Bankkundenurlaubskreditwerbung Projektdokumentation

Team: Reyer, Lampert

Inhaltsverzeichnis

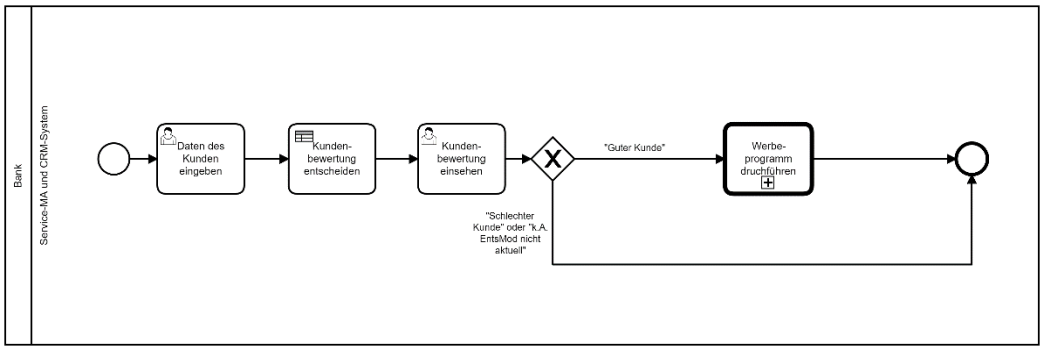
Beschreibung des BPMN Prozess- Bankkundenurlaubskreditwerbung

Mithilfe des Prozesses „Bankkundenurlaubskreditwerbung“ sollen geeignete, kreditfähige Kunden ausgewählt werden, denen eine Werbe-E-Mail über die Kreditaufnahme für einen Urlaub an der Ostsee gesendet wird. Je nach dem wie das durchschnittliche Wetter der nächsten 5 Tage in Stralsund ist, wird dem Kunden ein Badeurlaub, Fahrradurlaub , Wanderurlaub oder Kur-/Wellnessurlaub vorgeschlagen.

BPMN-Prozess

Der Gesamtprozess „Bankkundenurlaubskreditwerbung“ besteht aus 2 gesonderten Prozessmodellen (2 Teilen), die durch eine Call-Activity miteinander verbunden sind.

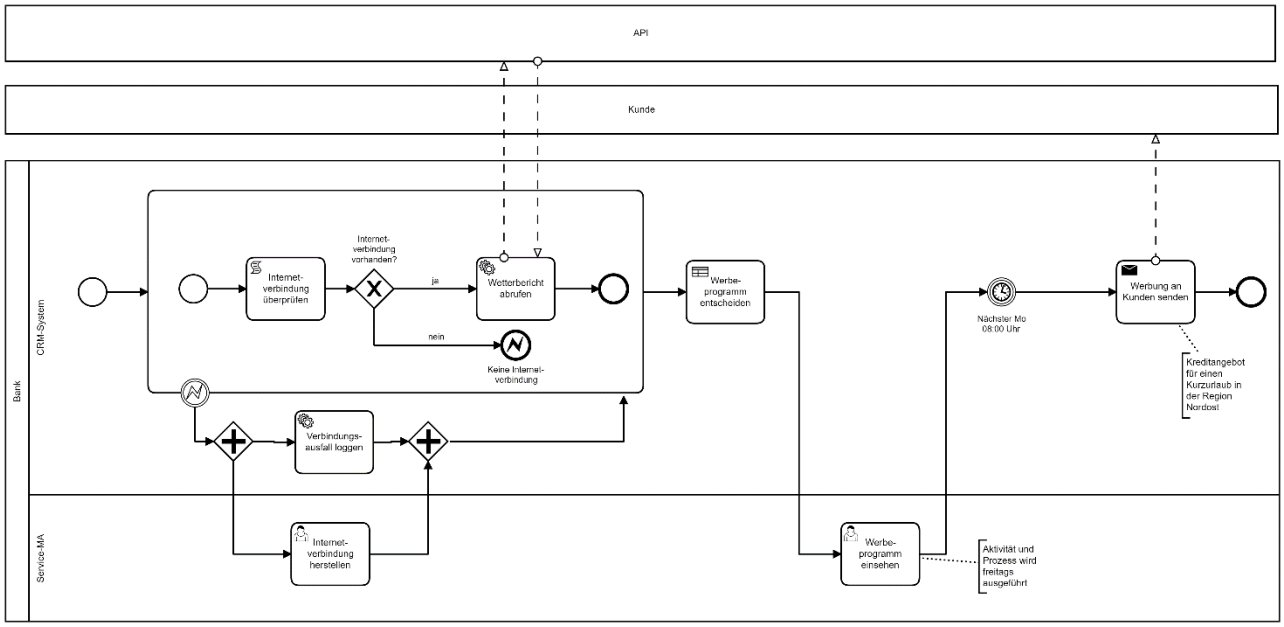
In dem Pool Bank ist die Lane Service Mitarbeiter und CRM- System enthalten. Diese wird sich im späteren Verlauf trennen. Der Prozess startet. Es wird mithilfe einer Usertask, der Servic-MA aufgefordert seine Kundendaten einzugeben. Folglich wird in der Businessrule-Task über die Bewertung des Kunden entschieden.



