Entangled Savage

Architecture document

# Entangled savage is a software system designed for hospitals and other health institutions management. Designed primarily with healthcare in mind, it covers all the aspects of a hospital related to healthcare. Nevertheless, the data it generates can be exported for use in other applications.

# A HIGH-LEVEL DESCRIPTION OF THE ARCHITECUTURE

Entangled savage is designed to help hospitals have a unique datasheet that all hospitals can consult, provided they have access to it. Countries laws apply here as to the ownership of the data sheet and as to whom has the right to authorize an entity to view or edit patient health data.

The software system is also designed with Governments in mind. The system provides governments with statistical data on patients, diseases and more. The following diagram explains how each of the user type interacts with the system.

**Healthcare User**

**[Organization / Person]**

A hospital or healthcare professional

**Patient**

**[Person]**

An organization supplying products.

**Government**

**[Organization]**

Government

**Entangled Savage**

**[Software System]**

On Premises

**Entangled Savage**

**[Software System]**

PaaS

Creates and edit patient data.

Views owned data. Request or cancels appointments

Views statistical data

## Rationale

**The choice of having an On-Premises and a PaaS** version of the same application is driven by the deployability quality attribute which states that we should have these two separate deployment types. The reason for this is that in some country, especially in the Democratic Republic of Congo, where the software is first intended to be deployed, a fast internet connection can come at a very high cost and not every health institution will be able to afford the usage cost of a PaaS deployment. For these types of clients, we should offer an On-Premises deployment that will periodically sync its cashed data with the cloud database.

**The choice of having a government type of user** is guided by the functional requirements as exposed in the vision and scope document. This choice provides the Government with statistical data about hospitals, patients diseases, medicines and more. This choice is also driven by the business goal of the application and the organization Recycling[[1]](#footnote-1).

**The choice of having a patient type of user**. Ultimately, the datasheet belongs to the patient according to healthcare laws in the Democratic Republic of Congo. This might be the case in other countries but as of now this choice satisfies the judicial constraints of the first geographic area targeted by the solution. In addition to this, patients will be offered the ability of requesting or canceling appointments with doctors or health institutions altogether, be notified of updates regarding a process he might be involved in with an health institution and a few other functionalities.

## Alternative considered

|  |  |
| --- | --- |
| Area | Alternatives |
| Deployment type | * On-Premises only * PaaS only |

# DESCRIPTION OF THE DEOPLOYMENT

Entangled Savage is designed for cloud. Mainly for cost efficiency and performance. Both cannot be achieved with local deployment in the Democratic Republic of Congo in which the cost of having a private cloud will be exorbitant let alone the performance and cost issues related to internet. These constraints have driven some of the architectural choice for Entangled Savage. The following section does not make a difference between the On-Premises and the PaaS version. Rather it focuses on the common architectural aspects of both deployment types. In the end the only difference there will be between the two deployments is the environment and perhaps the physical locations of the deployment. One being a remote cloud and another being local servers in a healthcare institution facility. Added to this, there might be limited computational resources with an On-Premises installation. The differences stop there. From this point on, the document will treat the two deployment types as one except when explicitly stated that one configuration or functionality belongs to one deployment type or another.

**IMPORTANT**

In the following document, each section describes a precise aspect of the architecture. A C4 or UML diagram may follow to illustrate the aspect under consideration. The aspect under consideration may also be backed up by a list of explanation on architecture choices made and a list of alternatives to those choices that were considered but not selected.

The following diagram gives a very high level of the deployment.

**User**

**[Organization / Person]**

An organization or an household

**UI Service**

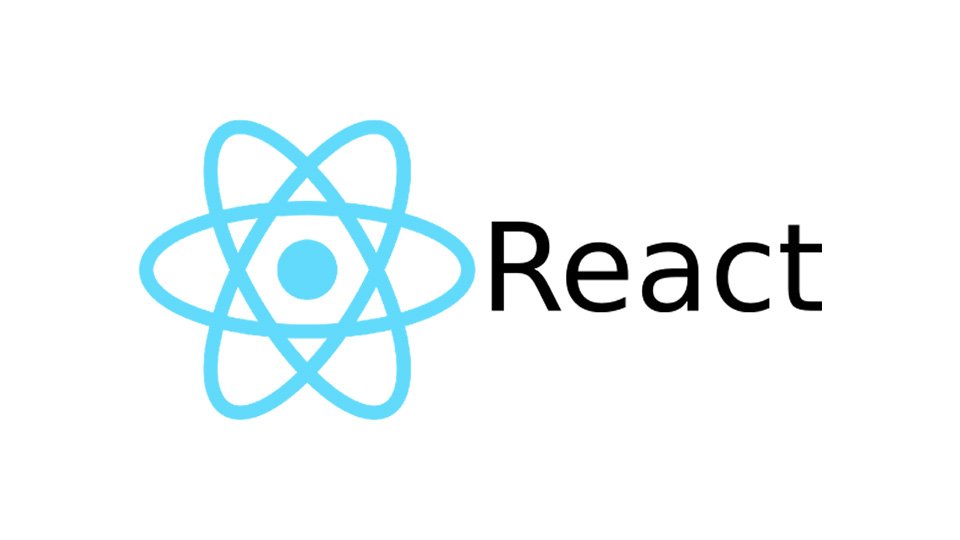
[Container: nodeJS]

****

Delivers the single page web application and other static contents.

