1/22/2019

Documentation project Kampot Cement

ITV2H-01

Client: Kampot Cement
Projectgroup: ITV2H01

Dennis Harms	380176
Joppe Klaver	373471
Kyle Gravenhorst	376772
Maiwand Rasulzadeh	340238
Karel Koster	361252

Table of contents

Functional Design

Technical Design

1. Introduction

Kampot Cement is a company that makes Cement. Kampot Cement is located in Cambodia and needs actual weather data to decide if the conditions for making and transporting cement are right. Our company got the opportunity to make an application for Kampot Cement where they can view the weather in and around Cambodia in real time. In this functional design we will show the first draft of our application and how to use it, the technical aspect will be discussed in the technical design.

2. Requirements

After a first meeting with Kampot Cement they made clear what the application should be able to do. All requirements can be divided in 4 groups:

Must have
(the application must have these features or else it can not be used)

Should have
(The application should have these features, but they are not necessary)

Could have
(The application will have these features if all the must and should haves are already implemented)

Won't have
(The application will not have these features, but the features could be implemented in a future project)

Must have	Should have	Could have	Won't have
Data encryption			
Secured login			
Admin account			
98% accurate data	100% accurate data	View raw data.	
Graphs displaying the data. Easy to use and understand	Linegraphs and bargraphs for rainfall, humidity and temperature.		
Web application refreshes every 10 seconds	Real time updates		
Option to download data	Download data as csv file(excel). Have an option to select more or less data.		

