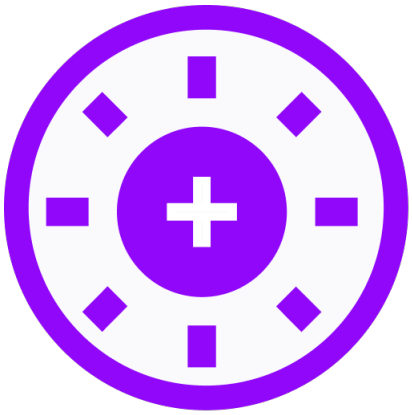

Chalmers University of Technology

OBJEKTORIENTERAT PROGRAMMERINGSPROJEKT

Requirement and Analysis document for EnergyCalculator

Alexander Larnemo Ask
Jonatan Bunis
Vegard Landrö
Mohamad Melhem
Alexander Larsson Vahlberg

VAMAJ

2019/10/25
Version 8.0

Contents

1 Introduction

The harnessing of solar power for both commercial and private use is a growing trend. This project aims to adopt that trend by providing value to the user through information about their potential gains with solar power. The user can through the program see whether solar power would be a viable investment economically, as well as if there would be a future possibility of profit.

The comparisons are made based on the users current electricity provider. This is meant to further encourage the user towards investing in solar power. The user should also be assisted in his/her choice of solar panel so that a good balance between installation cost and produced energy can be suited to the user. The program aims to help private residents as well as businesses with the decision process concerning the acquirement and use of solar power. These two groups are therefore the projects stakeholders.

1.1 Definitions, acronyms, and abbreviations

The following is a list of words used throughout this document and their explanations:

GUI ”Graphical User Interface” also ”User Interface”, the part of the program that the user sees and interacts with.

DoD ”Definition of Done”, a list of requirements that have to met for a User Story to be considered done.

User Story short, simple descriptions of a feature told from the perspective of the person who desires the new capability, usually a user or customer of the system.

RAD ”Requirement and Analysis document”.

SDD ”System Design Document”.

API ”Application Programming Interface”, an interface or communication protocol between a client and a server intended to simplify the building of client-side software.

Hi-Fi ”High fidelity”, a sketch with much detail.

UML ”Unified Modeling Language”, A visual language for mapping and representing systems. Can be used for modelling och class diagrams, sequence diagrams and more. In this project used for Domain and design model.

2 Requirements

2.1 User Stories

This subsection lists the user stories that define the application's requirements, in terms of the end-user's wanted functionality. User stories surrounded by asterisks (E.g. *UserStoryX*) are not implemented in the application due to time constraints. User stories that are developer focused can be found in the appendix.

2.1.1 UserStory1.2 : Solar panel choice

Description:
As a user I want to be able to select different solar panels and compare them with my current electricity provider plan, in terms of cost and produced energy.

Confirmation:
- The user can select different solar panels in the application.
- The user can input electricity provider specifics.
- Calculations are based on the given information.

2.1.2 UserStory3 : Available installation space

Description:
As a homeowner I want to know the cost of a potential solar-panel installation based on the available space my roof has.

Estimated time: 1 hour

Confirmation:
- The user is able to enter a house's available space for a solar panel installation.
- The user gets the installation expenses based on input.

2.1.3 UserStory4 : Current electricity consumption and cost

Description:
As a homeowner I want to be able enter my current electricity cost and consumption so that comparisons can be based on it.

Confirmation:

- The user can enter electricity cost
- The user can enter electricity consumption.

2.1.4 UserStory4.1 : Energy production surplus

Description:
As a money-loving person, I want to check if I would get electricity surplus so I can sell some of it.

Estimated time: 1 hour

Confirmation:
- The user gets expected solar energy production.
- The user can enter energy consumption.
- The program calculates surplus based on a user's energy consumption and solar energy production.

2.1.5 UserStory4.2 : Break-even

Description:
As a money-loving person, I want an estimation of when I would start turning a profit after a purchase of solar power.

Estimated time: 1 hour

Confirmation:
- The user can see an estimate of when a property's solar power setup would start turning a profit.

2.1.6 UserStory6 : Solar hour statistics

Description:
As a solar power prospector I want the yearly amount of solar insolation my property has to be a variable in calculations, so that I can determine if solar power would be viable.

Estimated time: 2-12h hours

Confirmation:
- The amount solar insolation a property has is based on actual real world data and statistics.

2.1.7 UserStory7 : Government subvention

Description:
As an economical person I want to be able to see how much the government would be willing to subvent in my potential purchase of solar power to save money.

Estimated time: 1 hour

Confirmation:
- The user gets an estimate of how much the government would be subventing the installation cost.
- The user gets an estimate of the subvented installation cost.

