Technical Report - Project specifications

NetCafe

Course:	IES - Introdução à Engenharia de Software
Date:	Aveiro, January 24th 2022
Students:	96123: Lucius Vinicius 97606: Diogo Monteiro 97880: Camila Fonseca 97939: Tomé Carvalho
Project abstract:	Web application for management of an Internet Café franchise (featuring multiple locations) that allows for analysis of current events as well as historical data.

Table of contents:

1. Introduction

The project’s main objective is to create a complete system to store and visualize data obtained from the normal operation of an internet café.

The use of Software Engineering practices is vital. It was achieved using tools like GitHub (with GitHub Projects) and ZenHub.

A specification phase involving the definition of the concept and product architecture took place in the beginning.

2. Product concept

Vision statement

Our system is meant to be used by managers of an Internet Café franchise, as well as IT technicians. Its purpose is to monitor activity inside the multiple cafés, as well as to alert the user about important events.

The following information can be accessed:

- Café location information (temperature and humidity)
- Locations’ machines and their position in the café (blueprint)
- Machine specifications and current usage statistics
- Historical machine usage statistics
- Historical user sessions
- Automatic alarms emitted
- User information

Personas

John Smith is fifty-three years old. He has a Master's in Business Administration. He has an average degree of computer literacy.

He is currently employed as a manager at the EfacTen Internet Café franchise. His duties include analyzing data to find out important information, such as whether any locations need to be expanded, downsized or remodeled based on usage statistics.

Motivation: John would like to use a user-friendly web application that facilitates monitoring real-time events and the analysis of statistics in order to fulfill his duties with ease.

Hanna Johnson is a twenty-eight year old IT technician. She is currently employed at EfacTen. She is responsible for maintenance: monitoring the machines' performance, reporting eventual issues to her coworkers and coordinating the actions necessary to troubleshoot them.

Motivation: Hanna would like a fully-featured web application that allows her to be notified promptly of any possible issue, as well as check on previous usage statistics to help her troubleshooting.

Main scenarios

Scenario #1: John wants to check if a location is comfortable, based on its temperature and humidity, in order to assess if it's necessary to take any action. He enters the dashboard, selects the location and takes a look at the temperature and humidity values.

Scenario #2: John wants to gauge how busy a location is at the moment, to find out if it needs to be upgraded or if it should be downsized due to lack of clients. He enters the dashboard, selects the location, selects either the list or map view and checks how full it is.

Scenario #3: John wants to check how busy each machine was throughout a time period, to find out which type of machines have higher demand (i.e., high-end gaming computers or workstations). John enters the history page, selects the location, selects a statistic such as power usage, and analyzes how often each machine was used.

Scenario #4: Hanna wants to check alarm notifications in order to perform her maintenance duties. She clicks the alerts button in the navbar and takes a glance at the recent alarms. She may dismiss some she doesn't find concerning, or access the notifications page through the pop-up to get a bigger view and investigate further.

Scenario #5: Hanna wants to check the previous CPU usage and temperature statistics of a machine after receiving an alarm about its temperature being too high, in order to find out if it is heating up too much without a valid cause and is thus malfunctioning. After checking the alarm, she navigates to the history page, selects the location and machine, selects CPU usage and then CPU temperature to make her comparison.

3. Architecture notebook

Key requirements and constraints

<Identify issues that will drive the choices for the architecture such as: Will the system be driven by complex deployment concerns, adapting to legacy systems, or performance issues? Does it need to be robust for long-term maintenance?

The architecture choice is based on the followings criterias:

- The system needs to be capable of generating and consuming data automatically, simulating what would happen in a real environment.
- A web app should provide user-friendly visualization of the API's contents
- The API data should be able to be used by other projects, so it should have methods independent from the current Web Service.
- The system uses a Data Generator for informing about the usage for machines.
- Locations and Machines' usages must be available in the Web Service.

