Software Architecture
Assignment 1b

Course: X_400170 Software Architecture 2021-2022
Group: 7 - Cluster A (on campus)
TA: Vlad Stan

Full name	Student nr.	Master program	Email address
Bogdan Andrei	2737546	Computer Science	b.andrei@student.vu.nl
Latife Dağ	2736983	Computer Science	l.dag@student.vu.nl
Thomas Dingemanse	2713755	Computer Science	t.dingemanse@student.vu.nl
Rumy Krumova	2718449	Artificial Intelligence	r.krumova@student.vu.nl
Balázs Szabó	2745953	Information Science	b.szabo@student.vu.nl

Date of submission: December 24, 2021

Contents

Revision history

Nr.	Date	Week	Description
1.	30/10/2021	W1 – Stakeholders and preliminary architecture	1. The stakeholders are determined. 2. The structure of the document is created. 3. The stakeholder profiles are described. 4. A sketch of the architecture based on the stakeholder roles and the case descriptions is drawn.
2.	12/11/2021	W2 – Business goals and ASRs	More details are given for the business goals of the stakeholders and the architecturally significant requirements (ASRs) are determined.
3.	19/11/2021	W3 - Viewpoints	Viewpoints which define the modeling approaches and visual elements that should be utilized to represent and connect terms are determined and briefly described.
4.	26/11/2021	W4 - Prepare the first submission	All improvements made so far have been edited and transferred to Overleaf.

Group responsibilities

Section	Team member(s)
1.1. Stakeholder profiles - Healthcare providers	Thomas
1.2. Stakeholder profiles - Informal caregivers	Bogdan
1.3. Stakeholder profiles - Patients	Rumy
1.4. Stakeholder profiles - Payers / insurers	Balázs
1.5. Stakeholder profiles - Privacy protection agency	Latife
2. Architecture sketch	Bogdan
3. Viewpoints metamodels	Bogdan
3. Viewpoints Notation	Thomas and Rumy
3.1. Viewpoint - Patient privacy	Bogdan
3.2. Viewpoint - Healthcare administration	Bogdan
3.3. Viewpoint - Healthcare insurance and payment capability	Thomas and Rumy
4.1. View - HealthCare not receiving payment for patient's medical bill	Bogdan
4.2. View - HealthCare detecting security intrusion attempts	Balázs
4.3. View - HealthCare administration providing reasearch cases to third parties	Rumy
5. Mapping between views	Balázs
6.1. Design decisions and rationale - Design decisions and rationale for the first View	Bogdan
6.2. Design decisions and rationale - Design decisions and rationale in case of failure	Bogdan
7.1. Assessment - Assesment for Informal caregivers	Bogdan
7.2. Assessment - Assesment for Healthcare providers	Thomas
7.3. Assessment - Assesment for Patients	Rumy
7.4. Assessment - Assesment for Payers/Insurers	Balázs
7.5. Assessment - Assesment for Privacy protection agency	Latife
8. Glossary	Thomas, Bogdan

1 Stakeholder profiles

1.1 Healthcare providers

Stakeholder	Healthcare providers
Stakeholder ID	SH1-HP
Description	The healthcare providers are institutions that provide healthcare to patients. This includes hospitals, but also other healthcare providers such as pharmacies, dentists, private clinics, elderly care institutions, and home care organizations.
Expectations	1. Healthcare providers expect software that is easy to use for their employees and supports their existing workflow. 2. They expect the system to reduce their employees' workload rather than to increase it. 3. Modular systems are highly preferred, because healthcare providers want to make their own decisions about what they want to use and when to use products from other vendors. 4. The systems provided should support all use cases / care paths that the healthcare provider offers. 5. Bugs and other technical issues can occur, but they are relatively rare.
Demands	1. The system should provide value to the healthcare provider that is at least proportional to the expenses required to adopt the system. In other words: it should either cut costs or increase revenue to a degree that is equal or higher than the expenses paid. This can include non-monetary value that contributes to increased revenue or lower costs in the long run, such as increasing employee or client satisfaction or having better health outcomes for patients. 2. The system should be able to interoperate with other systems that are commonly used. For example, there should be APIs to connect third party software like an appointment system, software from the insurance company or a database of a patient's medical records. 3. Technical support should be provided for all software in use.
Business goals	1. For the HealthSuite, healthcare providers desire that using the HealthSuite either increases revenue or decreases costs compared to the current situation by at least 5%. See figure 1. 2. For the HealthSuite, healthcare providers desire to make care processes take at least 10% less time than they do now. 3. For the HealthSuite, healthcare providers desire to not have the HealthSuite cause any decrease at all in the number of patients that can be treated per month. See figure 2. 4. For the HealthSuite, healthcare providers desire to not have the HealthSuite cause any decrease at all in reported health outcomes by patients. See figure 3. 5. For the HealthSuite, healthcare providers desire to not have the HealthSuite cause any reduction at all in their reputation compared to the current situation, measured by third party evaluation services and internal client satisfaction/opinion ratings. See figure 4. 6. For the HealthSuite, healthcare providers desire that using the HealthSuite increases employee satisfaction by at least 10%, based on internal employee satisfaction questionnaires/polls. See figure 5.

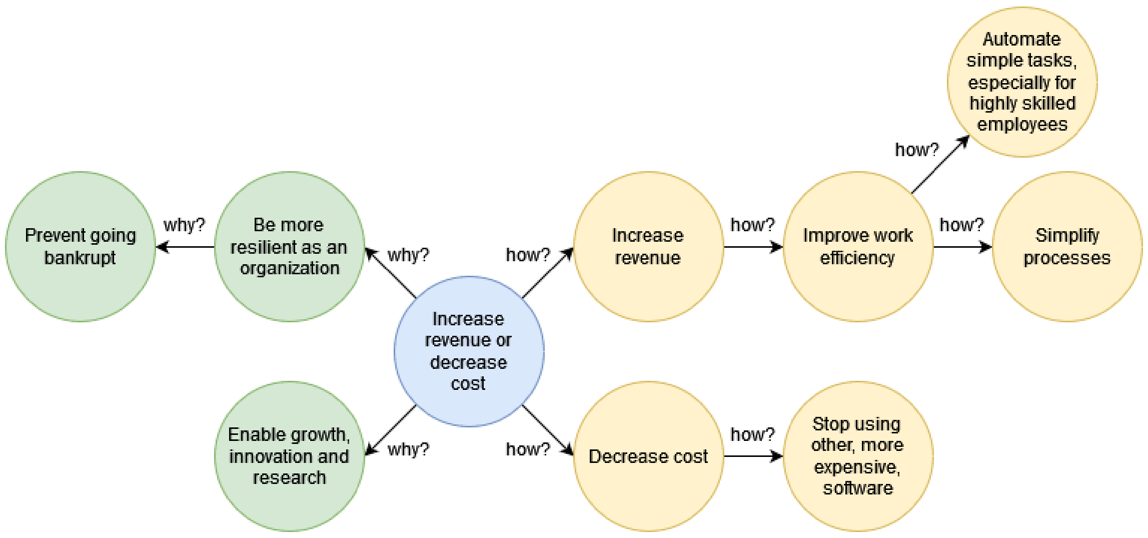

Prevent going bankrupt
why?
Be more resilient as an organization
why?
Enable growth, innovation and research
why?
Increase revenue or decrease cost
how?
Increase revenue
how?
Improve work efficiency
how?
Automate simple tasks, especially for highly skilled employees
how?
Simplify processes
how?
Decrease cost
how?
Stop using other, more expensive, software

Figure 1: Goal-oriented requirements graph for business goal 1: increase revenue or decrease cost.

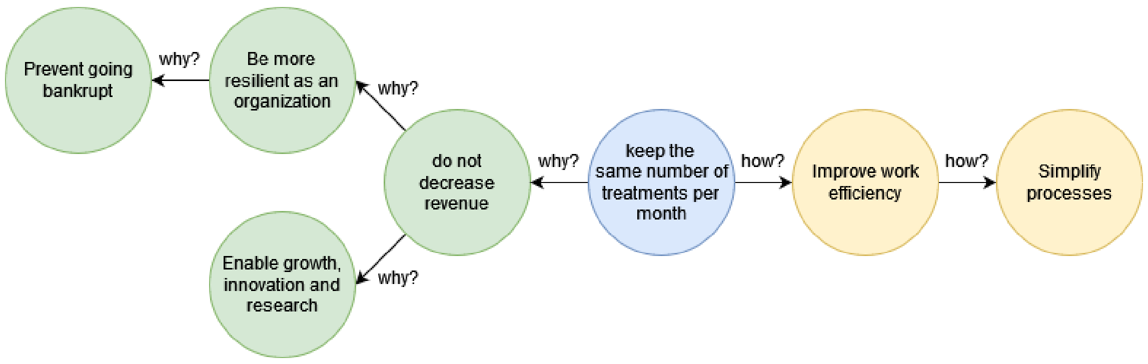

Prevent going bankrupt
why?
Be more resilient as an organization
why?
do not decrease revenue
why?
Enable growth, innovation and research
why?
keep the same number of treatments per month
how?
Improve work efficiency
how?
Simplify processes

Figure 2: Goal-oriented requirements graph for business goal 3: keep the same number of treatments per month.

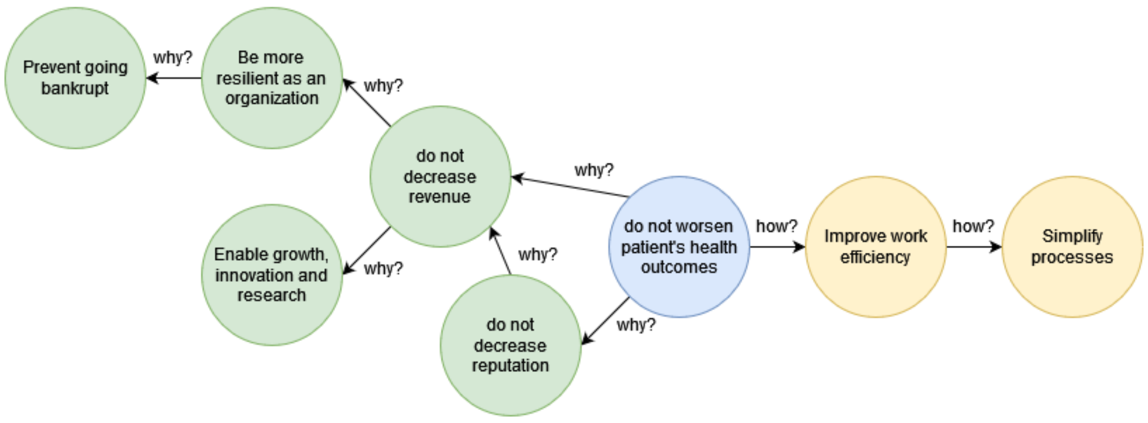

Prevent going bankrupt
why?
Be more resilient as an organization
why?
do not decrease revenue
why?
Enable growth, innovation and research
why?
why?
do not worsen patient's health outcomes
why?
do not decrease reputation
how?
Improve work efficiency
how?
Simplify processes

Figure 3: Goal-oriented requirements graph for business goal 4: don't worsen patients' health outcomes.