2.1.8 UserStory7.1 : User friendly design

Description:
As a user I want the program to have a straight forward GUI that conveys what input is needed for a prospect of a solar setup based on my prerequisites, since that's what I'm interested in.

Estimated time: 4 hours

Confirmation:
- The program clearly conveys to the user what input is required and where said input is needed.
- The user finds the GUI pleasant to look at and easy to navigate.

2.1.9 *UserStory10*: Estimated energy consumption

Description:
As a user I want to be able to calculate electricity consumption based on my address, in order to calculate electricity costs.

Estimated time: 5 hours

Confirmation:
- The user's address can be entered.
- The address is taken into account in all relevant calculations.

2.1.10 *UserStory10.2* : Environmental effect

Description:
As a homeowner I want to be able to see how much my current energy alternative effect global warming so that I can compare it with solar power

Estimated time: 6 hours

Confirmation:
- A user's current energy source's environmental effect is compared with a potential solar powered energy source.

2.1.11 UserStory12 : Find current location

Description:
As a user I want an easy way to use my current location in the calculations, since I want calculations based on where I live.

Estimated time: 2-12 hours

Confirmation:
- The user can press a button to find the current location.
- Current location is used in calculations.

2.2 Definition of Done

The following is the DoD used in the project:

1. All of the acceptance criteria of the user story have been fulfilled.
2. All code is commented.
3. All code has tests that run without failure.
4. All code has passed code review and potential changes have been made.
5. All changes have been committed and pushed to the master branch and the master branch is running without error.
6. The RAD and SDD Documents have a few paragraphs about the classes pertaining to the user story.
7. The Design UML has been updated to show relevant classes.

2.3 User interface

In the beginning of the project a preliminary sketch of the user interface was made to help with the visualization of the program (See figure 1). []

