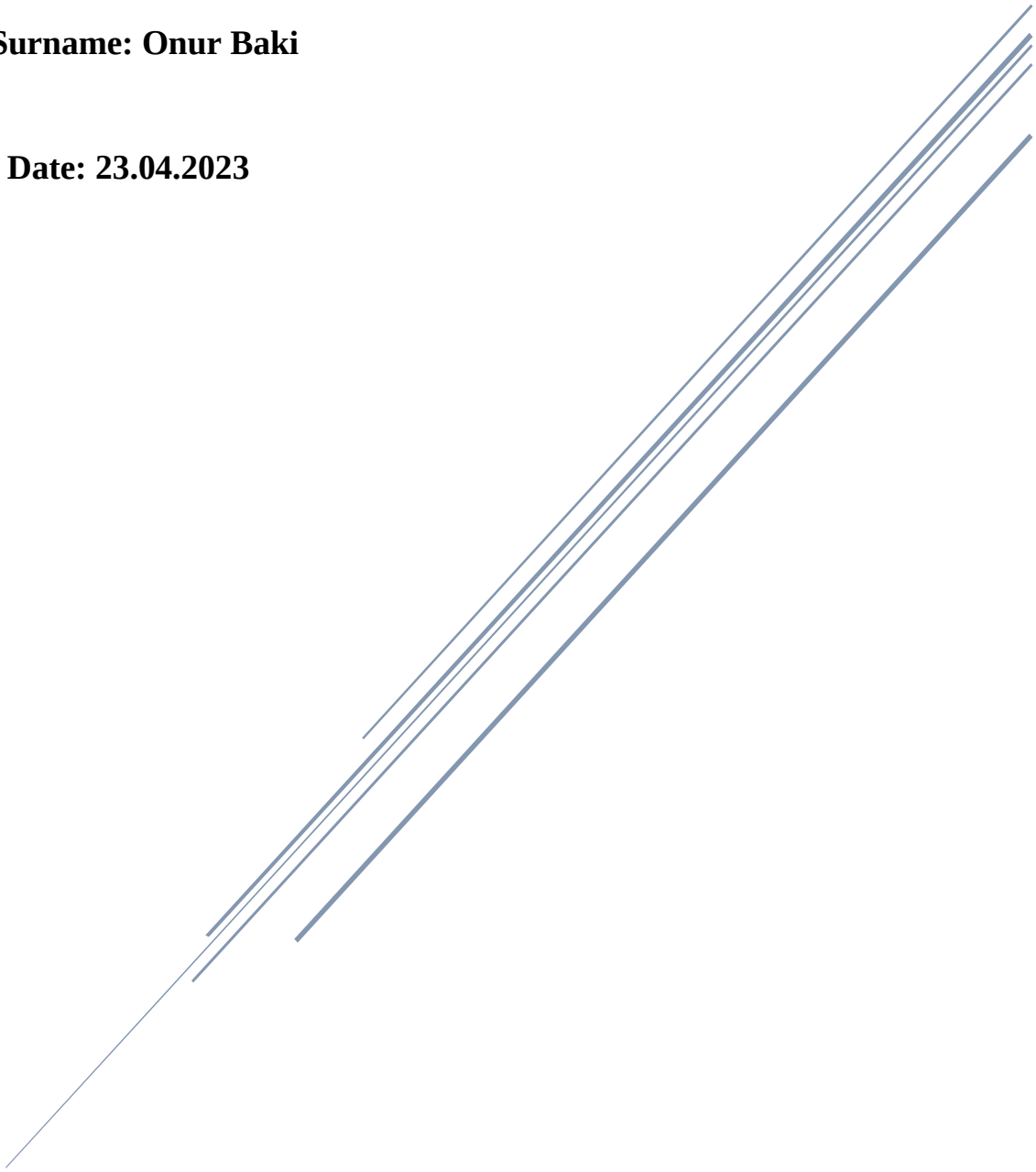
SMART HOME SYSTEM PROJECT

Student Number: 2210356081

Name - Surname: Onur Baki

Delivery Date: 23.04.2023



HACETTEPE UNIVERSITY

BBM104 SPRING 2023

Index

Benefits of OOP and Four Pillars of OOP

Object-Oriented Programming (It will be mentioned as OOP later that report) is a programming paradigm that focuses on creating objects that interact with each other to solve a problem. It is based on the concepts of classes, objects, encapsulation, abstraction, inheritance and polymorphism.

