S.; Tonello, a.m. GREEND: An Energy Consumption Dataset of Households in Italy and Austria. In Proceedings of the 2014 IEEE International Conference on Smart Grid Communications (SmartGridComm), Venice, Italy, 3–6 November 2014; pp. 511–516. [CrossRef]
33. Kelly, J.; Knottenbelt, W. The UK-DALE Dataset, Domestic Appliance-Level Electricity Demand and Whole-House Demand from Five UK Homes. *Sci. Data* **2015**, *2*, 150007. [CrossRef] [PubMed]
34. Castangia, M.; Barletta, N.; Camarda, C.; Quer, S.; Macii, E.; Patti, E. Clustering Appliance Operation Modes With Unsupervised Deep Learning Techniques. *IEEE Trans. Ind. Inform.* **2023**, *19*, 8196–8204. [CrossRef]
35. Batra, N.; Kelly, J.; Parson, O.; Dutta, H.; Knottenbelt, W.; Rogers, A.; Singh, A.; Srivastava, M. NILMTK: An open source toolkit for non-intrusive load monitoring. In Proceedings of the 5th International Conference on Future Energy Systems (e-Energy'14), Cambridge, UK, 11–13 June 2014. [CrossRef]
36. Hinton, G.E.; Salakhutdinov, R.R. Reducing the Dimensionality of Data with Neural Networks. *Science* **2006**, *313*, 504–507. [CrossRef] [PubMed]
37. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
38. Rousseeuw, P.J. Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis. *J. Comput. Appl. Math.* **1987**, *20*, 53–65. [CrossRef]
39. Evans, S.; Savian, C.; Burns, A.; Cooper, C. *Digital Twins for the Built Environment: An Introduction to the Opportunities, Benefits, Challenges and Risks*; Technical Report; The Institution of Engineering and Technology (IET): Stevenage, UK, 2019.
40. Botín-Sanabria, D.M.; Mihaita, A.S.; Peimbert-García, R.E.; Ramírez-Moreno, M.A.; Ramírez-Mendoza, R.A.; Lozoya-Santos, J.d.J. Digital Twin Technology Challenges and Applications: A Comprehensive Review. *Remote. Sens.* **2022**, *14*, 1335. [CrossRef]