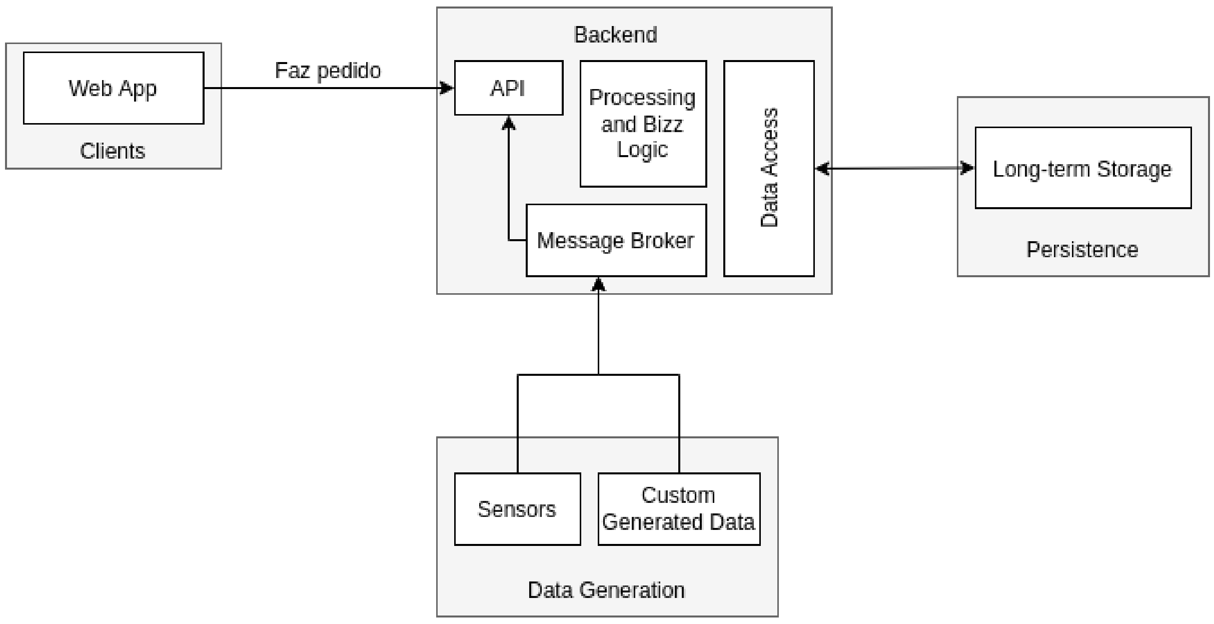
With those criteria in mind, there are some critical issues that need to be satisfied:

- Despite sending data to the same message broker, the data provided by the sensors and generator should never be mixed up on the database.
- The Message Broker must connect the Sensors and the Data Generator to the backend.

The sensors used need to be in a place with access to the internet to be capable of reporting their current status.

Architectural view

Diagram



→ Discuss architecture planned for the software solution.