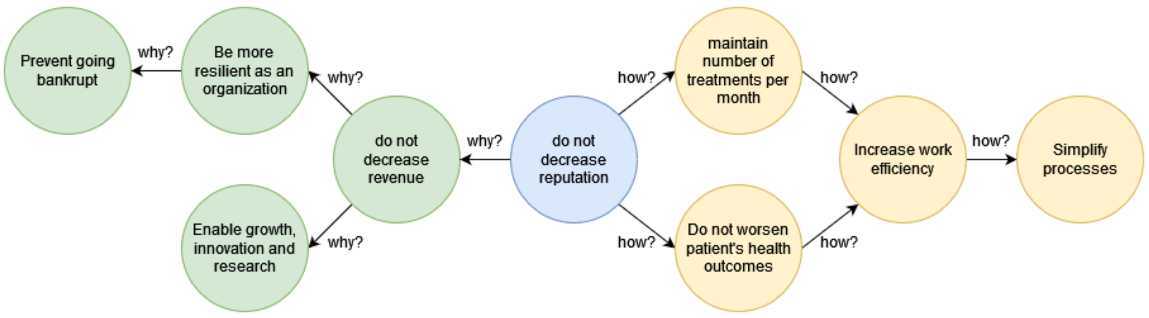

Prevent going bankrupt
why?
Be more resilient as an organization
why?
do not decrease revenue
why?
Enable growth, innovation and research
why?
do not decrease reputation
how?
maintain number of treatments per month
how?
Do not worsen patient's health outcomes
how?
how?
Increase work efficiency
how?
Simplify processes

Figure 4: Goal-oriented requirements graph for business goal 5: don't decrease reputation.

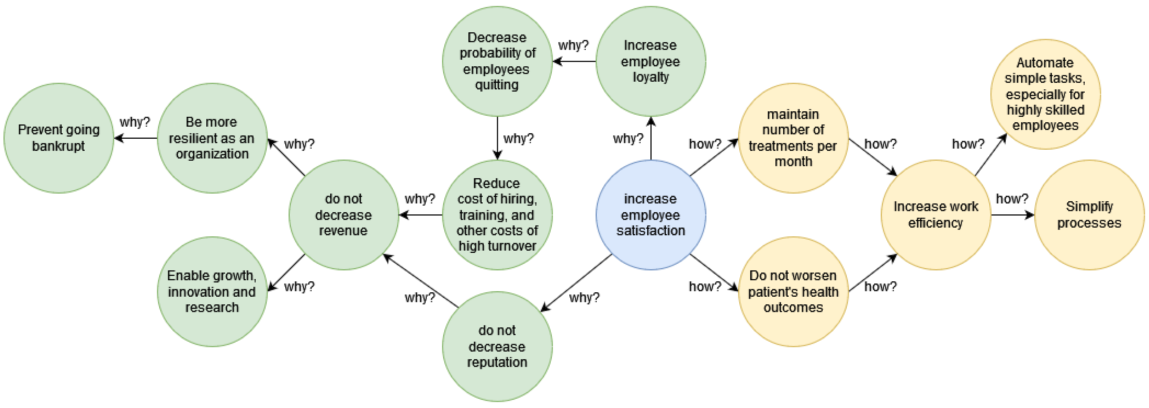

Prevent going bankrupt
why?
Be more resilient as an organization
why?
do not decrease revenue
why?
Enable growth, innovation and research
Decrease probability of employees quitting
why?
Increase employee loyalty
why?
Reduce cost of hiring, training, and other costs of high turnover
why?
why?
increase employee satisfaction
why?
do not decrease reputation
how?
maintain number of treatments per month
how?
Do not worsen patient's health outcomes
how?
how?
Increase work efficiency
how?
Automate simple tasks, especially for highly skilled employees
how?
Simplify processes

Figure 5: Goal-oriented requirements graph for business goal 6: increase employee satisfaction.

1.1.1 ASR 1: Accessibility

Portion of Scenario	Possible values
Source	Employee of a healthcare provider, such as a doctor, nurse, radiologist, receptionist.
Stimulus	Employee tries to: 1. Access the system on their personal devices such as: tablets, smartphones, laptops, etc. with varying specifications, screen size, internet speed, etc. 2. Find information easily, such as a patient's medical records. 3. Provide information easily, such as diagnoses, test results, summaries of meetings and consultations, contact details of patients and informal caregivers, etc. 4. Navigate and use the different features of the HealthSuite without being confused or overwhelmed.
Environment	At runtime
Artifacts	System (the HealthSuite)
Response	Provide optimized web app and/or mobile app, anticipate user's needs by studying current workflows and user behavior.
Response measure	User satisfaction measured by user surveys and by standard performance and usability metrics for web and mobile apps such as time to first paint.

1.1.2 ASR 2: Modifiability

Portion of Scenario	Possible values
Source	Employee of a healthcare provider, such as a doctor, nurse, radiologist, receptionist.
Stimulus	Employee tries to perform some action that was not originally implemented, such as using a new kind of medical scanner and storing the results in the system.
Environment	At runtime, at configuration time
Artifacts	System (the HealthSuite)
Response	The system does not support this yet, but technical support is ready to accept feature requests that can be implemented by the development team for a certain price and with a certain timeframe provided to the healthcare provider, given that the healthcare provider approves it's employee's request.
Response measure	Various measures described in the SLA (service level agreement), such as the time taken to implement new features compared to the time promised, or the time before new feature request tickets get a response from technical support.

1.1.3 ASR 3: Availability

Portion of Scenario	Possible values
Source	Employee of a healthcare provider, such as a doctor, nurse, radiologist, receptionist.
Stimulus	Employee tries to access the system
Environment	At runtime
Artifacts	System (the HealthSuite)
Response	The system should be online and usable, so it should respond to any actions the employee wants to perform. If some component of the system has failed, the rest of the system should still remain operational wherever possible.
Response measure	Uptime for each of the systems' components and for the system as a whole. Both are measured using the common uptime measure "nines of uptime" (four nines = 99.99%).

1.2 Informal caregivers

Stakeholder	Informal caregivers
Stakeholder ID	SH2-IC
Description	Informal caregivers are also called family caregivers. They are people who give care to family or friends, usually without payment. They provide personal care, monitor medication, or help with practical tasks such as shopping, laundry, or cleaning. Informal caregivers are important to the HealthSuite architecture because they can provide feedback about the overall performance of the software and they could have ideas about how to improve the software, for example a new feature. The architecture can be changed based on their feedback. Additionally, informal caregivers might need to communicate and exchange information with formal caregivers to optimize the way care can be given to patients. For instance, a doctor might provide information about required medications to family members of a patient who can get better care that way.

Expectations	1. The HealthSuite software is easy to use and intuitive. 2. Guidance and 24/7 support on how to use the software is provided by either the healthcare provider, or by the developer of the system. 3. The HealthSuite is a a reliable platform, without any errors. 4. The HealthSuite has a feature that helps them to monitor medication.
Demands	1. To be informed about the patient's status either by doctors, nurses or by medical devices for example: being able to see their patient's heart rate when they take a pulse oximeter test. 2. To communicate with formal caregivers. 3. To view data about that patient anytime, anywhere, of course, with the patient's approval. If the patient is unable to explicitly give approval, informal caregivers still demand access to this data so that they can help their friend or family member as much as possible.
Business goals	1. For the HealthSuite system, the informal caregivers desire that they can access their patient's data fast, in a reliable way, without errors. See figure 6. 2. They desire from the HealthSuite system to have a stable platform to communicate with formal caregivers, nurses or doctors. See figure 7. 3. They also desire that the HealthSuite system will be accessible on personal devices such as: smartphones, laptops, no matter the specifications or performance of their devices. See figure 8.

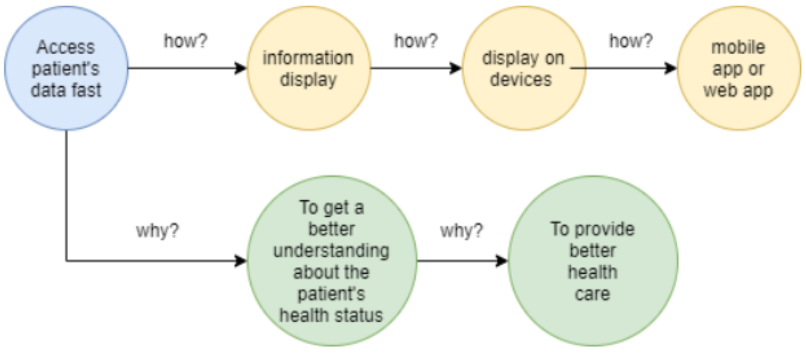

Access patient's data fast
how?
information display
how?
display on devices
how?
mobile app or web app
why?
To get a better understanding about the patient's health status
why?
To provide better health care

Figure 6: Requirements and goals for "Access patient's data fast"

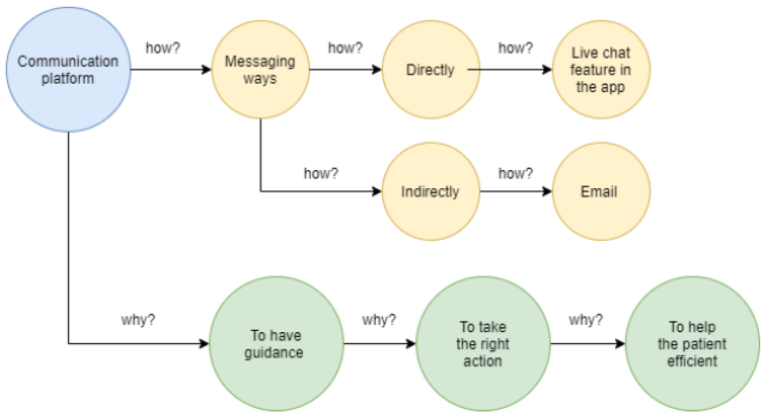


Figure 7: Requirements and goals for “Communication platform”

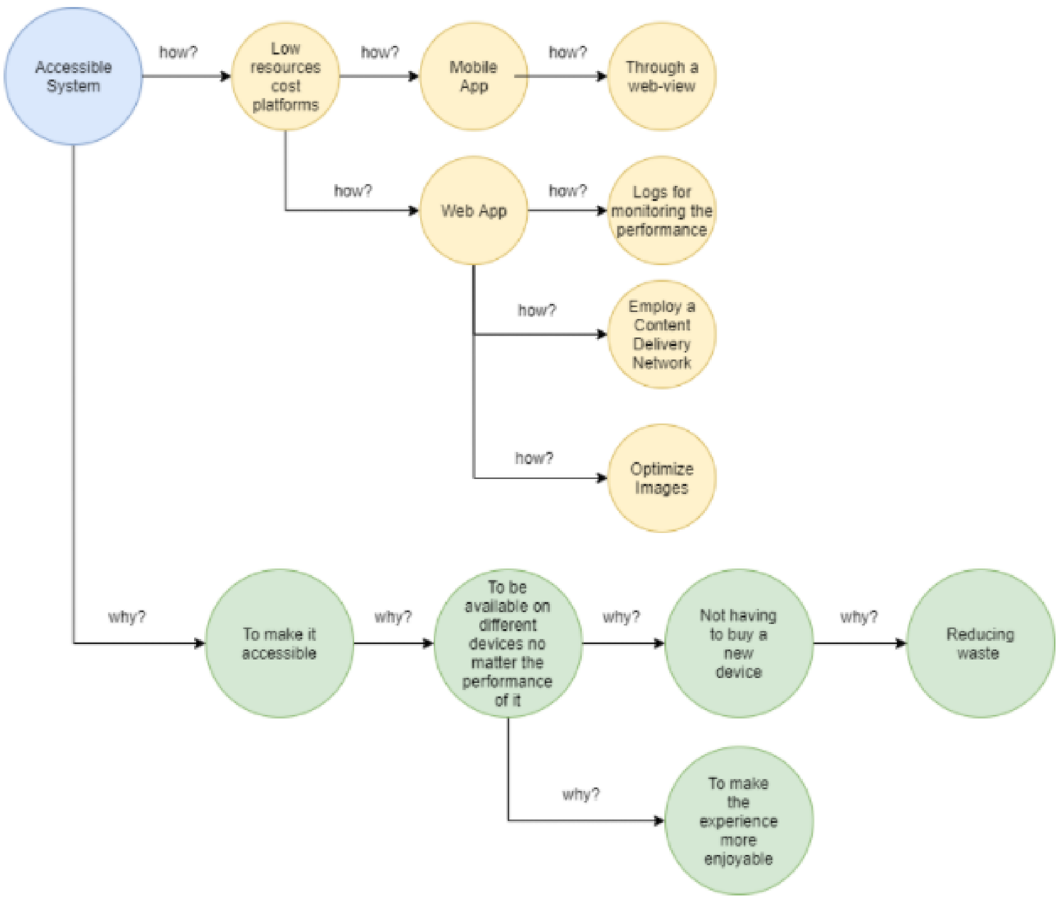

Accessible System
how?
Low resources cost platforms
how?
Mobile App
how?
Through a web-view
how?
Web App
how?
Logs for monitoring the performance
how?
Employ a Content Delivery Network
how?
Optimize Images
why?
To make it accessible
why?
To be available on different devices no matter the performance of it
why?
Not having to buy a new device
why?
Reducing waste
why?
To make the experience more enjoyable

Figure 8: Requirements and goals for “Accessible System”

1.2.1 ASR 1 as QA: Reliability

Portion of Scenario	Possible values
Source	End user, possibly in a specialized role, for example: doctors, nurses, formal/informal caregivers
Stimulus	End user tries to: 1. see the patient's data about their health status 2. communicate with other end users
Environment	At runtime
Artifacts	System, database
Response	Provide features needed
Response measure	Unique identification for patients in the database, correct associations between patients and their data, messaging protocols (SMTP, SMS), live chat, user satisfaction, amount of time or data lost when an error occurs.