Hinweis: Die ersten 3 Tasks entsprechen dem Prozess „Bankkundenbewertung“ und wurden in einer gesonderten Dokumentation im Rahmen des Executable DMN-Seminars bereits beschrieben.

Durch die Usertask „Kundenberatung einsehen“ ist es dem Service-MA möglich, diese Bewertung einzusehen. Darauffolgend wird nach der Kundenbewertung durch ein X-Or-Gateway die Entscheidung getroffen ob es sich um einen guten Kunden handelt oder ob es sich um keinen guten Kunden handelt. Ist dies der Fall ist der Prozess beendet. Handelt es sich um einen guten Kunden wird das Werbeprogramm der Bank gestartet. Dieses geschieht durch eine Call- Activiety-Task. Das Werbeprogramm wird durchgeführt.

Durch die Call-Activity wird der 2. Teil des Gesamtprozesses aufgerufen.



Der Pool Bank enthält die Lanes CRM-System und Service Mitarbeiter (o.g.). Diese ist nun in zwei Lanes aufgeteilt. Der Prozess startet im CRM-System mit dem Unterprozess Prüfung der Internetverbindung. Dieses erfolgt mit Hilfe einer Skript-Task. Mittels Throw-Catch-Pattern wird der Prozess an der Stelle beendet, sollte keine Internetverbindung vorhanden sein. Dabei ist es möglich durch eine Usertask eine Fehlermeldung einzusehen, anschließend wird alle 10 min durch das Timer-Event geprüft, ob eine Internetverbindung besteht. Anschließend wird dabei wird mithilfe einer Service Task Wetterbericht abrufen die Wetter-API aufgerufen. Damit endet der Unterprozess.

DMN „Werbeprogramm entscheiden“

Ist der Wetterbericht erfasst, wird folglich, mittels Business-Rule-Task über das Werbeprogramm entschieden und das DRD ausgeführt.

Werbeprogramm entscheiden

`Werbeprogramm_entscheiden`

F	Input +		Output +	
	Bewoelkung_in_Prozent	Temperatur_in_Celsius	Werbeprogramm	
	integer	double	string	Annotation
1	<=40	>=25	"Badeurlaub"	durchschnittliche Bewoelkung und Temperatur der nächsten Woche in Stralsund
2	<=40	>=10	"Fahrradurlaub"	-
3	>40	>=10	"Wanderurlaub"	-
4	-	-	"Kur-/Wellnessurlaub"	-
+	-	-	-	-

