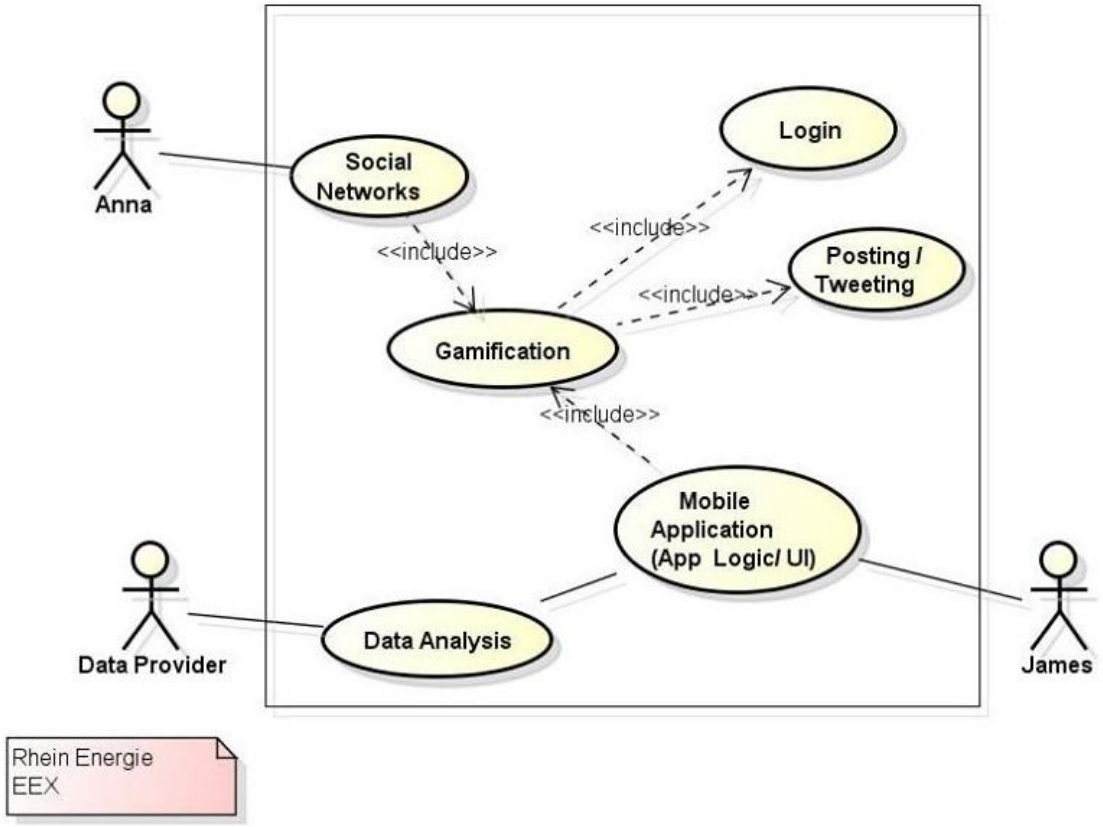
Architecture

1. Context

There are two resources of energy: one is renewable energy and the other is conventional energy. There are times that large a amount of renewable energy (sun, wave, wind) is produced. So how can we make more use of this energy? Every consumer has some influence on his energy consumption. For example, we can decide when to use our washing machines or dishwashers. So what we intend to do in this project is to advise consumers when it is the best time to turn their energy consumers on, (depending on when the percentage of green energy is higher) in order to save conventional energy.