1.2.2 ASR 2 as QA: Accessibility

Portion of Scenario	Possible values
Source	End user, possibly in a specialized role, for example: doctors, nurses, formal/informal caregivers
Stimulus	End user tries to access the system on their personal devices such as tablets, smartphones, laptops, etc., no matter the specifications of that device.
Environment	At runtime
Artifacts	System
Response	Provide optimized web app, mobile app
Response measure	User satisfaction, gain information of the user's device that is accessing the system, optimizing techniques for the mobile/web application.

1.3 Patients

Stakeholder	Patients
Stakeholder ID	SH3-P
Description	Patients are the main users of the system. They may vary in their needs and complaints. Some patients need informal caregivers, because they are unable to take care of themselves. Others need to stay in hospitals. Patients need to be covered financially by the insurance companies.
Expectations	1. Not paying extra for software. 2. The system should be easy and quick to use. 3. The data of the patients to be secure in the system.
Demands	1. Easy remote communication with doctors. 2. Clear communication with the insurance companies. 3. Communication with caretakers and sending them notifications, when needed. 4. Quick remote access and changing of medical status. 5. Checking the results from exams.

Business goals	1. For the HealthSuite system, the patients' desire is that they can access and change their data fast, in a reliable and secure way, without errors. See figure 9. 2. They desire from the HealthSuite system to have a stable platform to communicate with formal and informal caregivers and insurancers and send (Automatic) signals and interventions. See figure 10. 3. Another patients' desire is to be able to receive remote measurements and health care on different personal devices. See figure 11.

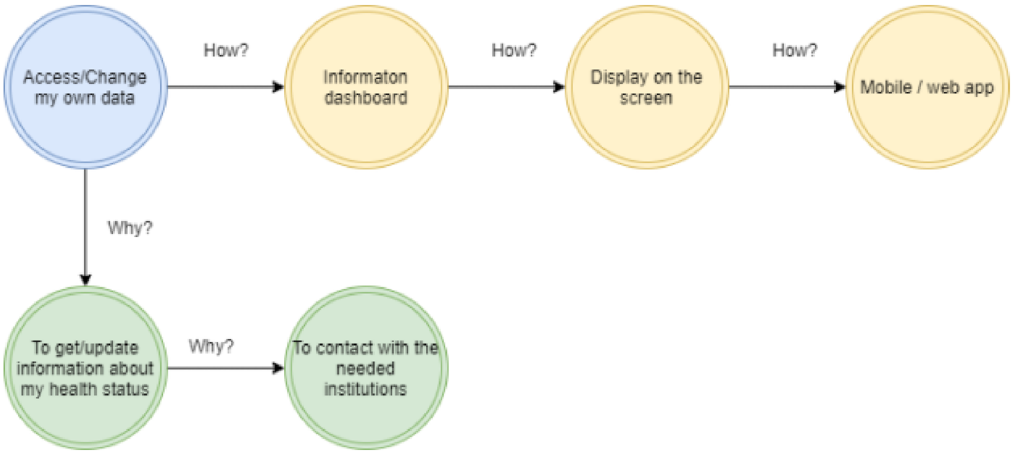

Access/Change my own data
How?
Informaton dashboard
How?
Display on the screen
How?
Mobile / web app
Why?
To get/update information about my health status
Why?
To contact with the needed institutions

Figure 9: "Access patient's own data"

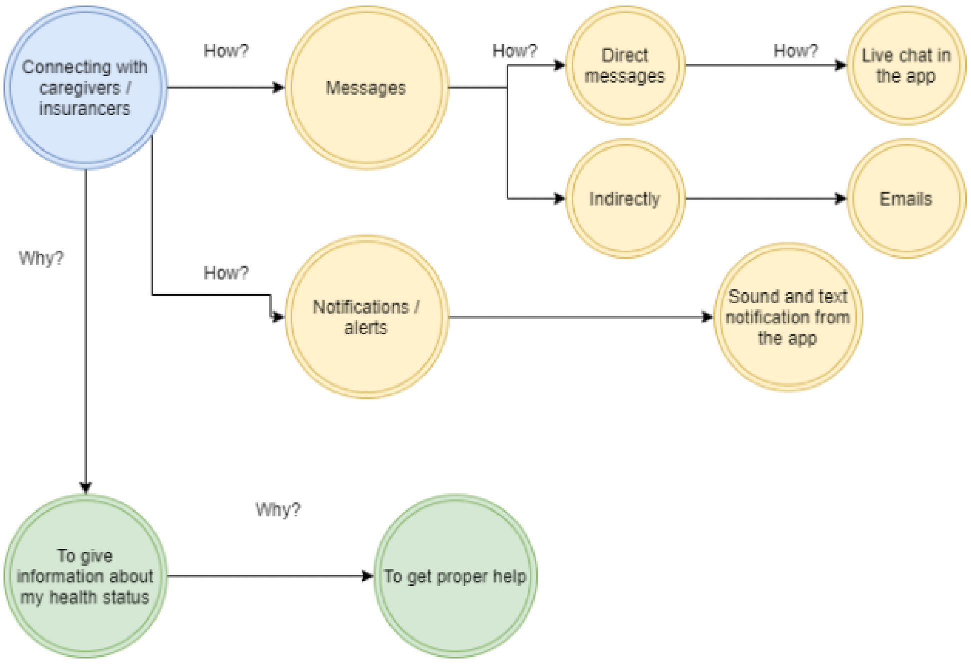


Figure 10: Notification system

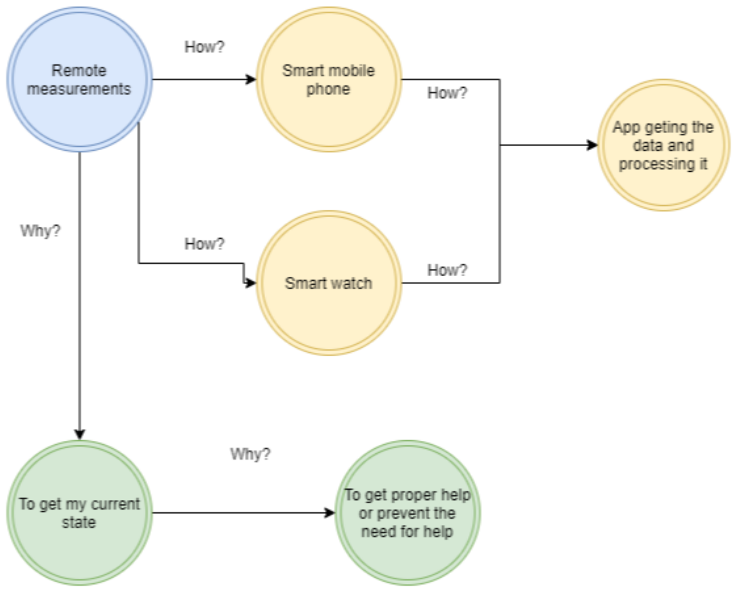


Figure 11: Remote measurement of data

1.3.1 ASR 1: Performance

Portion of Scenario	Possible values
Source	Patients
Stimulus	Patient tries: 1. to access his data 2. to get connection to his caregiver
Environment	At runtime time
Artifacts	System, Database
Response	Ensure a fast system use.
Response measure	Performance tests - Response time, Requests per second, User transactions,Virtual users per unit of time, Error rate.

1.3.2 ASR 2: Security

Portion of Scenario	Possible values
Source	Patients
Stimulus	Patients are using the system, they have access only to their own data, which is strictly protected.
Environment	At runtime
Artifacts	System
Response	The data is protected. The patients receive it in a secure way.
Response measure	Outcome reports from penetration tests, statistics about data leaks and security incidents related to the system (ideally none at all).

1.4 Payers / insurers

Stakeholder	Payers / insurers
Stakeholder ID	SH4-PI
Description	Insurance companies offer financial compensation that pays for medical and surgical expenses incurred by the insured. An insurer can establish a regular finance structure by assessing the overall risk of health risk and health system expenditures over the risk pool of the customer. In some situations, it is not the insurance company who pays for medical expenses, for instance in the case of treatments that are not covered by a patient's health insurance. Hence the term payers in addition to insurers.
Expectations	1. Insurers want access to a patient's data, including information on the cost and type of procedures they've had. This is in conflict with the privacy protection agency's business goal. 2. The insurance company expect that the software makes the communication flow and bill proccessing easier thus reducing costs. 3. The insurance company expects that by using the system the healthcare providers lowers costs, thus the insurer company have to pay less to cover the procedures of its customers.
Demands	1. Insurer companies demand clear feedback and fast and detailed information from the patient's status. 2. Insurer companies demand that the requested data will be delivered to them within 2 seconds they request it. 3. Insurer companies demands that by using the HealthSuite system the risk of insurance frauds will be reduced.
Business goals	1. For the HealthSuite, the insurer company desires that the system ensures a continous information advantage over the competitors. See figure 12. 2. For the HealthSuite, the insurer company desires that custom tailored services could be created for customers which better satisfy their needs. See figure 13. 3. For the HealthSuite, the insurer company desires that a more efficient administration flow could be achieved to increase productivity in the company. See figure 14.