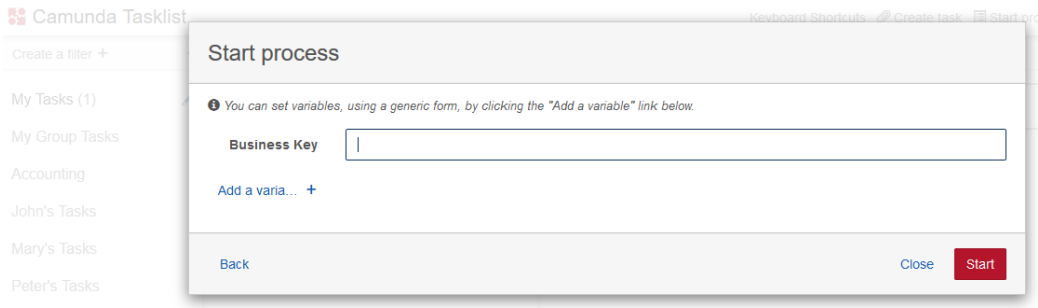
Bei dem DRD handelt es sich um eine Table mit der Hit Policy= First. Der Input für die DRD besteht aus der Bewölkung in Prozent (integer) und Temperatur in Celsius(double). Der Outputwert ist durch eine String-Enumeration dargestellt. Dabei sollen die durchschnittliche Bewölkung und Temperatur (Edit Number: beide Comparison) der nächsten 5 Tage dargestellt werden.

Es gibt genau 4 Möglichkeiten der Ausgabe. Ist die Bewölkung <=40 und die Temperatur >=25 wird ein Badeurlaub vorgeschlagen. Ist die Bewölkung <=40 und die Temperatur >=10 wird ein Fahrradurlaub vorgeschlagen. Ist die Bewölkung >40 und die Temperatur >=10 wird ein Wanderurlaub vorgeschlagen. Trifft die Bewölkung und Temperatur bei keinem der o.g. Werte zu wird ein "Kur-/Wellnessurlaub" vorgeschlagen.

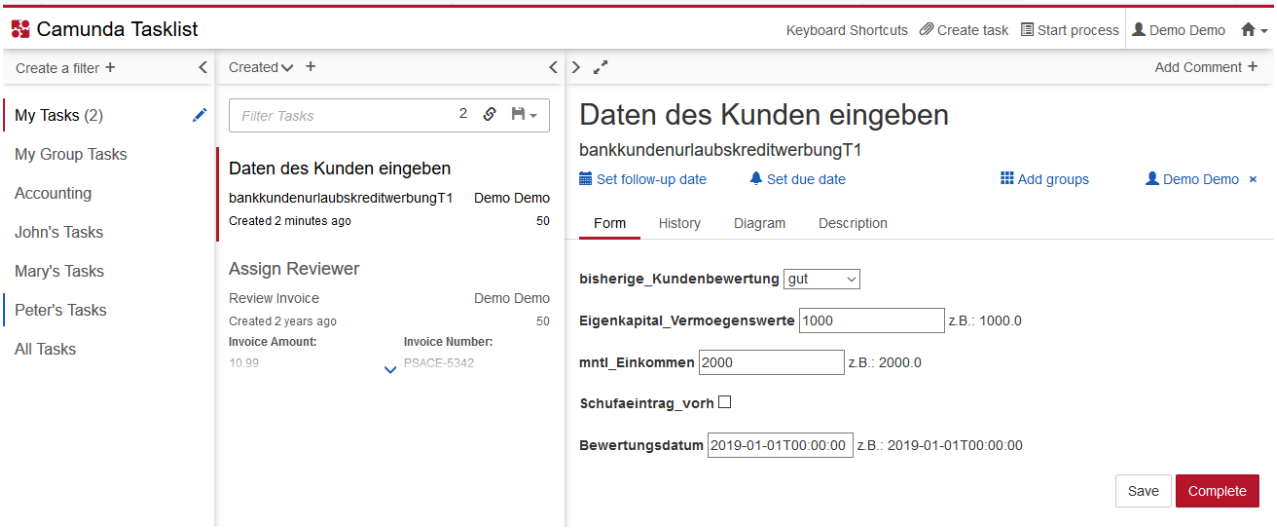
Weiter im BPMN-Prozess

Ist das Werbeprogramm entschieden, kann dieses am nächsten Montag um 8 Uhr, was durch ein Timer-Event gesteuert wird, durch eine User-Task eingesehen werden. Der Prozess soll freitags ausgeführt werden, sodass die E-Mail für den nächsten Montag scheduled wird. Abschließend wird mittels Send Task, ein Kreditangebot für einen Kurzurlaub in der Region Nordost als Werbung zu dem Kunden geschickt. Der Prozess ist beendet.