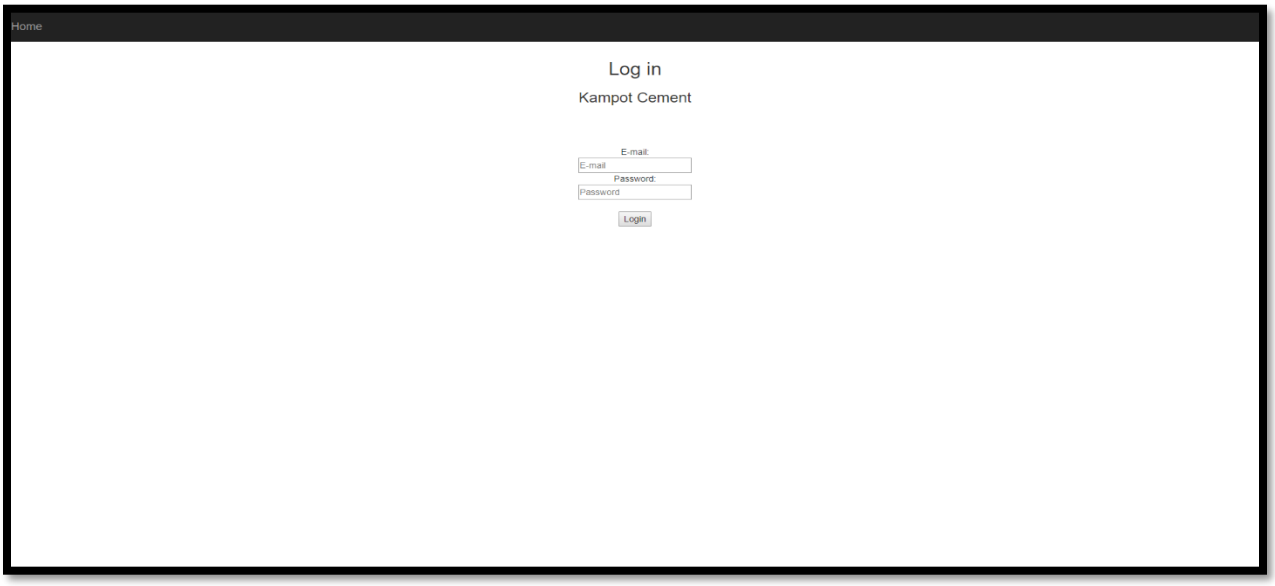
3. Web application

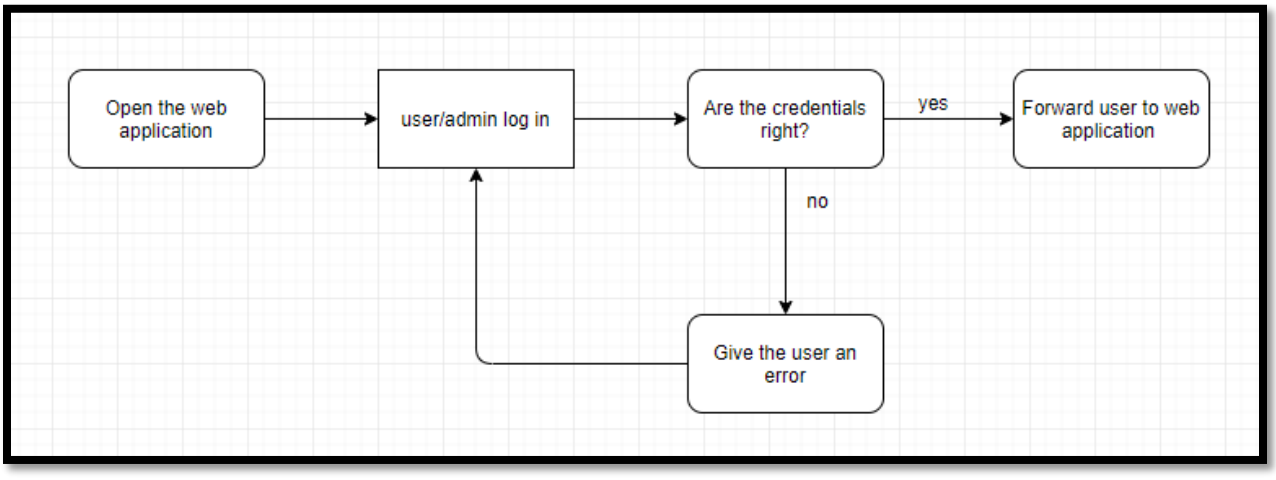
The web application will be easy to understand for the employees of Kampot Cement. To use the application they won't need to install anything new on their computer, once the application is online Kampot Cement can start using it immediately.

3.1 Login system

When going to the site, users and admins have to log in before being able to see the data. People from outside will not be able to log in to the site and see the data. Employees need to log in using their email and a password. The password needs to contain at least one uppercase letter, one lowercase letter and one number.

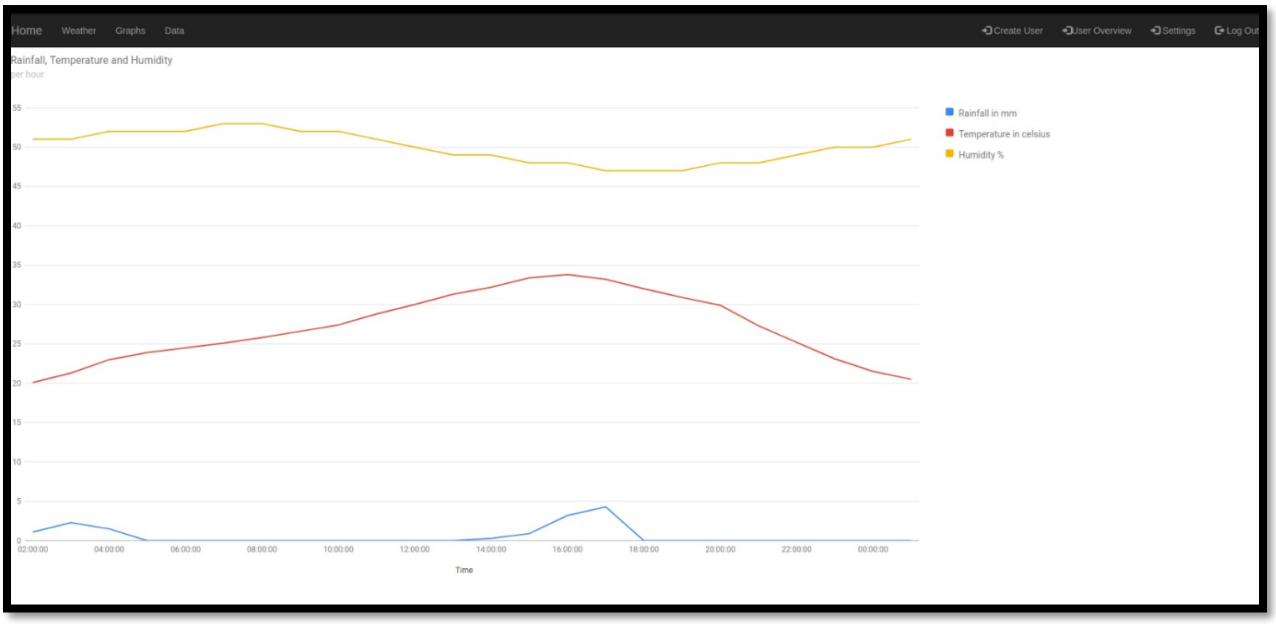
Only admins can create and delete accounts. So when a new employee needs access to the system an admin needs to create an account first. When an employee stops working for Kampot Cement admins can delete their account so they will not be able to log in to the system anymore. All passwords will be encrypted for security reasons.