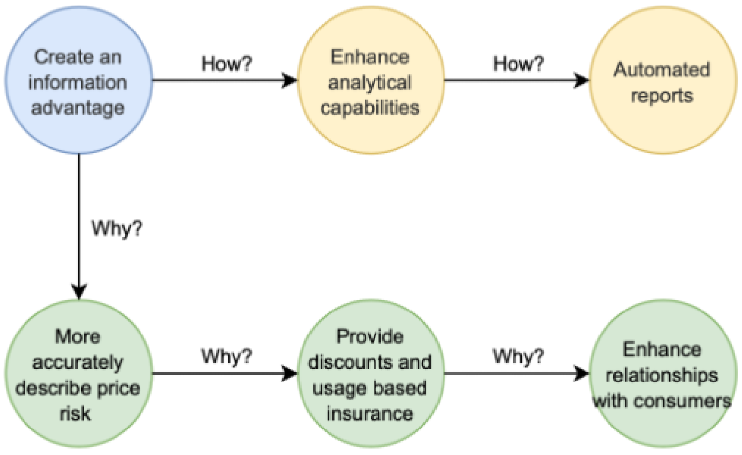

Create an information advantage
How?
Enhance analytical capabilities
How?
Automated reports
Why?
More accurately describe price risk
Why?
Provide discounts and usage based insurance
Why?
Enhance relationships with consumers

Figure 12: Requirements and goals for "Create information advantage"

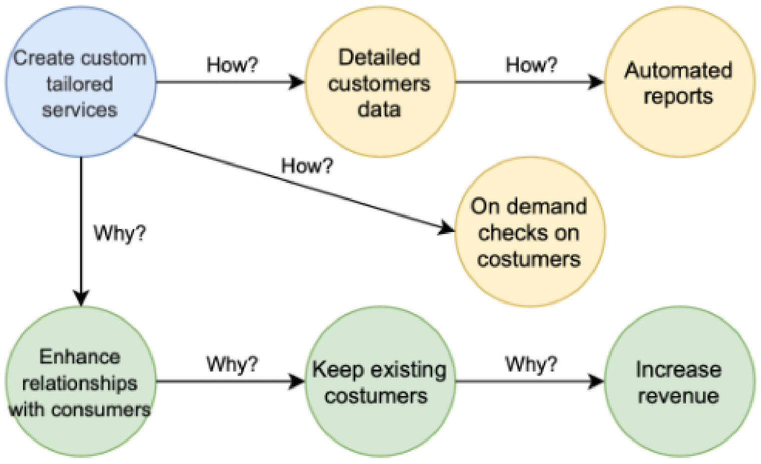


Figure 13: Requirements and goals for “Create custom tailored services”

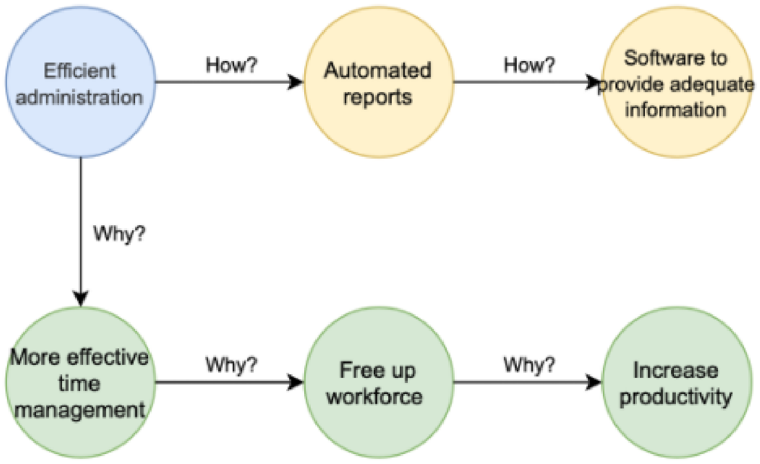


Figure 14: Requirements and goals for “More efficient administration”

1.4.1 ASR 1: Availability

Portion of Scenario	Possible values
Source	Insurer company, insurer agent
Stimulus	End user tries to: Access patient data on their computer.
Environment	At runtime
Artifacts	System
Response	Information is provided with data from the patient, even when other parts of the system failed.
Response measure	The system needs over 99.99% uptime during worktime when the insurer company usually makes requests to the system.

1.4.2 ASR 2: Interoperability

Portion of Scenario	Possible values
Source	Insurer company, insurer agent
Stimulus	End user tries to: Access patient data on their computer from different healthcare providers where the patient got a treatment.
Environment	At runtime
Artifacts	System
Response	Information is gathered from different healthcare providers.
Response measure	All insurers should be able to gather data from at least 99% of healthcare providers using different software, that means there could be always minor practices where no such system is established.

1.5 Privacy protection agency

Stakeholder	Privacy protection agency
Stakeholder ID	SH5-PPA
Description	The privacy protection agency is the organization that ensures the safe transfer, storage and use of information in the system. This agency must ensure the information security of all users in the system. Considering each user separately, it should offer the necessary security to all of them.
Expectations	1. The Healthcare suite is a safe digital environment. 2. To ensure that information is stored, transferred, shared and used in a protected manner. 3. The users' workflow is not blocked while providing security. 4. It is possible for the privacy protection agency to easily perform reliable audits to verify that data is in fact stored and transferred in a safe way and data is only accessible to those who are allowed to access it.
Demands	1. Easy operation while maintaining security. 2. Access to required information when requested. 3. No data is leaked to outside the system.

Business goals	1. The privacy protection agency desires to ensure the reliable flow of information in the system. See figure 15. 2. The privacy protection agency aims to ensure information security for all users. See figure 16.

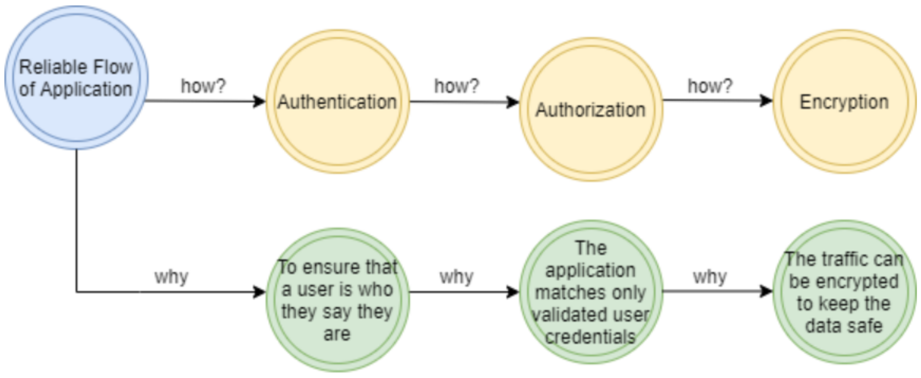

Reliable Flow of Application
how?
Authentication
how?
Authorization
how?
Encryption
why
To ensure that a user is who they say they are
why
The application matches only validated user credentials
why
The traffic can be encrypted to keep the data safe

Figure 15: Reliable Flow of Application

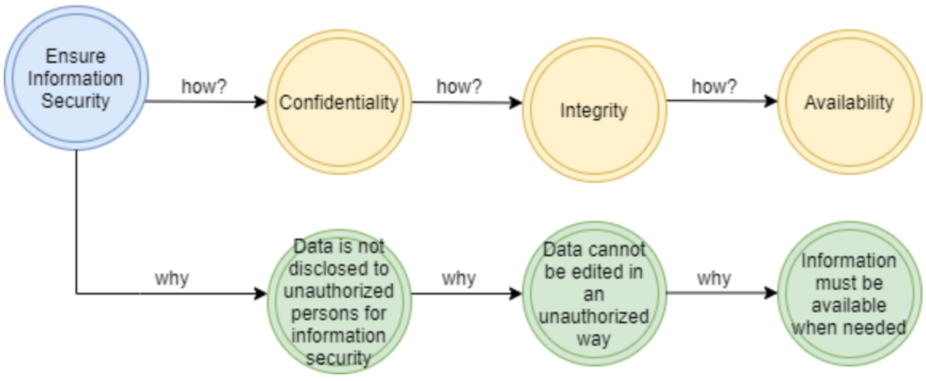

Ensure Information Security
how?
Confidentiality
how?
Integrity
how?
Availability
why
Data is not disclosed to unauthorized persons for information security
why
Data cannot be edited in an unauthorized way
why
Information must be available when needed

Figure 16: Ensure Information Security

1.5.1 ASR 1: Reliability

Portion of Scenario	Possible values
Source	Privacy protection agency employees
Stimulus	A privacy protection agency employee tries to influence the architecture of the system in such a way that it is capable of secure and stable usage.
Environment	At configuration time
Artifacts	System
Response	Ensure a reliable system use.
Response measure	Information security, Usage security, Reliable information flow.

1.5.2 ASR 2: Privacy

Portion of Scenario	Possible values
Source	End users, this could include patients but also staff, informal caregivers, or other users.
Stimulus	End user tries to use the system normally, for instance entering information into the system, accessing information stored in the system, making changes to data, making use of functions that rely on API connections to other software.
Environment	At runtime
Artifacts	System
Response	The data is stored or retrieved in a safe way, only when necessary, and no third parties can gain access. Particularly sensitive data such as medical history or passwords are always sufficiently encrypted.
Response measure	Outcome reports from penetration tests, statistics about data leaks and security incidents related to the system (ideally none at all).

1.6 Conflicts of interest

Healthcare providers aim to spend less time on care processes, whereas patients and informal caregivers might want them to spend more time on care processes. The same goes for the amount of money spent on care, even though patients and informal caregivers don't care as long as the quality of care remains the same. Informal caregivers want access to patient's data and communication with healthcare providers, but healthcare providers may not always want to provide data access, spend time on data access management/control, or spend time on communication with informal caregivers. Healthcare providers want to receive high amounts of compensation from insurance providers, whereas insurance providers want to limit the compensation for care given by healthcare providers to the minimum amount necessary. It is possible that patients do not want informal caregivers to have access to their data, but in some cases informal caregivers may want to ignore that and still have access (that can be a good thing or not). Insurance companies want to increase the proportion of care costs paid for by the patients, while the patients themselves want to reduce it and let insurance companies pay more.