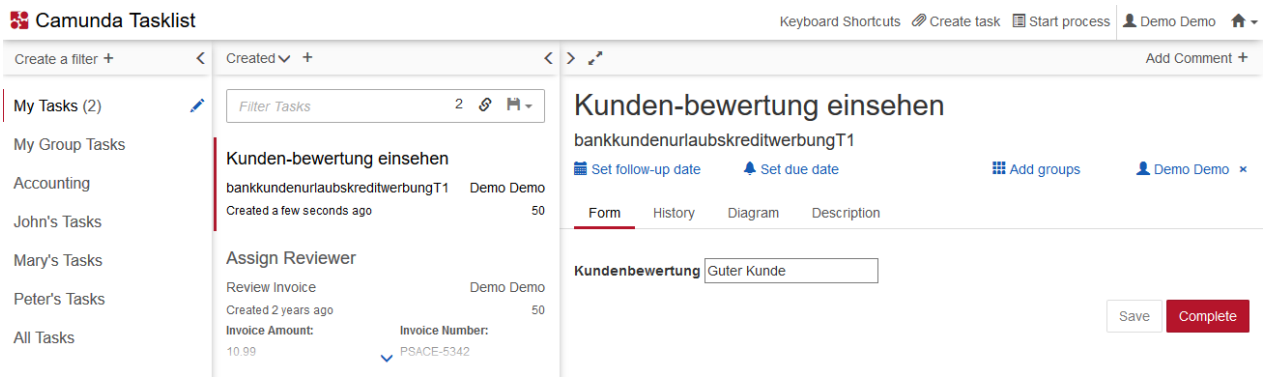
Beschreibung des Workflow



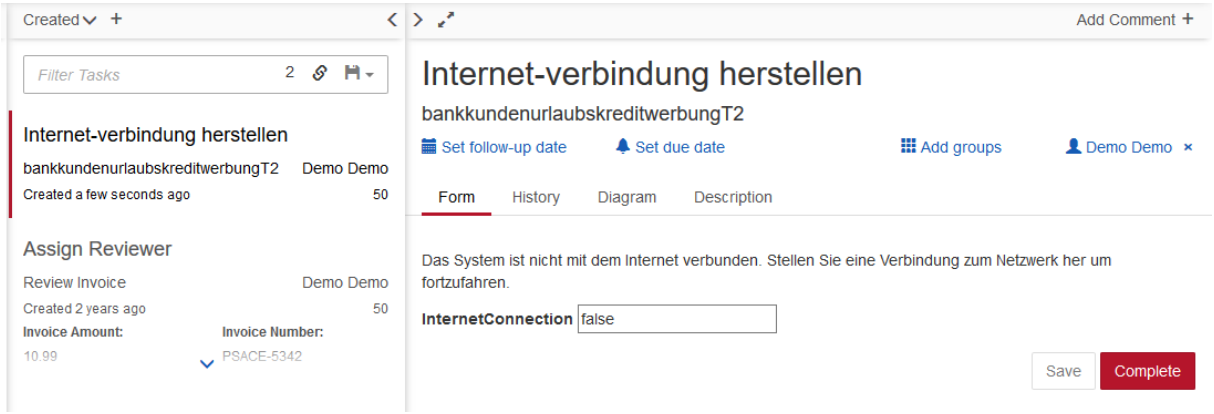
-> Den Prozess starten.