**Single-Page Application**

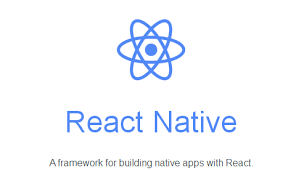
[Container: node/reactJS]

****

Provides the healthcare functionalities via the web browser.

**Mobile App**

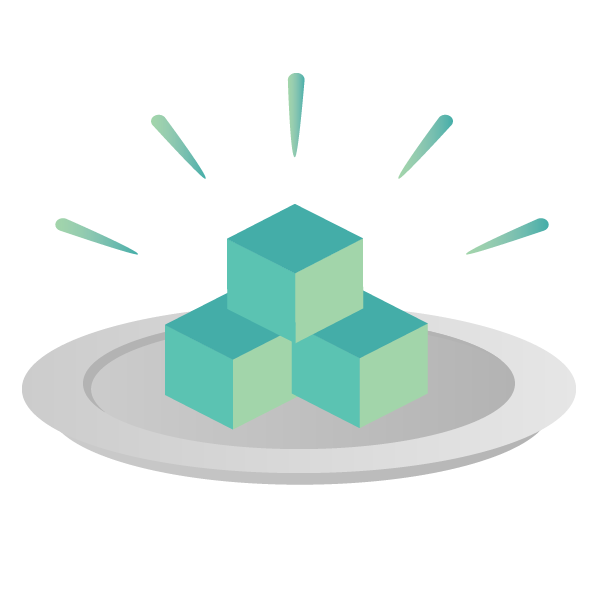
[Container: React Native]



Provides a subset of the functionalities offered by the Single-Page application.

**Application API**

[Container: CaaS]



Contains instances of all microservices of the software system.

**Uses**

[HTTPS]

**Uses**

**Uses**

**Delivers**

**Uses**

[JSON/HTTPS]

**Uses**

[JSON/HTTPS]

**Supply Chain Software**

[Software System]

## Rationale

**The choice of the React stack** is dictated by the constraint of having the whole team of developers mastering the React technology alone at the moment of the writing of this document.

**The choice of CaaS** is dictated by the deployability quality attribute which wants the deliverable not to be bound to any particular cloud provider. Choosing a CaaS allows us to retarget the cloud provider more easily. It is also a good choice for On-Premises deployments because even there we want to maintain a high availability.

## Alternative considered

|  |  |
| --- | --- |
| Area | Alternatives |
| UI frameworks | * Angular |
| Deployment platform type | * PaaS |

# DESCRIPTION OF THE APPLICATION API

An analysis of a hospital system helped define 5 major domains. The following table gives a summary of them.

|  |  |
| --- | --- |
|  | Alternatives |
| Hospital | Responsible for patient consultation, vital sign gatherings, diagnostic establishment and prescriptions. |
| Accounting | Responsible for the accounting of a healthcare institution. |
| Laboratory | Responsible for examining patient samples and making tests on them. |
| Pharmacy | Responsible for storing medical goods for patients and for some of the other components of a healthcare institution. |
| Statistics | Responsible for gathering statistical data from the other components of a healthcare institution. |

## Collaboration diagram

**Hospital**

[module]

Patient consultation, vital sign gatherings, diagnostics establishment and prescriptions.

**Accounting**

[module]

Patient consultation, vital sign gatherings, diagnostics establishment and prescriptions.

**Laboratory**

[module]

Patient consultation, vital sign gatherings, diagnostics establishment and prescriptions.

**Pharmacy**

[module]

Patient consultation, vital sign gatherings, diagnostics establishment and prescriptions.

**Statistics**

[module]

Patient consultation, vital sign gatherings, diagnostics establishment and prescriptions.

The flow of data between the module is highly customizable following the configurability quality attribute. However, in a typical configuration these are the main exchanges between the modules:

* The statistics module fetches data from all the other modules and makes calculations on them.
* The accounting module fetches data from the hospital, laboratory and pharmacy modules. Orders that are submitted to every other module are paid using this module.
* The hospital module communicates with the pharmacy module to order medical items and with the laboratory module to order tests.
* The laboratory module communicates with the pharmacy module to order medical items.

# MODULES

Module in this document means a logical grouping of related functionalities. They are implemented as a service or a set of services.

## Rationale

The choice of a Microservices architecture is dictated by the business need of having one MVP as fast as possible that ca be demoed to investors. Having the application split into microservices means that one or two services can be pushed to an acceptable completeness without having the whole application in place. This choice is also dictated by the desire to have each developer taking ownership of at least one business domain of the application and its related services.