There are several benefits of OOP:

1. Modularity: OOP allows you to break down a complex problem into smaller, more manageable parts. Each part can be developed, tested, and maintained separately, which makes it easier to modify and reuse code.
2. Reusability: OOP promotes code reuse. You can create classes that can be used in different parts of the program or even in different programs altogether. This saves development time and improves the quality of the code.
3. Encapsulation: OOP encapsulates data and methods within objects, which means that the internal workings of an object can be hidden from the outside world. This protects the data and ensures that it is manipulated only in the ways that the object's methods allow.
4. Abstraction: Abstraction is the process of simplifying complex systems by breaking them down into smaller, more manageable parts. It allows developers to focus on the most important aspects of a program and ignore irrelevant details. In OOP, abstraction is achieved by creating abstract classes and interfaces that define the essential properties and behaviors of an object without specifying their implementation.
5. Inheritance: OOP allows you to define a new class based on an existing one, inheriting its properties and methods. This saves development time and reduces the likelihood of errors.
6. Polymorphism: OOP allows you to define multiple methods with the same name, but with different implementations. This makes the code more flexible and easier to maintain.

What is UML

UML (Unified Modeling Language) is a standardized language used in software engineering to visually represent software systems. It is used to create diagrams that help in designing, documenting, and communicating different aspects of a software system, including its structure, behavior, and relationships between components. These diagrams can be used by developers, stakeholders, and clients to gain a better understanding of the system being developed.

Definition of The Problem

In the Smart Home Systems project, objects are used to represent the different types of smart devices, such as SmartPlugs, SmartCameras, SmartLamps, and SmartLampColors. Each object has its own set of attributes and methods that define its behavior within the system. For example, a SmartPlug object may have attributes such as the device name and status, and methods such as turnOn and turnOff to control its functionality. As it seems, the Project requires a complex OOP design. Thus, the Project is led to four pillars of the OOP which are Encapsulation, Abstraction, Inheritance and Polymorphism.

Solution Approach for The Problem

The development of a software system for controlling various types of smart devices in a home environment was required for the Smart Home Systems project. An object-oriented programming (OOP) approach was followed using the Java programming language to solve this problem.

The different types of smart devices that the system needed to support were identified, including smart plugs, cameras, lamps, and color lamps. Separate classes were created for each type of device, each with its own properties and methods.

A DeviceList class was created to store a list of all connected smart devices to manage the smart devices. The DeviceList class had methods for adding new devices, removing existing ones, and finding devices by name.

To keep track of the current time, a Time class was implemented, and a DateTimeValidator class was created for validating dates and times.

To interact with the smart devices, a SmartDeviceRunner class was created that provided a command-line interface for the user to input commands. The SmartDeviceRunner class had methods for adding, removing, and finding devices, as well as for performing actions on the devices such as turning them on or off.

Finally, a WriteToInputFile class was implemented that wrote the system outputs to a file for record-keeping purposes.

Overall, the solution approach for the Smart Home Systems project involved identifying the problem requirements, breaking down the problem into smaller components, and implementing an object-oriented solution that leveraged the strengths of the Java programming language.