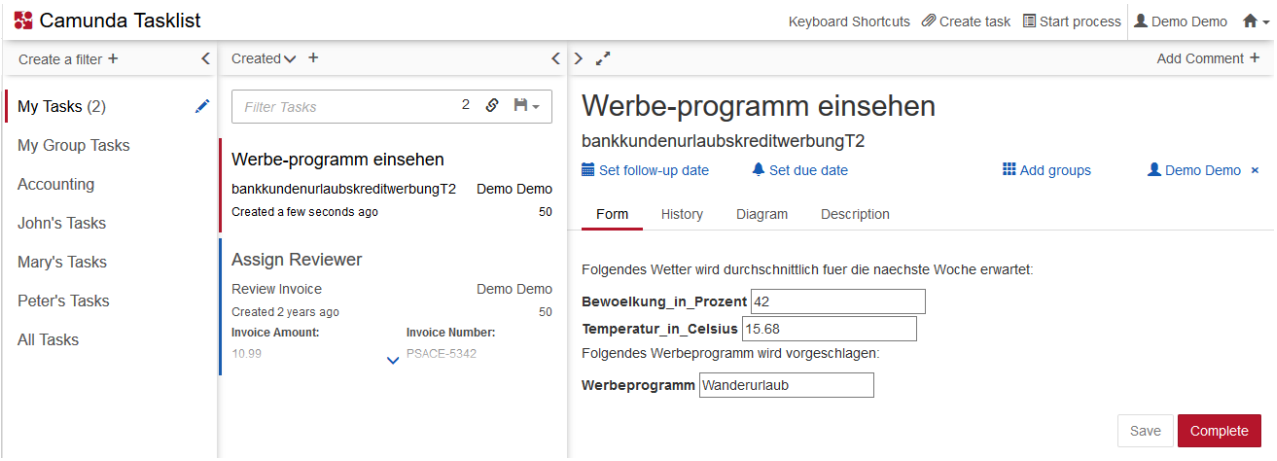
-> Kundendaten werden eingegeben und werden dem DMN-Prozess „Kundenbewertung entscheiden“ zur Verfügung gestellt. Die Kundendaten wurden hier so gewählt, dass der Kunde die Kundenbewertung „Guter Kunde“ erhält. Im weiteren Prozessverlauf wird damit die Call-Activity „Werbeprogramm durchführen“ ausgeführt.