3.2 Graphs

Once logged in on the website users and admins are able to see the data in graphs. The page is called graphs and will display multiple graphs containing relevant weather data. These graphs can be used by Kampot Cement to make decisions concerning the production or transport of their product.



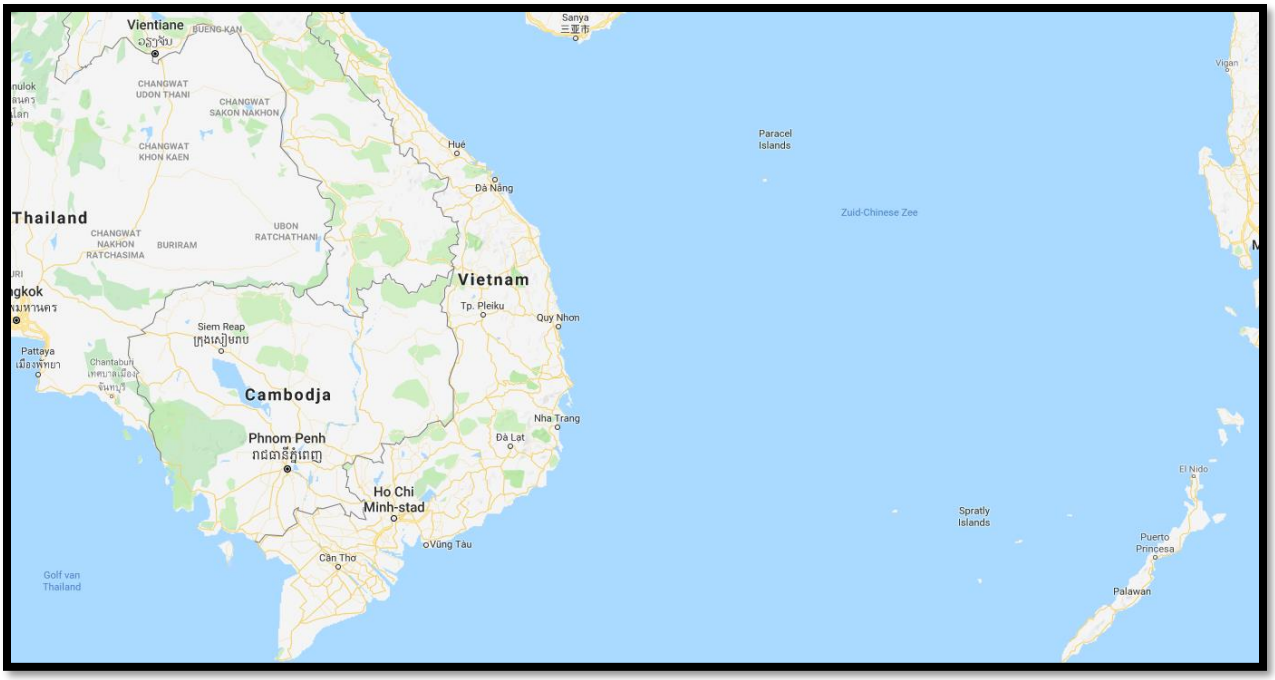
3.3 Admin privileges

There are 2 types of employees on the web application, normal users and admins. Admins can do more than normal users. The admin privileges are: creating users, deleting users, edit user info. The buttons in the menu bar on the right are only visible for admins and not for normal users.

4. Weather stations

The weather data will be collected from weather stations in and around Cambodia. We will use all weather stations in Cambodia, Laos, Thailand, Vietnam and the South Chinese Sea. All the data will be visible in the web application for Kampot Cement.