2. Functional View

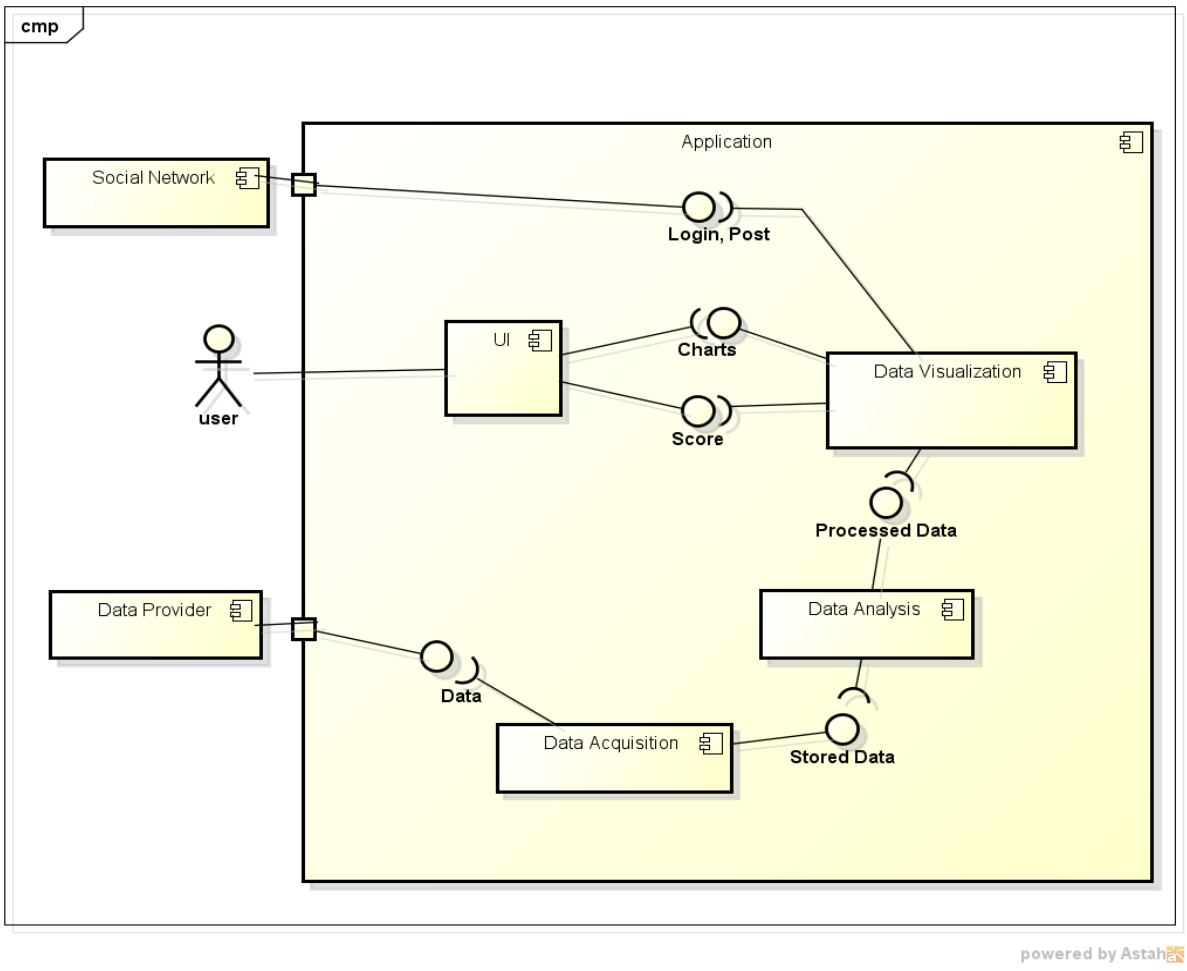


First, we need data providers to get data from, as it can be seen in the use case diagram. We need to know about current renewable energy production and also energy demand at several times of a day.

We use this information in our application to advise the users and also show them how many kilo-watt-hours are shifted to green energy production times. Our application consists of three main parts: data which was already mentioned above, the UI in which the data is visualized, and application logic which enables the visualization of data.

There are two scenarios here. Our application can be used by either one consumer or more. Imagine the person James wants to use the app. He would login and see the best times to turn his energy consumer on. The other scenario is that James logs in using a social network, like Facebook or Twitter. In that case he can share with others how many scores he got by saving conventional energy. This would result in encouraging others to also make use of this app. In our example Anna sees John's post and becomes interested in using the app as well.

3. Process View



The Component Diagram above shows what the system does and the major steps at a high level. It shows the overall process which will be implemented. More precisely, the first external component is

"Data Provider" which is connected to the internal component "Data Acquisition" via an interface. This internal component obtains data from the interface by sending requests to it and enables data analysis by storing the received information. The component "Data Analysis" converts stored data into processed data which is the result that we want to visualize to the user. Hence processed data is the input for the component "Data Visualization". The components "Data Visualization" and "UI" are double-sided connected to each other as they both are receivers as well as providers. To be exact the score which is achieved by a user is visualized by the UI in form of charts. Furthermore the component "Data Visualization" is connected to the external component "Social Network" as the user can decide to login and post to certain social networks. As the only (purpose) of the UI is to visualize data, the connection to social networks is initiated form "Data Visualization" as it is one of the data processing components. The user at the end is only connected to UI to use the application.

4. Non-Functional View

1. Data Processing:

1.1 Data Acquisition/Access: system gets data (renewable energy production times and energy consumption times) from data resources (*make clear that it's a program feature, not gathering by sending requests to companies...*).

- Source/Initiator: Developers
- Priority: High

1.2 Analyzing data: process the mentioned data to achieve the statistics (compute recommendation).

- Source/Initiator: Developers
- Priority: High

2. Data Visualization:

2.1 Visualizing data: show the computed recommendation to user.

- Source/Initiator: Developers and Users
- Priority: High

2.2 Posting data to social networks: system sends the data to interfaces of social networks, to attract more users to the system and encourage the user to continue to use the system more often.