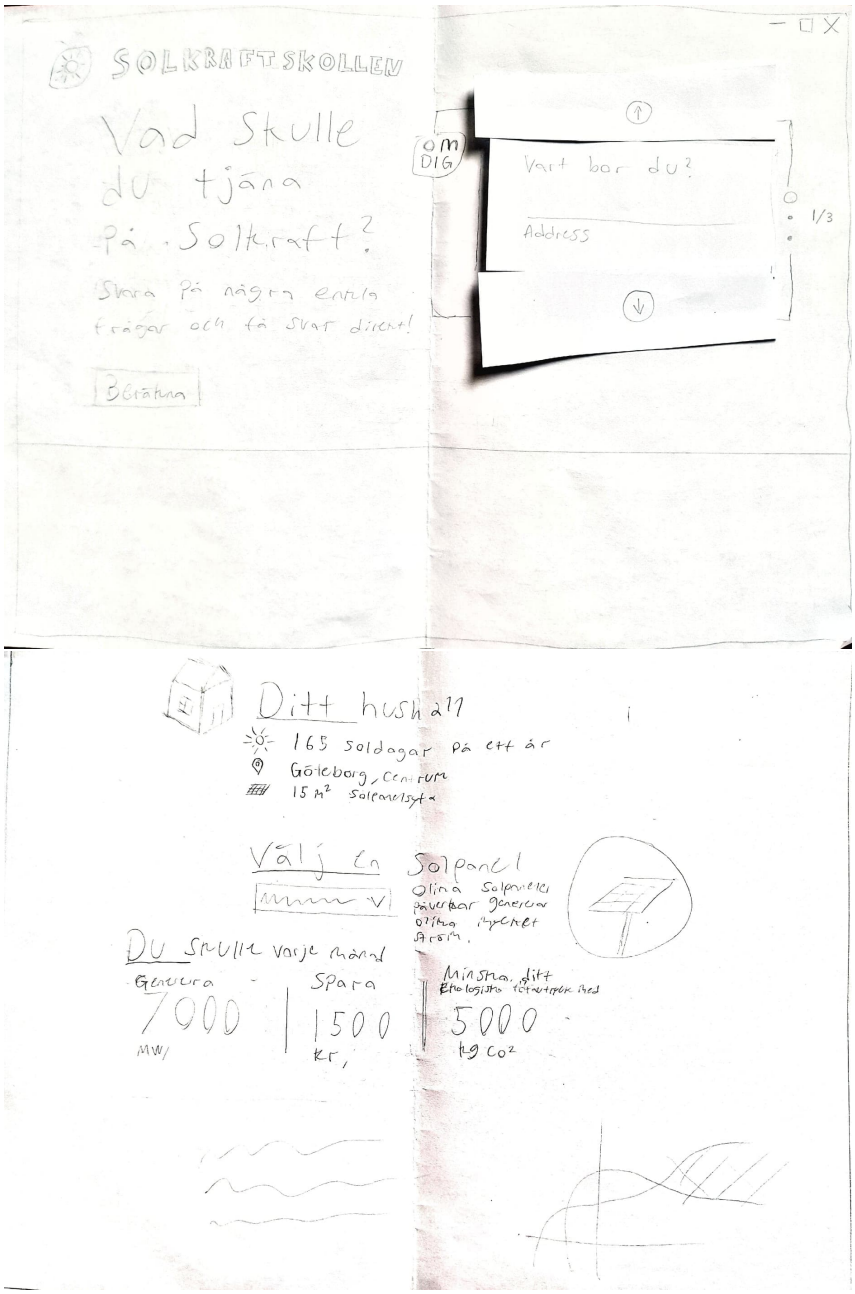

Figure 1: A Preliminary sketch of the interface

Later the interface was made more defined through the creation of a High fidelity (detailed) mockup sketch, created in ”Figma” (See figure 2). [1]

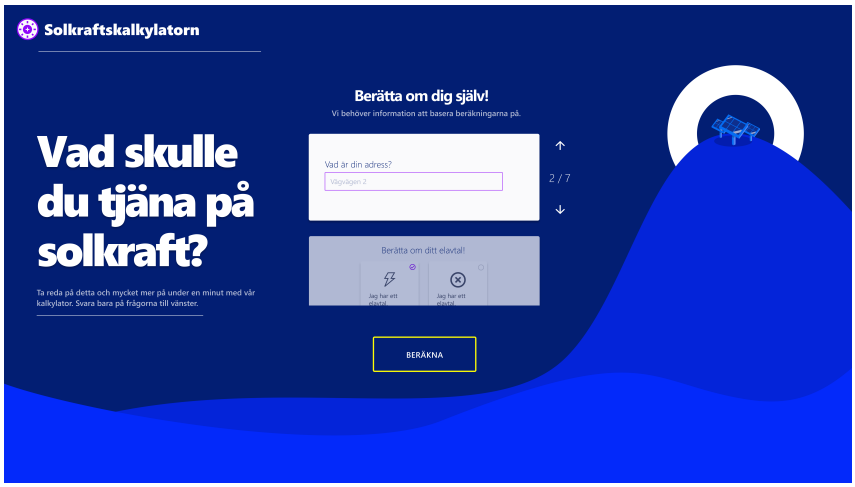

Figure 2: A Hi-Fi mockup of the interface

The Graphical user interface consists of a main scene which has a dynamic part where questions needed for the calculations are viewed as question cards. The user is be able to navigate between different question without breaking the flow or changing scenes. When answering the whole form, the results can be shown in another page. This flow was made to create a clear distinction between information gathering and information provision.

The GUI was created with the intent of making the process feel light and interesting. The final result of the implemented GUI can be seen in figure 3. The GUI differs from the mockup mostly due to technical limitations and practical reasons.

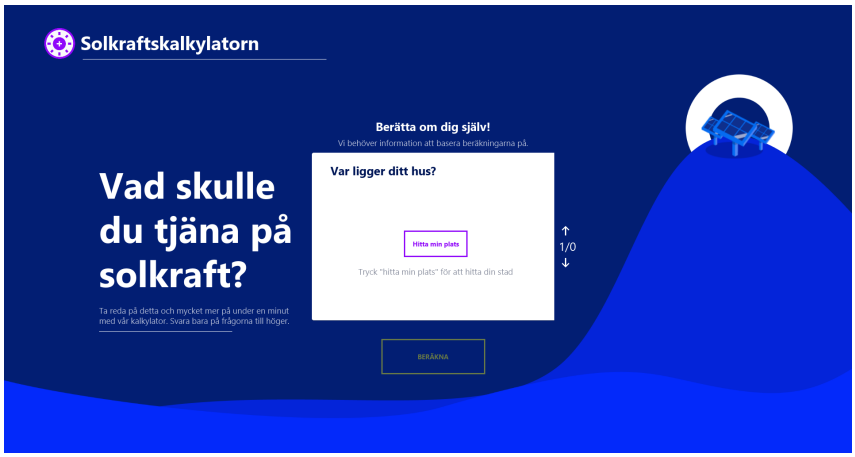

Figure 3: The GUI of the actual program

3 Domain model

The program is a type of calculator that uses user input and input from various **APIs** to produce informative results which is then fed back to the user through a **GUI**. The user is the main information object of the program and the user is the owner of a number of properties. The properties in turn hold most the information necessary in order to make the calculations. The **model** aims to mimic the information relations that would be found in the real world so that information can be accessed and encapsulated in intuitive ways. The way the application works is illustrated in the domain model (figure 4) and further explained in chapter 3.1.