The exact area of all weather stations is:



latitude between 10.755452 and 18.673342
longitude between 100.5017651 and 120.9842195

5. Hardware and software

For the creation of the Weather Application, the undermentioned hardware and software are used.

Java Application

The Java application connects with 8000 weather stations around the world. The Java version used for this application is Java JDK 11. The application generates a .csv file. This file is used for the actual web application.

Microsoft Azure Server

The Virtual Machine runs on Linux Ubuntu. The choice for Ubuntu has been made because it suits the client's requirements best.

The data gathered by the Java application is stored on this server. The server runs inside a Microsoft Azure environment. The aforementioned .csv file is stored on this Ubuntu Server.

Raspberry Pi 3B

The Raspberry Pi runs on Raspbian. This is a Debian Linux distribution designed for the Raspberry Pi. The choice for Raspbian has been made because it is optimized to run on the Raspberry Pi, and it suits the client's requirements.

The server on the Raspberry Pi runs on Java. Specifically, Java SE Development Kit 11. This is the newest Java JDK and thus the best choice. Moreover, the client agreed that this was the best option to implement.

The Raspberry Pi is stored at one of the group's team members. This way the Raspberry Pi is accessible by using a remote connection. There are no problems regarding dynamic IP-addresses due to the fact that the Raspberry Pi can get a static ip address.

6. Reception, correction and storage of weather data

In order to receive the weather data the server uses ServerSocket to listen to a specific port. In before the weather data is actually received a list of stations the customer is interested is generated. The list is automatically generated based on location of the weather stations in the regions the customer specified and is used to filter unnecessary data. Afterwards a BlockingQueue is initialized and immediately after that a ThreadPool is created. The final step the server takes before actually accepting the connections is to pre-start all the threads in the ThreadPool to reduce the initial load.

Once the server is fully initialized all incoming connections are placed in the BlockingQueue which assigns each connection its own thread. The thread receives the XML file containing the weather data and strips unnecessary formatting before sending the data to the XML parser.

Based on the list generated while initializing the server only the data of relevant weather stations is parsed. To reduce the final file size of the collected data the decision was made to only save a measurement every five seconds, this meets the requirement of the customer by a factor of two and is done by only parsing measurements with a timestamp ending on a five (5) or zero (0). The remaining XML is parsed in to a HashMap in order to save both the key and value.

The unedited data is then send to a class which first checks if there are any missing values, if so a placeholder value is added. Once every key in the HashMap contains a value the values are compared with the previous measurements if they are available. If for example the temperature is either 20% higher or lower than the previous measurements a new measurement is derived from the past 30 measurements.

Finally the corrected data is send to a single queue. The queue works based on a locking mechanism, if the queue is available to process the information it is directly added to the queue, if it is currently processing information the weather measurements are stored and will be processed after the ongoing processing is finished.

In order to write all the measurements to the disc there is a seperate class called Thoth is created which periodically gets all data from the queue and writes it to disc. The data is stored per day, in before writing to the disc Thoth checks if there is a existing file for the current day, if so it will append the data from the queue, if not it will create a new file. New files are stored per month per year.

The Java application gathers data from the different weather stations every 10 seconds and adds this to the csv file for the current day. After each day the Java application will make a new csv file. If the Java application has received the data the file will be stored in a map with the date of that day. The same will be done with the files after a month and after a year.
Synchronisation software runs on the Raspberry Pi and the Virtual machine. This software makes sure the .csv files get sent to the virtual machine. The Virtual machine can then use these csv files to show the graphs on the website.

7. Web application

The web application was made using JavaScript, html and PHP. The data is visible in graphs that is made using CanvasJS. CanvasJS is used for visualizing data with JavaScript. Data is coming from the virtual machine and is stored as a .csv file(comma separated value). JavaScript Is used to read the .csv file and convert this raw data to a graph. Graphs will be updated every 10 seconds using the javascript function setInterval(). by using this function only the graph will be updated and the webpage does not have to be reloaded. This should improve performance and waiting times.

The webserver is protected by an SSL-certificate which means the site uses the Hypertext Transfer Protocol Secure(https). https is used for secure exchange of data so others parties cannot see the data.

To fulfil the wish of Kampot Cement the stations are filtered on longitude and latitude to show only the relevant data to Kampot Cement.

All data surrounding the login system is stored in a MYSQL database hosted on the webserver. The login system is made using PHP.