An analysis of healthcare institutions in the Democratic Republic of the Congo helped us draw the seams which are called modules and are defined in pervious sections of this document. In addition to them, there is the Store service which has been drawn. It’s a virtual store in the system and it sells items which cannot be sold by the pharmacy module. Virtual items like a consultation, a laboratory test etc. This service acts as a proxy between some of the services and the accounting module in that those service do not order things from the accounting module but from the store module which will use the accounting module to process payments accordingly.

## The Hospital Module

This module is responsible for patient consultation, vital signs gathering and for determining diagnostics. Communicates with the store module to order virtual items and with the pharmacy module to order medical items.

The Hospital module is the core of the healthcare system providing many of the most important features of the healthcare process.

The following is a list of operation that the hospital module supports:

* Management of patient data sheets (creation, deletion, lookup and modification).
* Vital signs gathering
* Consultation management
* Diagnostic management

The datatypes identified form examining the process handled by the hospital module are:

* Patient
* Doctor
* Consultation
* Health check
* Diagnostic
* Disease
* Disease Case
* Prescription
* Appointment
* Laboratory test result

The following are a set of tables that describe the interface of the Hospital module.

### Patients

|  |  |
| --- | --- |
| Endpoint | Description |
| GET /patients | Gets patient with pagination |
| POST /patients | Create a new patient record. |
| PUT /patients/:id | Updates a patient record. |
| DELETE /patients/:id | Deletes a patient record. |
| GET /patient/:id | Get a patient record. |
| POST /search/patients | Search for patient |

### Consultations

|  |  |
| --- | --- |
| Endpoint | Description |
| POST /patients/:id/consultation | Creates a consultation case. |
| PUT /patients/:id/consultation/:id | Updates a consultation case |
| DELETE /patients/:id/consultation/:id | Deletes a consultation case |
| GET /patients/:id/consultation/:id | Gets a consultation |
| POST /search/consultations | Search for consultations |

### Vital signs

|  |  |
| --- | --- |
| Endpoint | Description |
| POST /patients/:id/healthcheck | Creates healtcheck data |
| GET /patients/:id/healthcheck/:id | Gets healthcheck data |
| PUT /patients/:id/healthcheck/:id | Modifies healthcheck data |
| DELETE /patients/:id/healthcheck/:id | Deletes healthcheck data |

### Diagnostics

|  |  |
| --- | --- |
| Endpoint | Description |
| POST /patients/:id/diagnostic | Creates a diagnostic |
| GET /patients/:id/diagnostic/:id | Gets a diagnostic |
| PUT /patients/:id/diagnostic/:id | Modifies a diagnostic |
| DELETE /patients/:id/diagnostic/:id | Deletes a diagnostic |
| POST /search/diagnostics | Search for diagnostics |

## Rules

A disease case is an occurrence of a disease. It is an association of a patient and a disease. A diagnostic is always made for a patient and may concern one or many diseases but one doctor. A consultation relates always to one patient and one doctor. A health check relates to one patient.

A patient data sheet is a collection of all data related to that patient. This data includes his consultation cases, his diagnostics / disease cases, his allergies etc. This sheet will also include physical data like height and racial type.

The following the standard sequence that the hospital module supports.[[2]](#footnote-2)

|  |  |  |
| --- | --- | --- |
| Sequence | | Description |
| 1. Lookup of patient datasheet | | A patient presents himself to a hospital and his datasheet is looked up. |
| 1. Ordering consultation | | The virtual store is contacted and an order is placed for a consultation. The order will be marked as unpaid and will appear in the accounting module. |
| 1. Payment for the consultation | | The consultation order is paid for and the datasheet is available for vital signs gathering. |
| 1. Ordering vital signs gathering | | The virtual store is contacted and an order is placed for vital signs gathering. This order is almost always free in the majority of hospitals. |
| 1. Payment for the vital signs gathering | | Usually no action is required here but making the datasheet available for consultation. |
| 1. Consultation takes place | | The consulting doctor will record any complaint, symptoms and notes into the consultation case. |
| 1. The doctor… | Redirect the patient to another doctor, jump to step 2. | The consulting doctor may redirect the patient to another doctor in which case the process goes back to step 2. |
| Orders laboratory tests | The consulting doctor places an order for laboratory tests through the virtual store. The patient will then have to confirm the order by paying it. It belongs to each hospital to state for how long an order will remain unconfirmed. The consultation case remains opened until the laboratory modules makes the results available. The process goes back to step 2 or 6 depending on the hospital established procedure. |
| Emits a prescription | The emits a prescription which is saved and related to the disease case. |
| Schedules an appointment | The doctor schedules an appointment with the patient and the patient is notified. |
| Emits a diagnostic | The doctor emits a diagnostic which indicates one or more disease cases. |
| 8. The process ends | | The consultation case is closed. |

1. Architecture details specific to each type of deployment are presented in later parts of this document. [↑](#footnote-ref-1)
2. The procedure may slightly vary from hospital to hospital hence the configurability quality attribute scenario that states that each hospital should be free to configure processes and data for their specific needs. [↑](#footnote-ref-2)