2 Architecture sketch

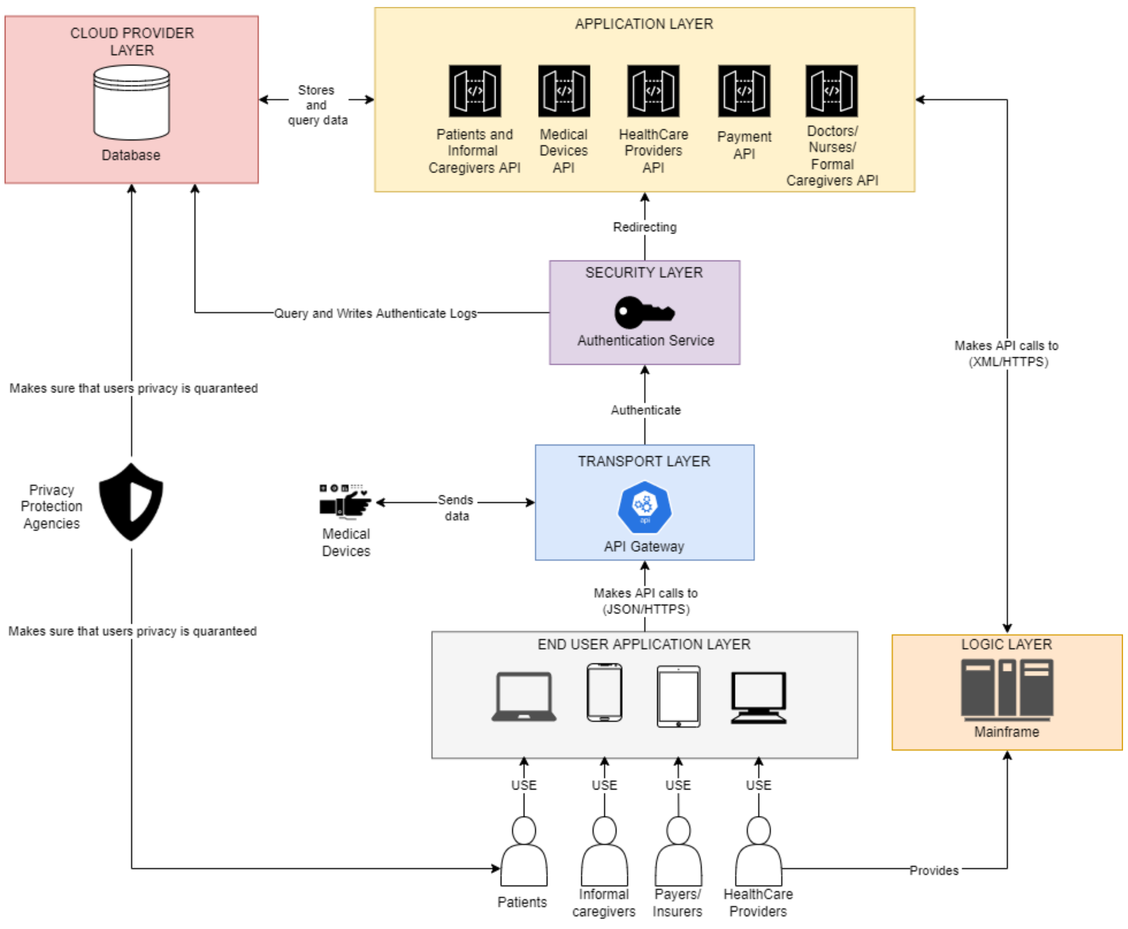


Figure 17: Architectural sketch of the project

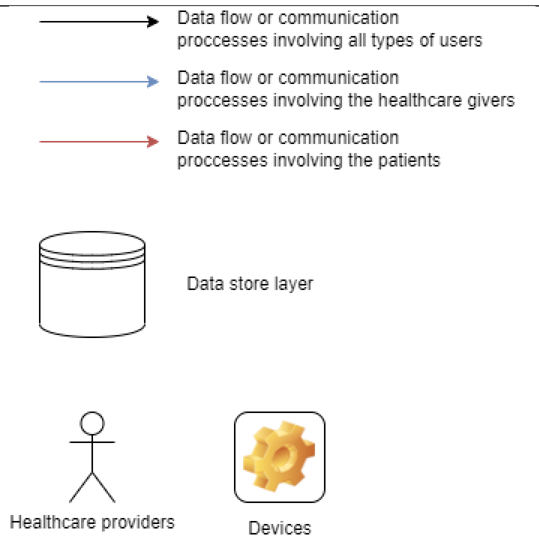
The project offers to end users two ways to use the software, first: a mobile application, second: a dynamic web application. It is important that the software is represented as a web application because it offers to its users an efficient way to use it, without keeping track of or focusing on which hardware the software might run on. The mobile application is not a native application for smartphones, because that would make it accessible on fewer device types and requires the expensive development and maintenance of multiple parallel versions. It is just a viewpoint of the web application itself. It is important to offer users an easy way to access the software, for example: opening a mobile application is easier than opening the browser and typing the URL of the web application.

The software offers to its users two ways of contact, first: an email system, second: a notification system. The email system is just a back-up of the notification system in case that the users missed it. It will notify informal caregivers about the patient's medical status and medication. Also, the email system along side with the notification one, will notify Insurers about the Patient's medical bill, if the Patient is not insured, then the Patient itself will receive an email/notification to pay the medical bill. Also, it will notify the Patient about his medical status, medication or future appointments that the patient might have planned.

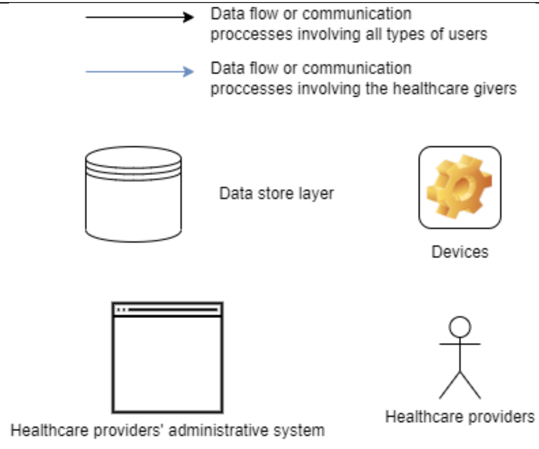
The Mobile Application and the Web application makes API calls to the API Gateway, afterwards if the user has the right credentials, it will be authenticated to the Application Layer. The Application Layer is build of five different APIs, each with its purpose, for example: a patient is interested in the features that the Application has for them, if they connect to the Medical Devices API, beside the fact that is a security thread, they will not know how to use it and they will not know what it does, for them, it is useless. The Application layer also is responsable of establish the connection with the Cloud Provider's Database, also it is connected to the Logic side of the Mainframe that is provided by the HealthCare Providers. The Privacy Protection Agency is responsible for making sure that Cloud Providers are not violating any GDPR regulations or users requests upon their data.

3 Viewpoints

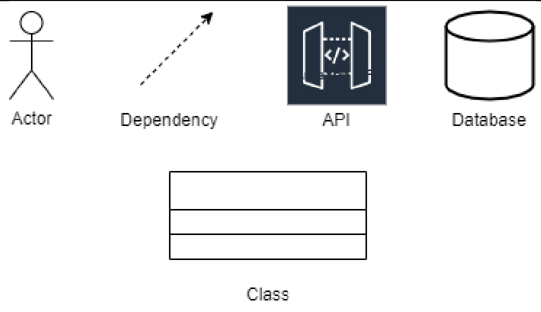
3.1 Patient privacy

Field	Description
Viewpoint name	Patient privacy and security
Viewpoint ID	`VP1-PPS`
Viewpoint overview	This viewpoint's main objective is to ensure that patients' data is kept securely and that data is secure while processing, with giving patients full control over their data that is in the system at that time.
Concerns	1. Access to a patient's personal information, such as their medical history or their home address, should be limited to people who really need access to it. 2. The patient has control over their own data. 3. A patient's personal information has to be accessible to doctors and other medical staff for diagnosis or research purposes without asking the patient's consent every time, as that would reduce their work efficiency. 4. A patient's personal information can not be accessed by anyone outside the system.
Stakeholders	`SH3-P`, `SH4-PI`, `SH5-PPA`
Modeling language	The modeling language for this view is a custom modelling language that describes the architecture of the Cloud, where the Database is, from the privacy agencies point of view, by depicting only those entities and the relationship among them that are relevant to the viewpoint overview.
Notation	 Data flow or communication proccesses involving all types of users Data flow or communication proccesses involving the healthcare givers Data flow or communication proccesses involving the patients Data store layer Healthcare providers Devices
Meta Model	See Figure 18
Rationale	On the Privacy Agencies side, they want to make sure that patients have control over their data. The patients should be able to manage their data, for example: they should give consent whenever a doctor, nurse or informal caregiver tries to see their medical data. The view is essential for showing patients that their data, either medical or personal, is safe while using the system, without violating GDPR.

3.2 Healthcare administration

Field	Description
Viewpoint name	Healthcare administration
Viewpoint ID	`VP2-HA`
Viewpoint overview	This viewpoint focuses on responsibilities, requirements and tasks of the HealthCare Administration staff. It shows what functionalities are implemented to address these requirements and concerns.
Concerns	1. The administration wants to be able to integrate this software with software and hardware already owned by healthcare providers, without having to spend a large chunk of the budget on new hardware or equipment. 2. The software should be intuitive and fast-responding, without troubling or consuming too much time when healthcare staff use it. 3. Provide research on medical cases to authorized third parties such as research facilities, with patients' approval.
Stakeholders	`SH1-HP`
Modeling language	The modeling language for this view is a custom modelling language that describes the architecture of the project from healthcare providers point of view, by depicting only those entities and the relationships among them that are relevant to solve concerns that the HealthCare providers stakeholder might have.
Notation	 Data flow or communication proccesses involving all types of users Data flow or communication proccesses involving the healthcare givers Data store layer Devices Healthcare providers' administrative system Healthcare providers
Meta Model	See Figure 19
Rationale	Healthcare providers are complex organizations with a lot of different processes, stakeholders and goals. If it is not managed properly, it can easily lead to confusion and mistakes, which can be disastrous in a medical context. Therefore it is crucial that every aspect, including the software used, does not cause problems for the administration staff and prevent them from making sure the healthcare provider can keep running smoothly. In this particular case, the software can even help to manage, archive, and control healthcare processes and make it easier for the administration staff to do their work correctly.

3.3 Healthcare insurance and payment capability

Field	Description
Viewpoint name	Healthcare insurance and payment capability
Viewpoint ID	`VP3-HIPC`
Viewpoint overview	This viewpoint has its main focus on ensuring that patients' financing is available, communicated securely and that the hospitals have direct contact with the insurancers. Also, this viewpoint focuses on HealthCare Providers worries that they might not receive the payment for the medical bill.
Concerns	1. Patient should have valid healthcare insurance and be able to cover their expenses. 2. The software should be scam-proof when the patients check their health insurance upon the registration process. 3. The system should be able to handle extreme cases, like homeless people or tourists. 4. Communication about payments should be clear and secure between the patients, healthcare providers and the insurancers.
Stakeholders	`SH3-P, SH4-I, SH1-HP`
Modeling language	The modeling language for this view is a custom modelling language that describes the architecture of the project from healthcare providers point of view, by depicting only those entities and the relationships among them that are relevant in order to avoid scams upon verification of the patient's health insurance and avoiding a hole in the healthcare providers budget.
Notation	
Meta Model	See Figure 20
Rationale	The insurance capability should be secure 100 percent because upon the registration the user will be asked if they are insured or not. If they are, they will need to input the identification of their medical insurance and a photo of their insurance. Not having this feature 100 percent secure can lead to scams and holes in the budget's of Healthcare Providers. If the patients do not have insurance because they might be tourists and they might have a health issue, they will be able to pay the medical bill via e-payment services.