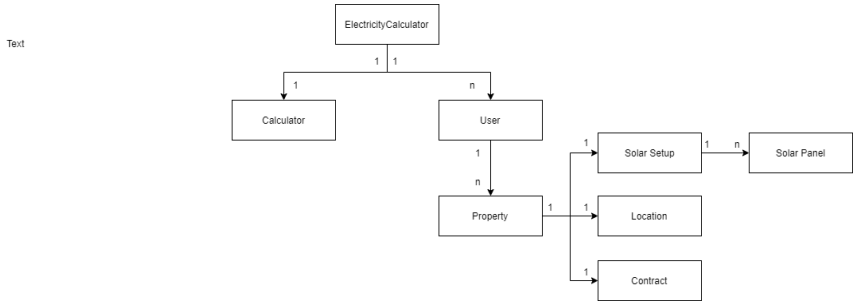

Figure 4: The domain model of the program

3.1 Class responsibilities

The **calculator** is responsible for the main logic of the program i.e. the actual calculations in the program. The **user** is an object that holds some data specific to the current user, it also holds the important paets of the model that are going to be used by the **calculator**, that is to say, the user is the central object in the model hierarchy. The **calculator** will retrieve information from the **user** (i.e. the model) perform its calculations after which the view will be updated to display results.

The structure of the program has been created specifically so that the information structure mimics reality (for readability and structure) and so that the model is independent of the rest of the program so that the program is modular.

Every user can have multiple Properties. A **property** is the object with the actual necessary information to perform the calculations. The information is stored as objects; **Solar Setups**, **Location**, Electricity Provider **(contract)**. The actual information in theses objects is gathered through different measures: through data from APIs and through direct input from the user.

The **location** class is a representation of a location. It holds data and attributes that are used in the calculations.

The **Solar Setup** holds the needed logic to instantiate a **solar setup** object. A **solar setup** has multiple **Solar panel**s and information about the collection of **solar panel**s as a whole.

Contract is an abstract representation of an electricity provider contract. The **contract** is used by different parts of the code in order to store needed information about the users electricity provider to carry out further calculations.

References

[1] Figma, “Mockup tool,” 2019. [Online]. Tillgänglig: https://www.figma.com Hämtad: 2019-10-07.

Appendices

Developer-related user stories

.0.1 UserStory0 : Development environment

Description:
As a group of developers, we want to have a well-functioning developing environment with all needed tools installed and ready, so we can do our job.

Confirmation:
- All members of the group have functioning development tools (Git, Maven, Junit, SceneBuilder och Intellij)

.0.2 UserStory1.1 : Basic functionality

Description:
As a Developer I want to be able to see a visual representation of a rudimentary version of the program as well as have a good model structure of the program with basic functionality so that I can get a clearer view of the next development steps and so that I know where things should go.

Confirmation:
- There is a simple GUI that represents the information input in the program.
- The classes specified in the design model have been created. and associated with each other appropriately.
- There are static hard coded values that can be used by the calculators.
- There is a simple economical algorithm to ensure the credibility of the results.

.0.3 UserStory2 : Attributes

Description:
As a developer I want to be able to estimate solar electricity production based on a property's attributes, in order to lay a foundation for the core functionality of the program.

Confirmation:
- Solar electricity production is calculated based on predefined variables in the property

- Appropriate classes have been created and the data has now been moved to there.

.0.4 UserStory3.1 :

Description:
As a developer, I want to create a service module to communicate with different API:s, in order to gather data for my domain model and calculations.

Estimated time: 2h-12 hours

Confirmation:
- We have (service) classes that can create model objects based on data from the API:s.

- We have created interfaces that allow us to add new API:s, without disturbing other parts of the program.

.0.5 UserStory3.2 : Consistently organised

Description:
As a developer I want the way that information travels into the model through user input is consistent with how it travels from API:s so that the code structure is logical.

Estimated time: 3h

Confirmation:
- The controller forwards input to the model that creates the objects.

- Immutability is used as much as possible.

.0.6 UserStory10.9 : Defined program structure

Description:
As a developer i want all the parts of the program to be connected and working together so that i can get a clear view of how the program works.

Confirmation:
- The inner functionality of the program has been linked together so that the program uses actual values and performs actual calculations.

- The data flow inside the program works as intended.

- The GUI has been coupled with the model in way the the user can interact and get real responses.

.0.7 UserStory13 : Foundational GUI

Story name:
Description:
As a developer I would like to see how the functionality of the GUI should work so that I know how to implement my features.

Estimated time: 3 hours

Confirmation:
- Placeholder features are implemented in the GUI .