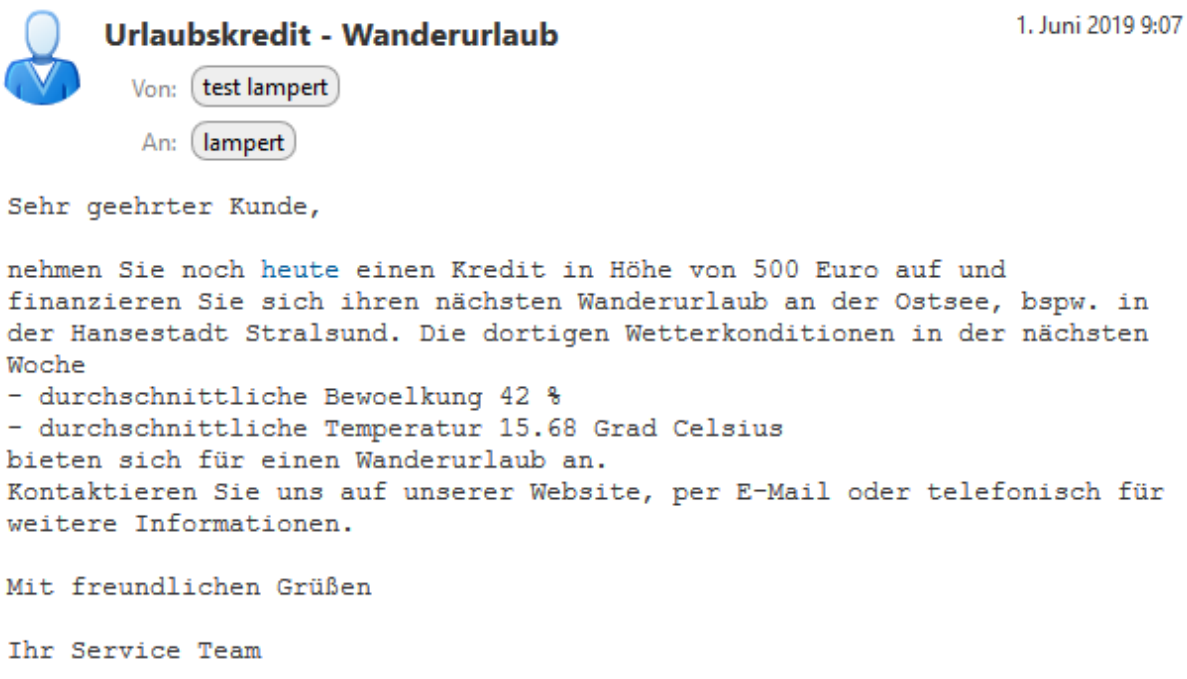
-> Kundenbewertung „Guter Kunde“



-> Der Prozess kann nicht fortfahren, da das Workflowmanagementsystem nicht mit dem Internet verbunden ist. Der Error „Keine Internetverbindung“ wird geworfen und die Ausnahmebehandlung gestartet. Nach herstellen der Internetverbindung kann der Prozess fortfahren.



-> Die Service Task „Wetterbericht abrufen“ hat von OpenWeatherMaps die Bewölkungsdaten und die Temperatur der nächsten 5 Tage abgerufen und Durchschnittswerte aus den Tagwerten gebildet. Der DMN-Prozess „Werbeprogramm entscheiden“ empfiehlt anhand der Wetterdaten einen Wanderurlaub.



Urlaubskredit - Wanderurlaub

1. Juni 2019 9:07

Von: test lampert

An: lampert

Sehr geehrter Kunde,

nehmen Sie noch heute einen Kredit in Höhe von 500 Euro auf und finanzieren Sie sich ihren nächsten Wanderurlaub an der Ostsee, bspw. in der Hansestadt Stralsund. Die dortigen Wetterkonditionen in der nächsten Woche
- durchschnittliche Bewoelkung 42 %
- durchschnittliche Temperatur 15.68 Grad Celsius
bieten sich für einen Wanderurlaub an.
Kontaktieren Sie uns auf unserer Website, per E-Mail oder telefonisch für weitere Informationen.

Mit freundlichen Grüßen

Ihr Service Team

-> Es wird eine E-Mail an das E-Mail-Postfach des Kunden gesendet, die den Werbetext und die Wetterkonditionen enthält.

Ausgewählte Elemente der technischen Implementierung

Script Task „Internetverbindung überprüfen“

Die Script Task hat das Script Format groovy. Es wird ein URL-Objekt „http://ww.google.de“ erzeugt. Anschließend wird ein HTTP-GET-Request auf diese URL durchgeführt. Ist die Internetverbindung nicht vorhanden, so wird einer der Methoden-Aufrufe im try-Block einen Fehler werfen und der catch-Block wird ausgeführt. Wenn der HTTP-GET-Request erfolgreich ist, wird die Prozessvariable „InternetConnection“ auf true gesetzt. Ansonsten, wenn der HTTP-GET-Request fehlschlägt, ist die Prozessvariable false. Die Abb. unten zeigt das groovy-Script der Script Task.