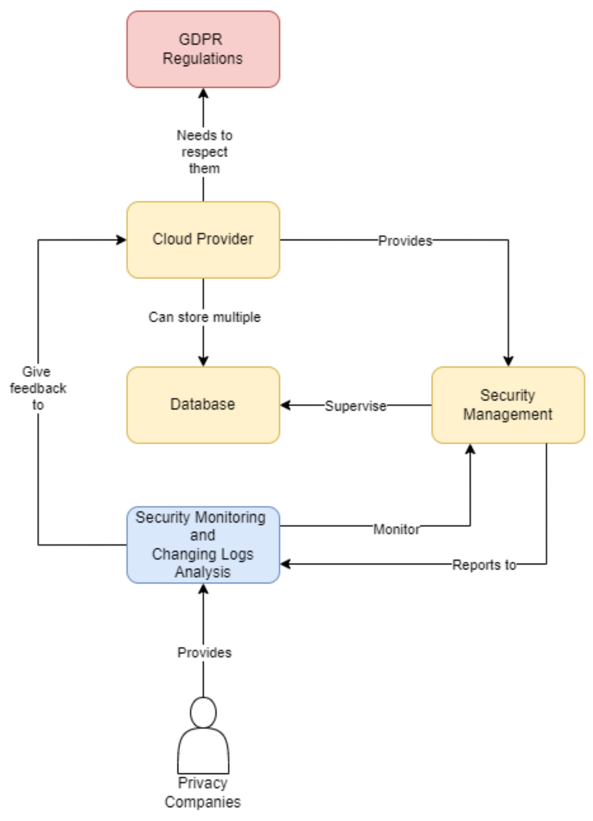


Figure 18: Metamodel for VP1-PPS

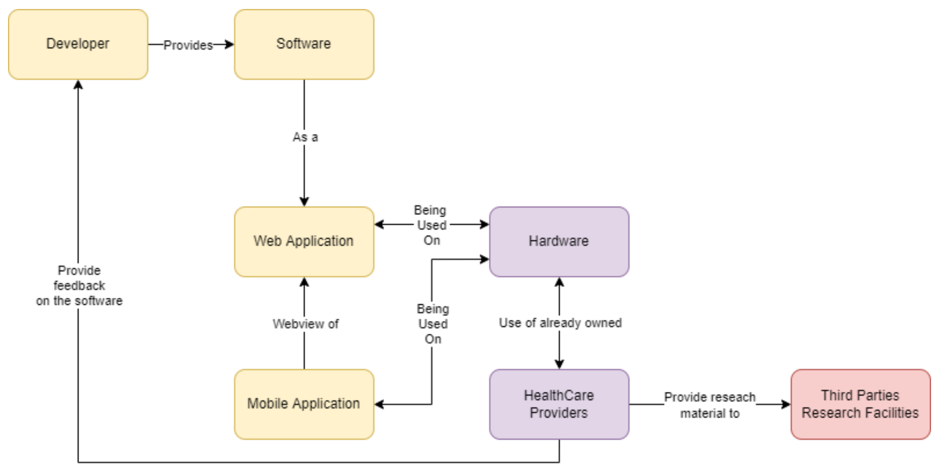


Figure 19: Metamodel for VP2-HA

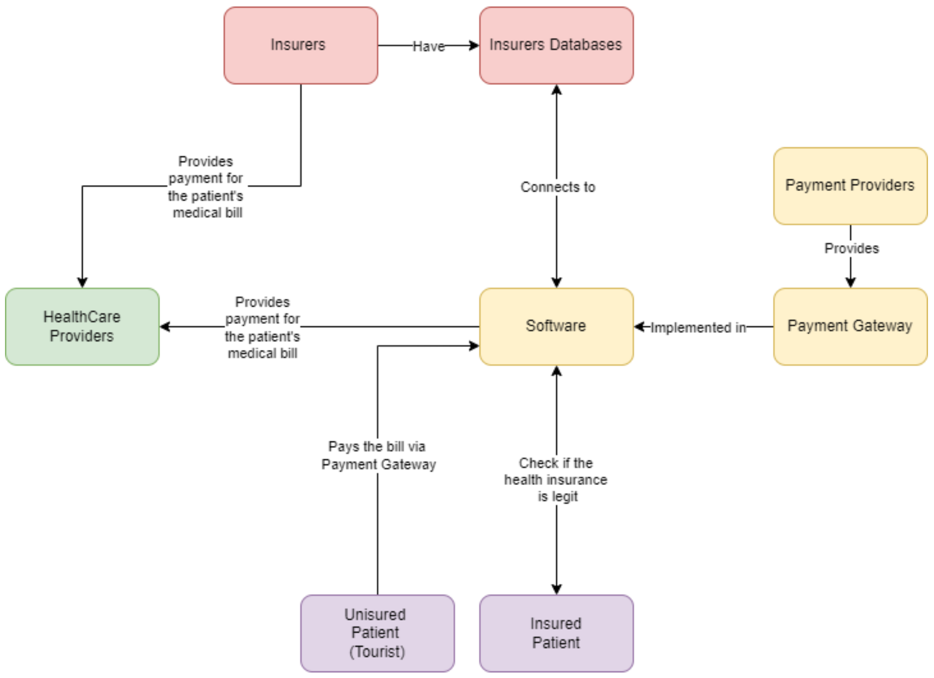

Insurers
Have
Insurers Databases
Provides payment for the patient's medical bill
Connects to
Payment Providers
Provides
HealthCare Providers
Provides payment for the patient's medical bill
Software
Implemented in
Payment Gateway
Pays the bill via Payment Gateway
Check if the health insurance is legit
Unisured Patient (Tourist)
Insured Patient

Figure 20: Metamodel for VP3-HIPC

4 Views

4.1 HealthCare not receiving the payment for patient's medical bill

Field	Description
View name	HealthCare not receiving payment for patient's medical bill
View ID	`V1-HW`
Related Viewpoint ID	`VP3-HIPC`
View Type	Logical View
View primary presentation	This view aims to prevent the worries from HealthCare Providers that can be caused by how the software will make sure that the patients are paying for their medical bills or medical services.
Logical Diagram	See Figure 21
Logical Diagram Legend	See Figure 21
Design Decisions	See subsection 6.1
Correlated Design Decisions	See subsection 6.2

View Rationale	This view is from HealthCare Service providers' concern that they will not be able to receive payment for patients' medical bill because of scams. This view especially focuses on the steps that the patient has to follow in the software in order to provide payment for the medical bill. There are two possibilities: first, the patient is covered by an Insurance Company, second, the patient is not covered, they are either tourists, either they stay temporarily in Netherlands. For the first option, the software, upon the registration process, it will ask for an ID of the insurance certificate and for the name of the Insurance Company. The patient will type in the field the ID of their insurance certificate and they will select from a Dropdown Menu the name of the Insurance Company. After that, the software which is connected to the Insurance Companies databases, will verify if that patient has health insurance. For the second option, for every appointment that the patient have to make, the software via Payment Providers, will ask the patient to pay in advance for that, either is by Credit Card or Debit Card. If that patient had a surgery, the software will ask for payment by sending a medical bill to the patient's email address along side with a notification. If the patient refuses to pay or ignores the emails or the notifications about payment, the local authorities will be announced.

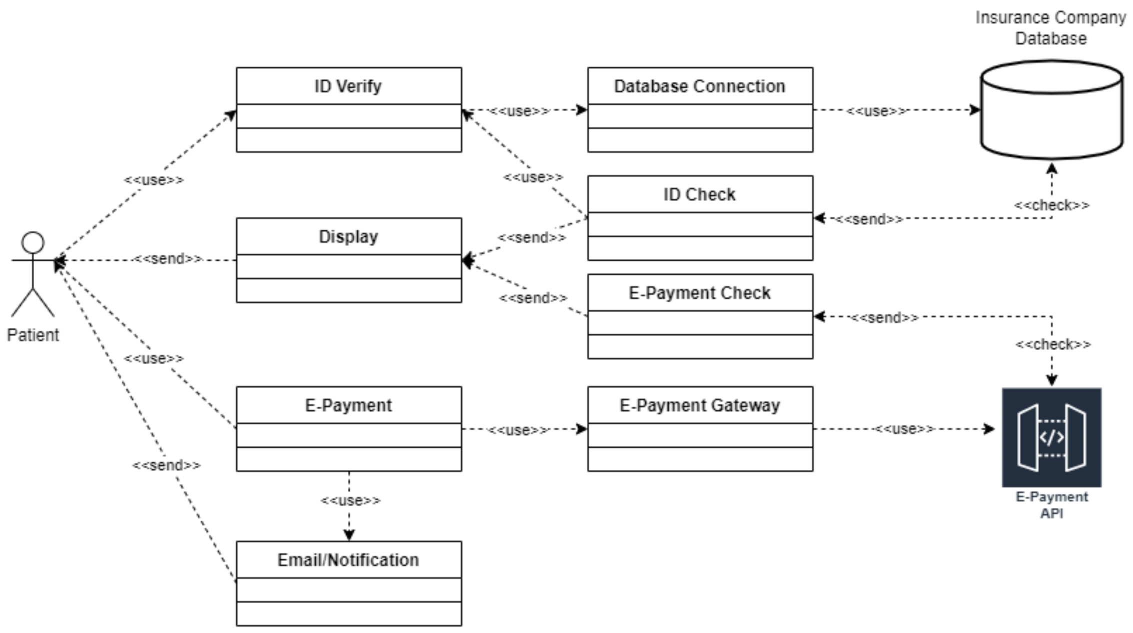


Figure 21: Logical Diagram for V1-HW

4.1.1 Element catalog

Element	Responsibility
Patient	Patient represent the end-user who will go through registration process in the Software.
ID Verify	This class is reponsable with storing the input from Patient (identification number of the health insurance).
Database Connection	This class is reponsable with connecting the software to the Insurance Company Database.

ID Check	This class is reponsable to verify if the identification number of the health insurance that the Patient typed in is valid. For this, this class will take the ID of the health insurance from ID Verify Class and it will querry directly from the Insurance Company Database.
Insurance Company Database	This database belongs to the Insurance Company provider.
E-Payment	This class is reponsable of two things. First, is to redirect the user to the e-payment gateway. Second, is to ”communicate” to the Email and Notification classes that the e-payment procedure is complete or incomplete.
E-Payment Gateway	This class is reponsable of making the transition from the software web page to the e-payment provider web page.
E-Payment API	This API is implemented by the e-payment provider in their web site. Also, it will return to the E-payment class a response in which it announce if the user in cause did complete the payment or not.
Email and Notification	These classes are responsable with notifying the user if they did complete or not the payment procedure. The Email class will notify the user by email, the Notification class will notify the user by sending push notifications in software.

4.2 HealthCare detecting security intrusion attempts

Field	**Description**
View name	HealthCare detecting security intrusion attempts
View ID	`V2-HS`
Related Viewpoint ID	`VP1-PPS`
View Type	Process View
View primary presentation	The system aims to detect any suspicious activity by constantly monitoring the traffic using an Intrusion Detection and Prevention System (IDS) and issues alerts when such activity is discovered. The program, which compares observed activity to usual usage patterns, detects and logs any behavior that differs from routine user access patterns. For certain users and applications in the system, such as healthcare providers and insurers accessing or updating data on the network, different profiles are created. An anomaly can be identified in one of two ways by the system: Threshold monitoring establishes acceptable behavior levels and monitors whether they are met, such as a defined number of unsuccessful login attempts or monitoring the amount of time a user is engaged and the amount of data he or she downloads. The system-wide consumption of resources is observed through resource profiling. When a large volume of user data is requested by a single user, abnormal readings might indicate that illegal behavior is taking place. A security information and event management (SIEM) system collects any harmful activity or violation and records it centrally. The SIEM system combines data from many sources and use alert filtering algorithms to distinguish between malicious and false alarms. Intrusion prevention systems also monitor network packets entering the system to look for malicious activity and provide warning signals immediately.
`Process View`	See Figure 22
`Element catalog`	Elements and their properties: See Table 4.2.1 Relations and their properties: See Table 4.2.2

Rationale	Many attackers can get access to user credentials that enable them access to networks and data. No firewall is failsafe, and no network is impregnable. Because it allows for the identification and response to hostile traffic, an intrusion detection system (IDS) is critical for the security of patient data and privacy. The privacy protection agency, as well as patients and insurers, are all concerned about data privacy.
Design Decisions	The reason an intrusion detection system is built in and doesn't use off-site solutions or just uses firewalls is that they impose a resource load and once an attack goes through the perimeters, they don't help much anymore. A standalone IDS can be used as an additional layer of protection, and can be configured to be more complex and granular. A SIEM also enhances system functionality, maximizes security, and increases system availability.