→ include a diagram

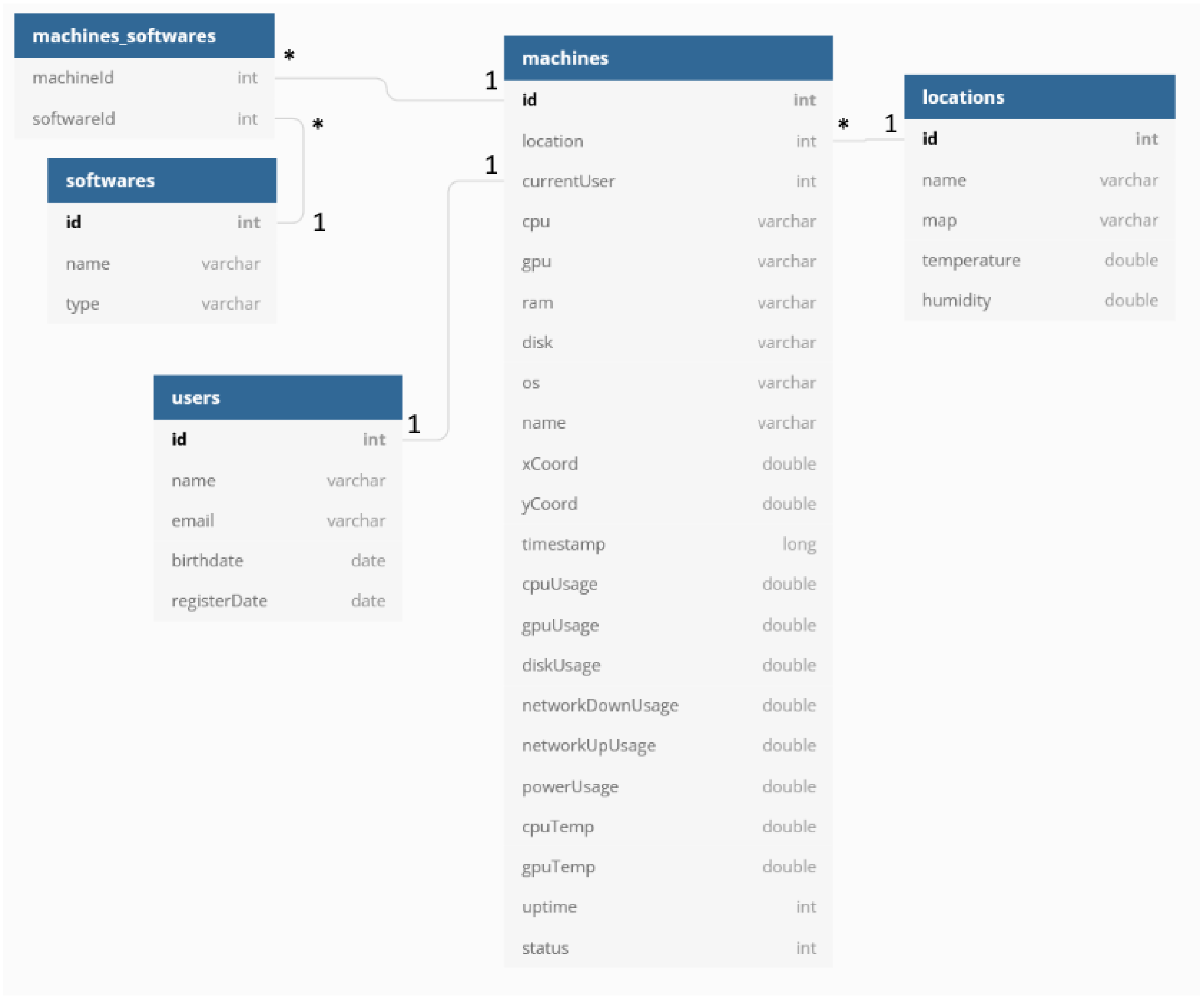
Module interactions

→ explain how the identified modules will interact. Use sequence diagrams to clarify the interactions along time, when needed

→ discuss more advanced app design issues: integration with Internet-based external services, data synchronization strategy, distributed workflows, push notifications mechanism, distribution of updates to distributed devices, etc.>