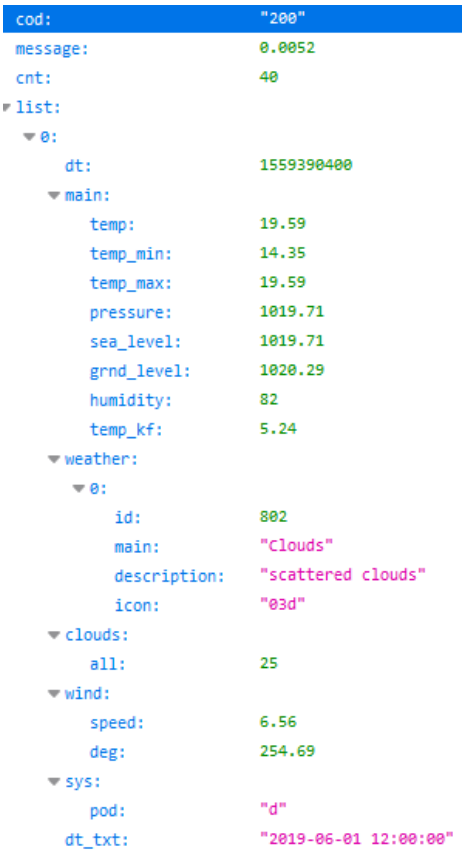
```
def url = new URL('http://www.google.de/')
sum = false
try{
  def connection = url.openConnection()
  connection.requestMethod = 'GET'
  connection.responseCode
  sum = true
}
catch(e){
  sum = false
}
execution.setVariable("InternetConnection", sum);
```

Service Task „Wetterbericht abrufen“

Mit der Folgenden URL kann eine Wettervorhersage für die nächsten 5 Tage für Stralsund abgerufen werden.

https://api.openweathermap.org/data/2.5/forecast?q=Stralsund&units=metric&appid=aValidAPIKey

Die Abfrage liefert ein json-Dokument zurück, das Wetterdaten in 3-Stunden-Intervallen beginnend bei der aktuellen abgerundeten Stunde plus 3 Stunden enthält. Die Abb. unten zeigt das erste Listen-Element des json-Dokumentes.



Die untenstehende Abb. zeigt den Quellcode der Funktion „apiRequest“, die den API-Request ausführt und die Wetterdaten des zurückgegebenen json-Dokumentes ausließt.

Mithilfe einer Schleife werden die Attribut-Werte der Attribute main.temp und clouds.all aus dem json-Dokument ausgelesen und aufsummiert. Hierbei werden nur Werte betrachtet, die zwischen 6 Uhr und 22 Uhr liegen, da die Wetterkonditionen in der Nacht für den Urlaub hier nicht betrachtet werden sollen. Nach dem Schleifendurchlauf werden die Summen durch die Anzahl der betrachteten Werte geteilt, sodass sich jeweils ein Durchschnittswert für die Temperatur und die Bewölkung ergibt. Diese Werte werden dem Prozess als Prozessvariablen dann zur Verfügung gestellt.

```java
public static WeatherData apiRequest() throws JSONException, IOException {

    //reads cloudiness and temperature from web
    String url = "https://api.openweathermap.org/data/2.5/forecast?q=Stralsund&units=metric&appid=aValidAPIKey";
    JSONObject json = readJsonFromUrl(url);
    System.out.println(json);
    JSONArray list = json.getJSONArray("list");
    int n = list.length();
    int m = 0; //count of weather vectors which time stamp is between 6 and 22 o'clock
               //we only want to consider the weather vectors which are during the day (not in the night)
    Double averageCloudiness = 0.0;
    Double averageTemp = 0.0;
    for (int i = 0; i < n; i++) {
      JSONObject listEntry = list.getJSONObject(i);
      //extract time of future
      long unixTimeStamp = listEntry.getLong("dt");
      java.util.Date dt = new java.util.Date(unixTimeStamp*1000);
      int hour = dt.getHours();
      if (hour >= 6 && hour <= 22) {
          //extract cloudiness
          JSONObject clouds = listEntry.getJSONObject("clouds");
          averageCloudiness += clouds.getDouble("all");
          //extract temperature
          JSONObject main = listEntry.getJSONObject("main");
          averageTemp += main.getDouble("temp");
          m++;
      }
    }
    averageTemp = averageTemp / m;
    averageTemp = ((double) Math.round(averageTemp*100))/100;
    int averageCloudiness_int = (int) Math.round(averageCloudiness / m);

    return new WeatherData(averageTemp, averageCloudiness_int);
}
```