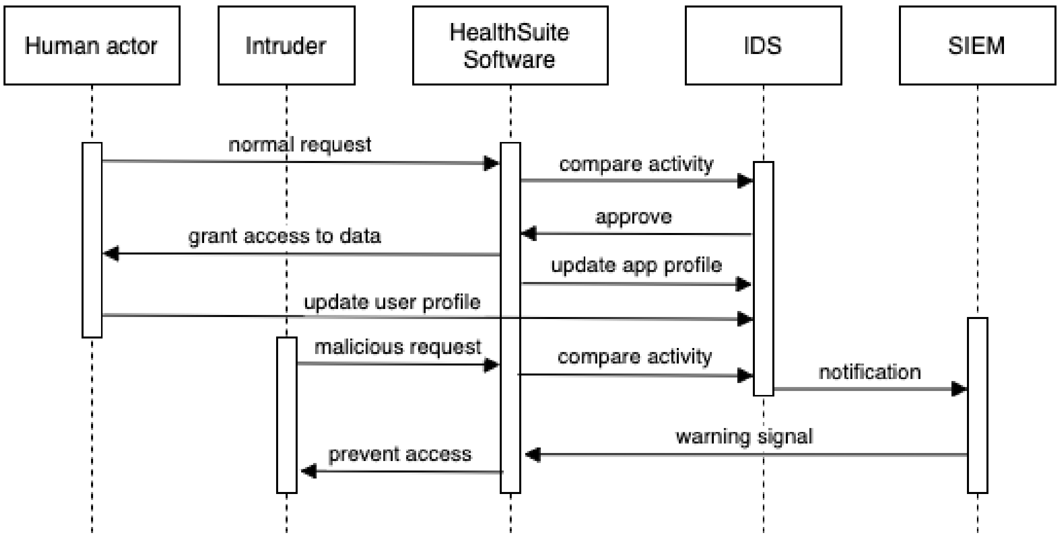


Figure 22: Process View for V2

4.2.1 Elements and their properties

Element	Definition	Subtypes
Human actor	System users accessing or updating data on the network	Healthcare provider, patient, insurer
Intruder	Unauthorized user who penetrates a system exploiting a legitimate user's account or legitimate user who makes unauthorized accesses or misuses his privileges.	
HealthSuite Software	The program that allows users to engage with a certain layout while respecting the various levels of end-user application.	Web application, mobile application
IDS	Software constantly monitoring the traffic for malicious requests using an Intrusion Detection and Prevention System (IDS) and issues alerts when such activity is detected.	Threshold monitoring, resource profiling
SIEM	Security Information and Event Management system. A system which uses alarm filtering to differentiate malicious activity from false alarms.	

4.2.2 Relations and their properties

Relation	Description	Source	Target
Normal request	Usual request from non-malicious source	Human actor	HealthSuite Software
Compare activity	Compares observed activity to usual usage patterns.	HealthSuite Software	IDS
Approve	If the IDS didn't detect any abnormalities IDS allows the information flow.	IDS	HealthSuite Software
Grant access to data	The software complies for the users request	HealthSuite Software	Human actor
Update app profile	Update the historical usage patterns based on normal observed activities of the system-wide resource usage in the application.	HealthSuite Software	IDS
Update user profile	Update the historical usage patterns based on normal observed activities of individual work profiles about users and groups. For example from what device is XY usually logged in or what kind of requests a usual patient commences.	Human actor	IDS
Malicious request	Request coming form an intruder who is either exploiting a legitimate user's account or misuses his privileges.	Intruder	HealthSuite Software
Notification	In response to a suspected malicious request, the SIEM receives a notification with all of the necessary data for alert filtering.	IDS	SIEM
Warning signal	The SIEM notifies the appliciton after the malicious attempts have been recognized.	SIEM	HealthSuite Software
Prevent access	The program refuses to give data access and instead performs further inspections.	HealthSuite Software	Intruder

4.3 HealthCare administration providing reasearch cases to third parties

Field	Description
View name	HealthCare administration providing reasearch cases to third parties
View ID	`V3-HA`
Related Viewpoint ID	`VP2-HA`
View Type	Development View

View primary presentation	This view is describing the process of providing research on medical cases to authorized third parties such as research facilities, reflecting on the Healthcare providers concerns about the usage of the system and the process of sending data to third party provider, from developer's perspective. The view especially focuses on the steps that the administrator has to follow in the software in order to provide research to the third party system. First of all, the administrator needs to log in the system. Then the administrator needs to access the specific medical case - if he does not have permission, he needs to ask for it the doctor working on this case. Then the doctor accesses the file on the cloud and gives permission for sending. After that he sends it to the third party research facility through the platform.
Development Diagram	See Package diagram: 23
Development Diagram Legend	See Figure 19
Design decisions	HealthCare Administration staff is an important user group of the system, which has some specific requirements, as it manages the data of the patients. Administration part of the system is supposed to be developed in a fast, easily deployed and easy for use from the administration staff. That is why a fast and secure development flow was chosen.

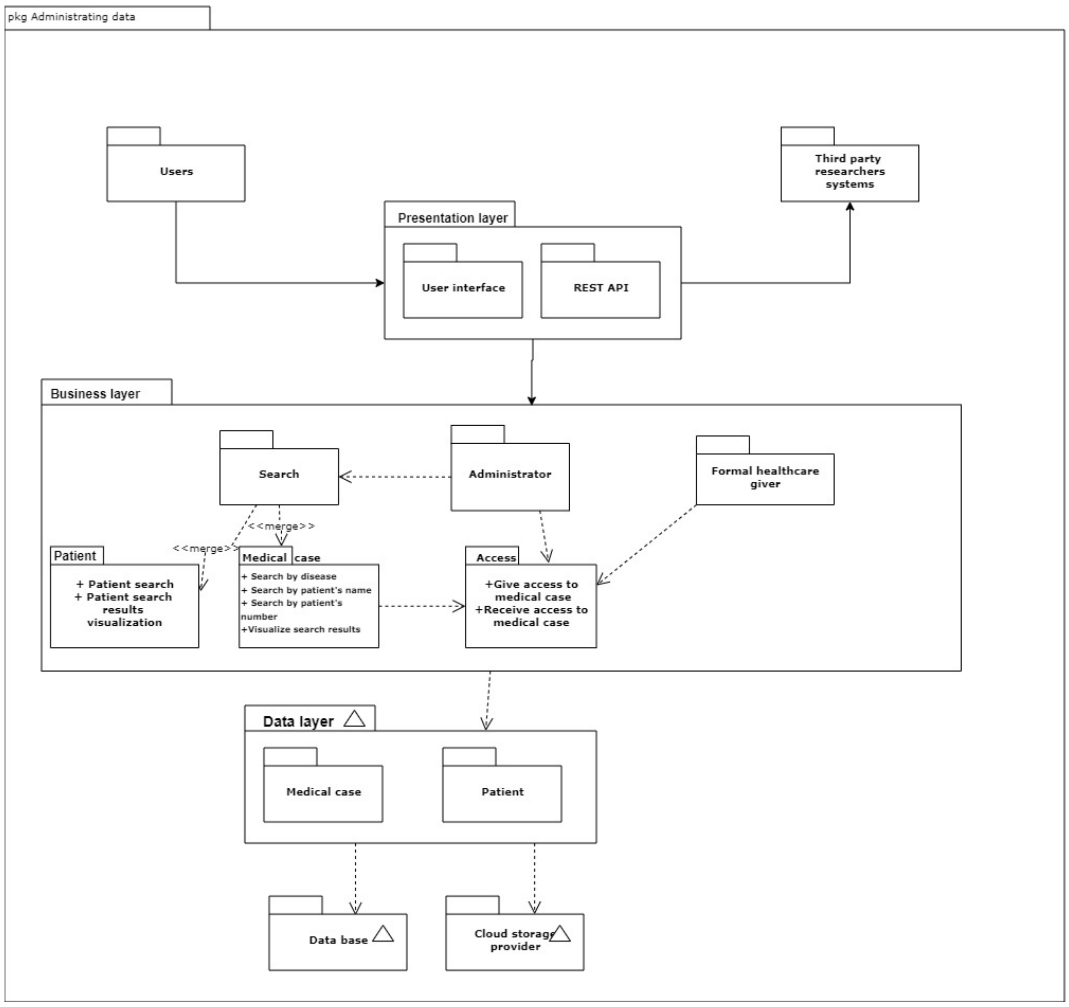


Figure 23: Development View for V3

4.3.1 Elements and their properties

Element	Definition
Users	System users accessing or updating data on the platform
Administrator	The user that is making the search and sending the data
Formal healthcare giver	The user that is given the access to the given medical case
Patient	Patient's data stored in the system
Medical case	Medical cases' data stored in the system
Database	HealthCare database
Cloud storage provider	A third party system provider, that gives HealthCare platform additional storage
Search	The class, responsible for searching the patient's and medical cases' data
Patient search	Subclass of Search, having the specific task of finding the patient's data
Medical case search	Subclass of Search, having the specific task of finding the medical cases' data
Access	The process of giving access to the patient's or medical cases' data
User interface	The part of the system with which the users are interacting with.
REST API	The layer of the system responsible for sending the data to the third party one.
Third party researchers systems	The third party to which the medical cases' data is sent

5 Mapping between views

View1	View2	View3
Display is subtype of Health-Suite Software	**HealthSuite Software**	**Presentation Layer** is subtype of HealthSuite Software
Patient is subtype of Users	**Human actor** equal to Users	**Users** is equal to Human actor
Insurance company database subtype of Data base		**Data Base**
ID check subtype of Access	**Approve** is equal to Access	**Access** is equal to Approve

6 Design decisions and rationale

6.1 Design decisions and rationale for V1-HW

Field	Description
Rationale	In case that not all patients are insured from a medical point of view. They also have the option to pay for the medical services and bills directly via the software, through an e-payment platform provided by an e-payment provider. If the patient has health insurance, upon registration they will be asked to input their identification number of the health insurance. After that, it will be checked by directly querry from the health insurance company's database. Because of the variety of companies that provide health insurances, an agreement was done between Healthcare providers and Health insurance companies in which it will allow the software to access their databases only to check if the indentification number of a health insurance is valid. To know that the software connected to the right Health insurance company database to check if the ID is available, the users will be ask to select the company name from a drop-down box in the application. If a health insurance company refuse to accept this agreement, then the ID of the health insurance will be checked the old way but with a little twist. The patient will have to go to the healthcare providers to see if their health insurance is valid, after that healthcare providers will give the patient a 4 digit activation code. The 4 digit activation code will be introduced in the software, and if it is valid, the software will see that the patient is covered by a health insurance. From now on, the medical bills will be directly redirected through the software from the healthcare providers to the health insurance companies.
Notations of Taking decisions	1. Design problem = How the patient will pay for the medical bill? 2. Sub-Problem 1 = E-payment or physical payment? 3. Sub-Problem 2 = What if the health insurance company do not agree with the fact that the software needs to access their databases in order to verify the IDs of the health insurances?
Taking decisions	See Figure 24
Design Space	See Figure 25

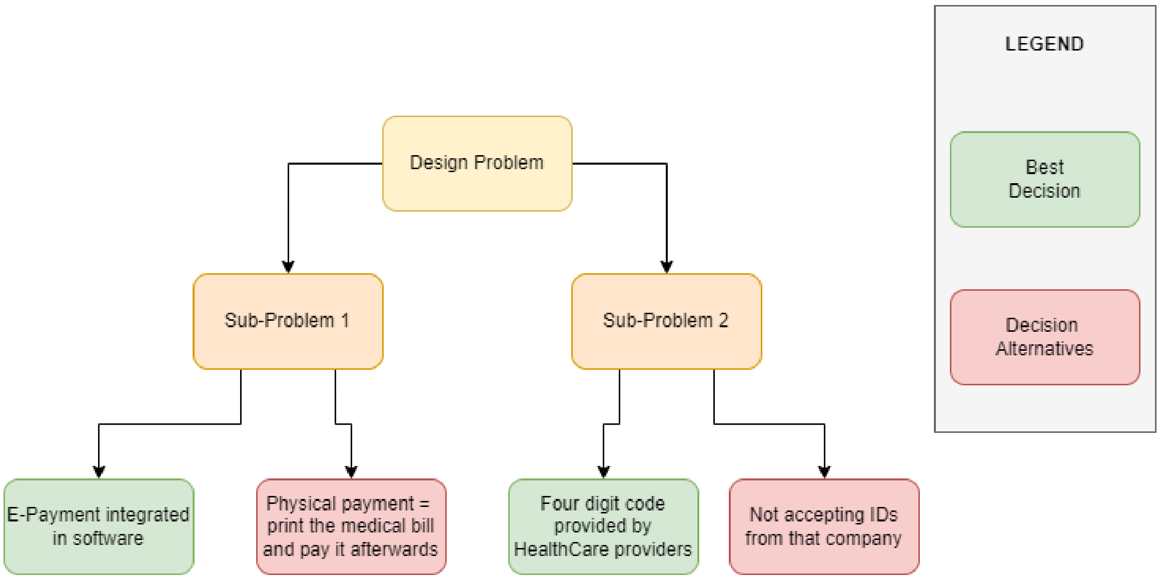

LEGEND
Best Decision
Decision Alternatives
Design Problem
Sub-Problem 1
Sub-Problem 2
E-Payment integrated in software
Physical payment = print the medical bill and pay it afterwards
Four digit code provided by HealthCare providers
Not accepting IDs from that company