- Source/Initiator: Developers and Users
- Priority: High

3. User Interface:

3.1 main window, which shows the current percentage of green energy

3.2 comparison page, which shows energy consumption in different time slots

Non-functional Requirements

1. **Usability:** The visualized information should be in such way that users can easily understand, and not in too much details.
    - *Source/Initiator: Developers and Users*
    - *Priority: Medium*

2. **Performance (Availability):** In case that there is no internet connection, the system is not up to date.
    - *Source/Initiator: Developers and Users*
    - *Priority: high*

3. **Reliability:** It is guaranteed that the system processes and visualizes the data properly. But since the data depends on other resources, it may not be working, in case that the format of the data from providers has changed or if they do not deliver data anymore.
    - *Source/Initiator: Developers*
    - *Priority: Medium*

4. **Implementation requirement:** The system will be developed only for the Android platform.
   We did some research about cross-platform development and we realized it would take too much time to set up the environment, installation, read guides, and running on different devices. You can find more about this in the attached document (*Research on PhoneGap.pdf*).
    - *Source/Initiator: Developers*
    - *Priority: High*

5. **Interface Requirement:** The system should have
   5.1 Interface to a social network
   5.2 Interface to a data provide r(see document *Research on Data Providers.pdf*).
   The interface includes:
    - *Source/Initiator: Developers*
    - *Priority: High (interface to data provider), Medium (interface to social networks)*

6. **Legal Requirement:** For the system we may need an agreement with data providers and we might have to agree to the license agreements of social networks. Further research on this will be done in the future.
    - *Source/Initiator: Developers*
    - *Priority: High*

7. **Scalability:** The application is supposed to be equipped with the feature to show the amount of kilowatt-hours shifted to green energy production times to the users. When the number of users grows this feature becomes more and more useful.
    - *Source/Initiator: Developers*
    - *Priority: Low*

Related Documents: "Research on Data Providers.pdf", "Research on PhoneGap.pdf"

5. Data View

5.1 Data about energy in total and amount of green energy

The type of data

The data we need from providers is the amount of energy in total and the amount of green energy in NRW. Preferably the data would be specified in KW/H but dealing with percentage rates would also be tolerable. The data should be as detailed as possible, in an ideal case it would be updated every day several times.

How the data is stored

This kind of data needs to be available to every user and therefore it needs to be on a server in order to guarantee its availability and consistency.

In an ideal case we could get data from a company or government which is filed in a database and can be accessed via an API. This would enable us to retrieve data in form of e.g. XML or JSON which we can parse in a way that it fits to the data structure of our application and enable further processing of data. It is favorable to access and use the data which is stored on the server of a provider and not copy this data to databases of ourselves as this would require a lot of storage.

If we cannot get real, current data for now, we have to draw on made up test data. In order to simulate the data acquisition in our application as accurate as possible, this data should also be stored in a database on a server.

5.2 User Data

The type of data

For our application the user data in the first place consists of the score that we award to the user. Of course the amount of energy which is consumed or saved by the user also belongs to this

type of data. User data which is also processed by our application are login information if the user decided to connect our application to a social network.

How the data is stored

There are several possibilities on how to store user data. If the data (i.e. a user's score) is supposed to only be accessed by the user himself or when he uses the application it can be stored on the users device. Depending on the amount of data we store it in a file or in a local SQLite-database. Since it provides to store data in a structured way the SQLite-database is preferable.
In general storing data which only needs to be accessed locally on the user's device is the better choice (i.e. run time), but there are two aspects that we have to consider which are indicative of storing user data in our own databases.

The first aspect is that we should give the user an opportunity to backup his user data in order to restore it on another device. As we decided to integrate the *log in* as an optional feature we can only support backup of data from users who made use of this feature. More precisely we need login-data to identify the data correctly which is related to a user.

The other aspect is that we need to access the user's data in order to gain/create statistical data. Depending on the statistical features that we want to provide we might also need the user's login data (*see next section*).

5.3 Statistical Data

The type of data

The statistical data should be identified in the first place is the amount of shifted energy. There is also the possibility to create statistical data by analyzing user data. An example for this would be the creation of a high score which points out the "greenest users".

How the data is stored

As statistical data can only be acquired by accessing data from several users this kind of data cannot be stored only locally. As we need access to the data it has to be stored on a server. Information about the amount of shifted energy can be stored in a database and the data sets are not required to lead back to the users. More precisely we only need the total amount of shifted energy, maybe related to the time/days when it was shifted.

If features which require further statistical data are implemented, like a high score of the "greenest users", we have to store data about energy consumption related to user data in a database on our server.