Faced Problems and Solutioun Approachs

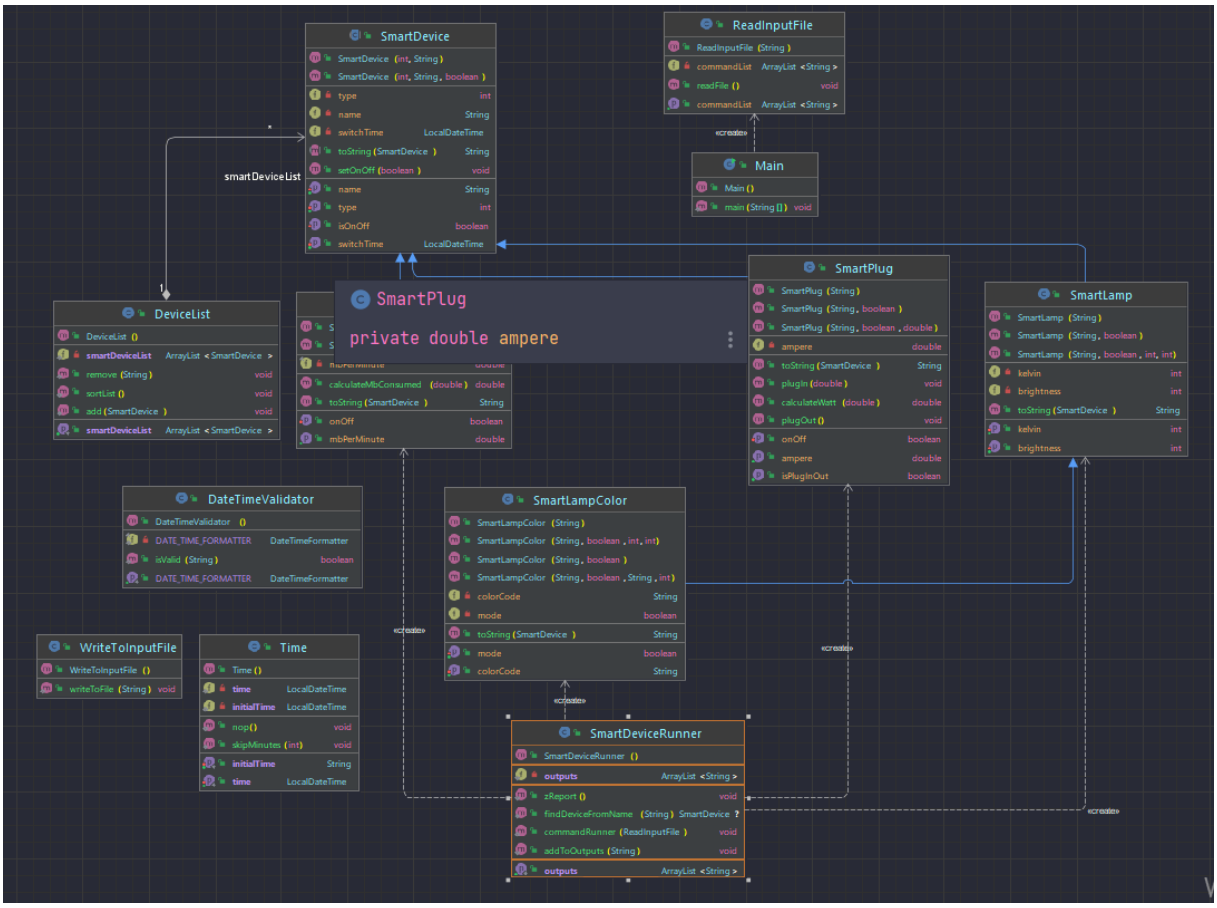
Several issues are occured and several solutions are developed while the Project is being designed.

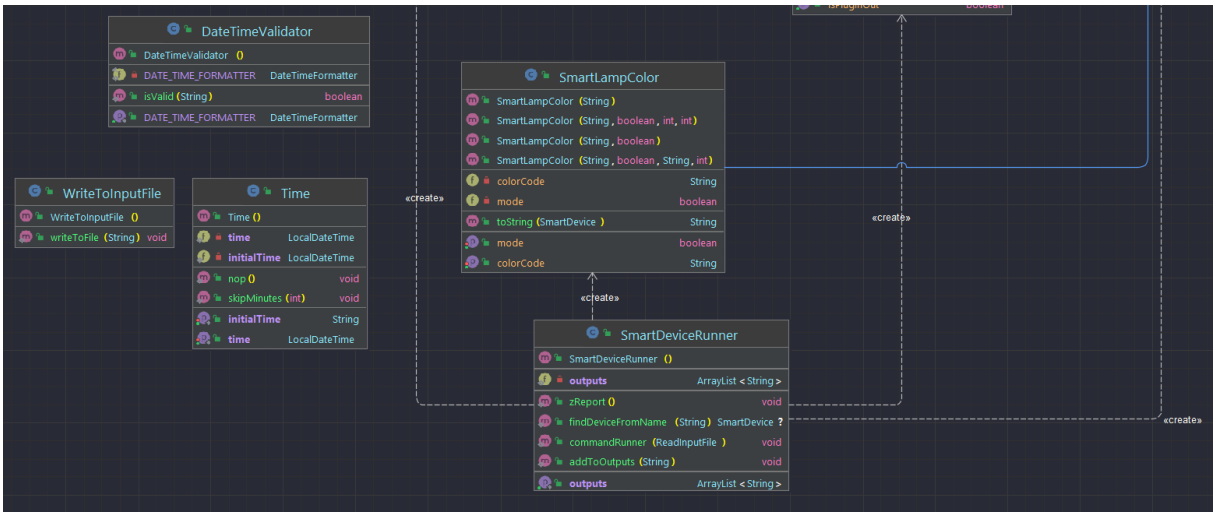
1. Parsing given time values was a big problem since lack of information about DateTimeFormatter class, so a major research are made before the problem is solved.
2. Devices are added in the smartDeviceList type of SmartDevice, not with that devices specific type like SmartLamp. Therefore, necessary casting operations are made before using an smartDeviceList item, but this approach causes ClassCastException sometimes, so necessary try and catch blocks are used.
3. Another major problem is controlling time flow and doing switch operations. Several classes and methods are used to solve that problem such as SmartDevice class, Time.nop() method and Time.setTime() method.
4. The biggest problem of the Project is sorting smartDeviceList list. Comparator class is used to solve this problem based on switchTime, nullsLast and naturalOrder. Yet, for some cases that sorting method is not enough as I also explained in my checklist.