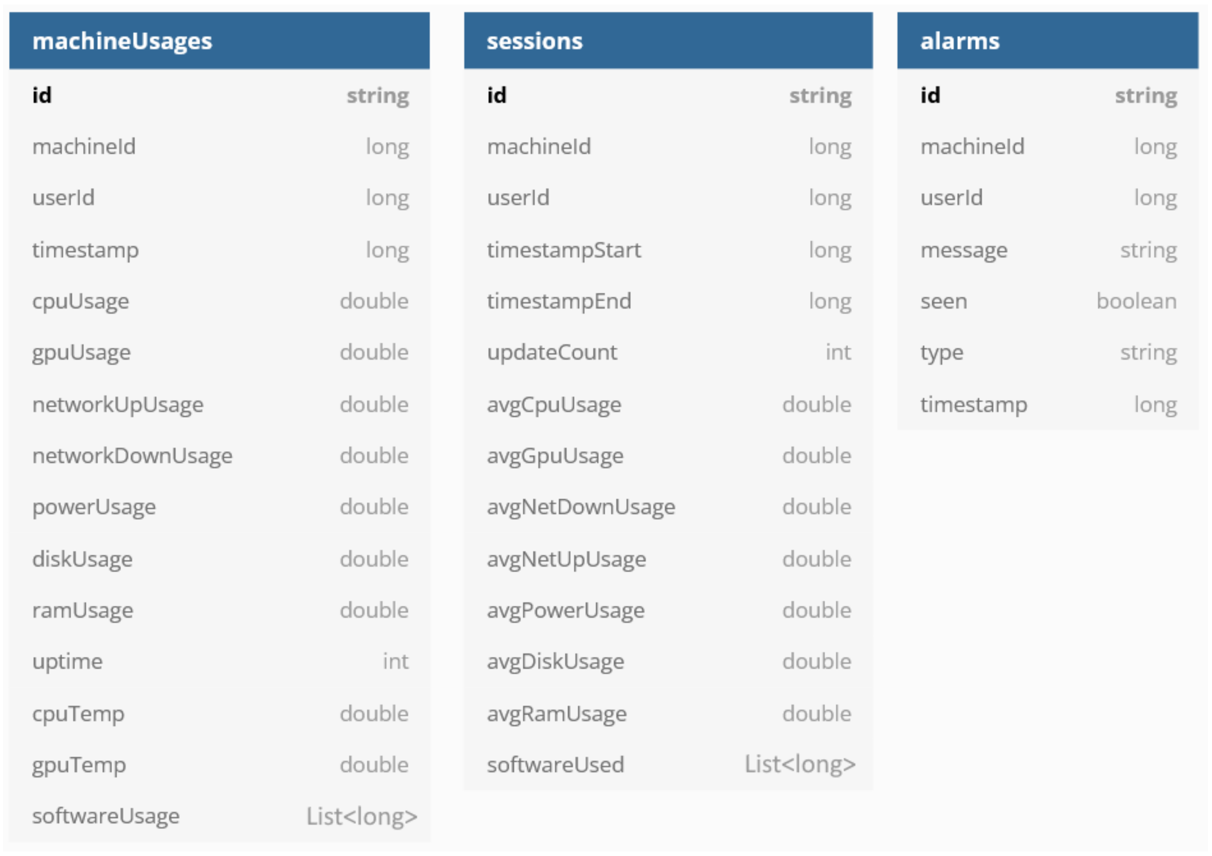
4. Information perspective

MySQL



The machines table contains not only information about the machine itself but also about its current usage (every attribute from timestamp below, including the former). Multiple software can be in use by a machine. One (and only one) user may be using the machine. The machine belongs to a location.

MongoDB



machineUsages	
id	**string**
machineId	long
userId	long
timestamp	long
cpuUsage	double
gpuUsage	double
networkUpUsage	double
networkDownUsage	double
powerUsage	double
diskUsage	double
ramUsage	double
uptime	int
cpuTemp	double
gpuTemp	double
softwareUsage	List<long>

sessions	
id	**string**
machineId	long
userId	long
timestampStart	long
timestampEnd	long
updateCount	int
avgCpuUsage	double
avgGpuUsage	double
avgNetDownUsage	double
avgNetUpUsage	double
avgPowerUsage	double
avgDiskUsage	double
avgRamUsage	double
softwareUsed	List<long>

alarms	
id	**string**
machineId	long
userId	long
message	string
seen	boolean
type	string
timestamp	long

machineId: machines.id in MySQL

userId: users.id in MySQL

softwareUsage / softwareUsed: List of softwares.id in MySQL

MachineUsage represents a snapshot of a machine’s usage information at a certain timestamp.

Sessions are created by the “merging” of usages. They refer to an uninterrupted usage period of a machine by a user. timestampStart represents the beginning of this period, while timestampEnd represents the end. The numerical statistics are averaged (the reason why we need the updateCount field, which essentially counts the number of MachineUsages used to create the Session), the softwareUsed is the list of software used throughout the session. A session is considered ongoing if its timestampEnd is null.

Alarms are associated with a machine and a user. They contain a message. The type field represents the gravity of the alarm. The seen attribute allows us to mark them as seen/unseen.

5. References and resources

<document the key components (e.g.: libraries, web services) or key references (e.g.: blog post) used that were really helpful and certainly would help other students pursuing a similar work>

Springdoc - to generate Swagger UI documentation
Used instead of the Open/API Swagger framework linked in the project guidelines, as the latter was troublesome in recent Spring Boot versions.