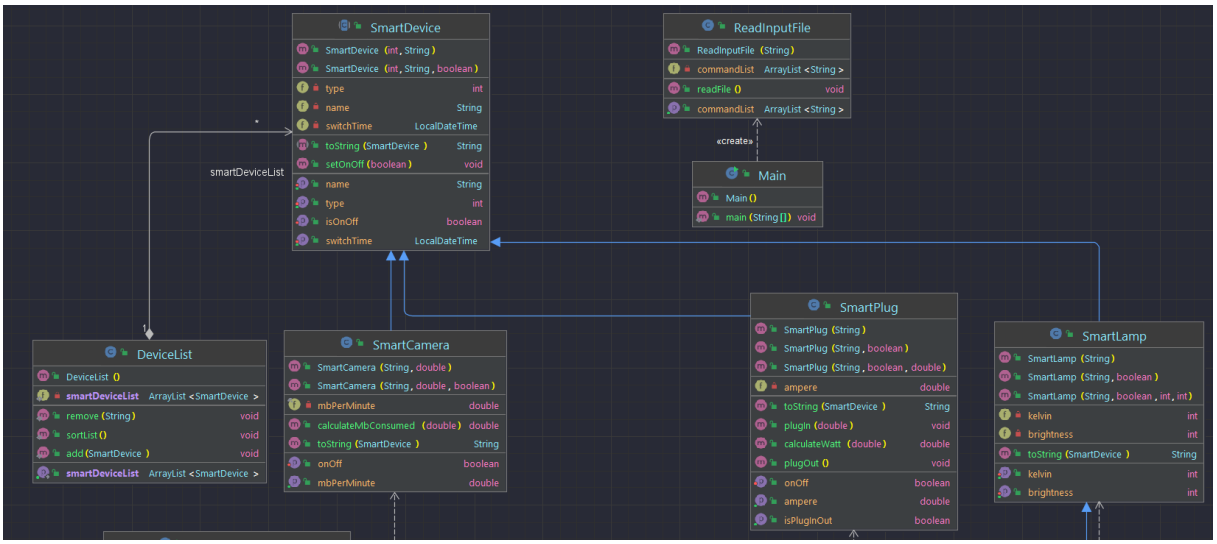
Benefits of Smart Home System

The Smart Home System offers several benefits such as the ability to control various devices from one centralized location, making it more convenient and time-saving. The system also offers improved energy efficiency by allowing users to easily monitor and control their energy usage. Additionally, the system offers improved security features by allowing users to monitor their homes remotely and receive alerts in case of any security breaches.

UML Diagram of the Project and Its Explanation







As it can be seen from UML diagram, there is an abstract SmartDevice base class, and it contains some of the common fields and methods such as name and isOnOff fields, their getters and setters. There is 4 subclass of SmartDevice class: SmartPlug, SmartCamera, SmartLamp and SmartLampColor.

1. SmartPlug: The Class is used to create SmartPlug objects and makes several operations. It has special fields like isPlugInOut and ampere.
2. SmartCamera: The Class is used to create SmartCamera objects and makes several operations. It has special field mbPerMinut.
3. SmartLamp: The Class is used to create SmartLamp objects and makes several operations. It has special fields like kelvin and brightness.
4. SmartLampColor: The class is the subclass of SmartLamp, it is used to create SmartLampColor objects and makes several operations. It has special fields like mode and colorCode.

ReadInputFile class reads input file, where commands are written, and stores them into the commandList field line-by-line. It also has fields like fr type of FileReader and br type of BufferedReader.

DeviceList class stores the smartDevice objects in the field smartDeviceList type of ArrayList. It also has add and remove methods for add and remove commands.

DateTimeValidator class checks wheter a string is suitable for DATE_TIME_FORMATTER constant's format by isValid method.

Time class holds initial time and instant time in initialTime and time fields. It also makes setTime, skipMinutes and nop operations.

SmartDeviceRunner class is the runner for commands, it has outputs field type of ArrayList which contains results of each command line. It launchs command lines line-by-line through necessary if-else or swtich-case statements.

WriteToInputFile class writes results of commands to a file through writeToFile method.

Main class is just a runner class for other classes.