Figure 24: Taking decisions for V1-HW Design Decision

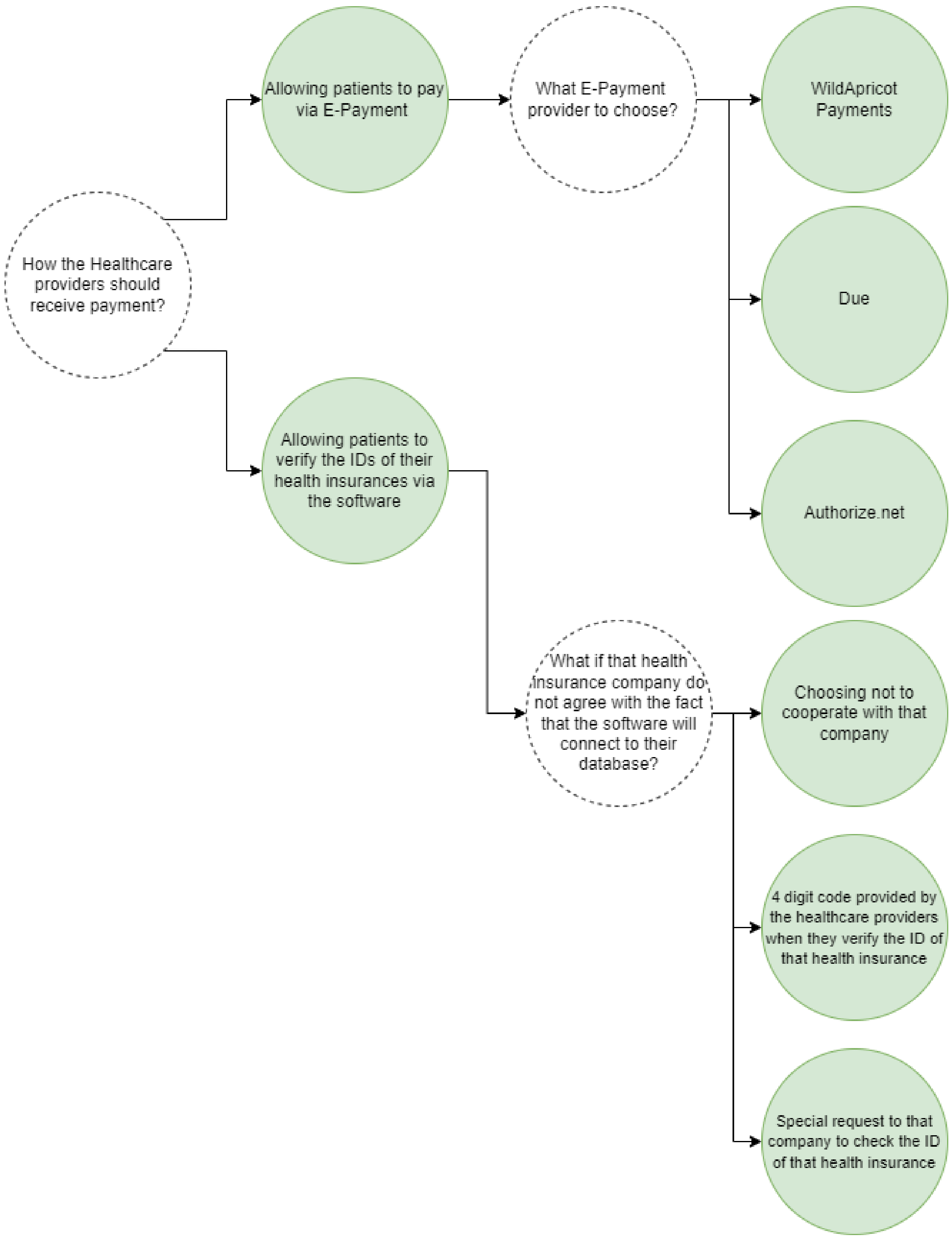

How the Healthcare providers should receive payment?
Allowing patients to pay via E-Payment
What E-Payment provider to choose?
WildApricot Payments
Due
Authorize.net
Allowing patients to verify the IDs of their health insurances via the software
What if that health insurance company do not agree with the fact that the software will connect to their database?
Choosing not to cooperate with that company
4 digit code provided by the healthcare providers when they verify the ID of that health insurance
Special request to that company to check the ID of that health insurance

Figure 25: Design Space for V1-HW Design Decision

6.2 Design decisions and rationale in case of failure

Field	Description
Rationale	In case of failure upon software level, for example: the request servers failed due to high volume in user requests, the architecture provides two possibilities to avoid this: first, developing E-Smart Sensor, second: making use of checkpoint mechanism. First, the E-Smart Sensor will be responsable with assigning request servers in case that one server cannot handle a number of request, by distributing the requests among the servers. Also, it will be responsable with transfering the request to another request server if the initial request server failed. The failed server, will be rebooted, if the failed server is not recovering after a reboot, a technician will be notified about the problem. The failed server will be exluded from the distributed system until it is working properly. Second, checkpoints will be used, if the E-Smart Sensor fails, and the request server is not working, the user request will be saved and transfered to a back-up request server to be handled. The back-up request servers are not part of the E-Smart Sensor servers, they are especially for checkpointing mechanism and their only use is when E-Smart Sensor fails. All toghether, beside the beneficial part for the performance of the software, these decisions are beneficial for environment too, by reducing energy consumption and carbon footprints of the servers. The energy consumption along side with the carbon footprints are reduced because not all request servers are working and powered on, the only servers that are working are the ones who are handling requests. For example: instead of distributing 10 requests to 10 servers that are powered on and working 24/7, distribute 10 requests to 3 servers that can handle those 10 requests by powering them on only when needed.
Notations of Taking decisions	1. Design problem = How the requests will be handled when the request server fails? 2. Sub-Problem 1 = What it will happen if E-Smart Sensor fails? 3. Sub-Problem 2 = What it will happen if a large number of requests cannot be handled? 4. Sub-Problem 3 = How to reduce the energy consumption and the carbon footprint of request servers?
Taking decisions	See Figure 26
Design Space	See Figure 27

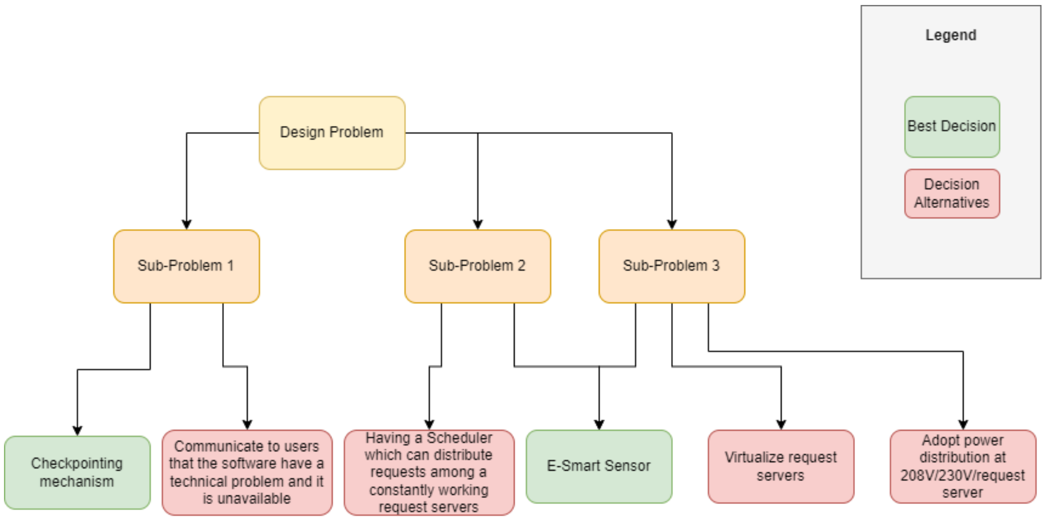

Legend
Best Decision
Decision Alternatives
Design Problem
Sub-Problem 1
Sub-Problem 2
Sub-Problem 3
Checkpointing mechanism
Communicate to users that the software have a technical problem and it is unavailable
Having a Scheduler which can distribute requests among a constantly working request servers
E-Smart Sensor
Virtualize request servers
Adopt power distribution at 208V/230V/request server

Figure 26: Taking decisions in case of failure

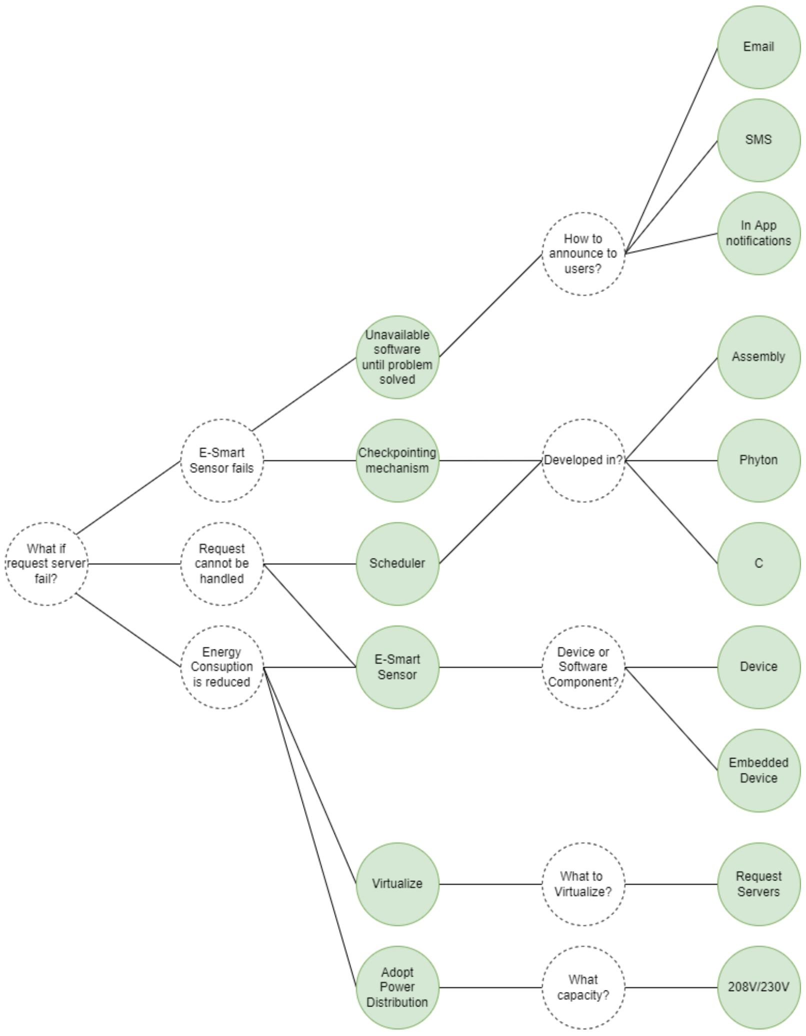


Figure 27: Design Space in case of failure

6.3 Economic factors

Field	Description
Rationale	For any product or service to be successful, it should be economically viable. The HealthSuite is no exception; if the development and maintanance costs cannot be covered, it is a bad investment. The software is unlikely to receive sufficient maintenance or may not be created in the first place. Furthermore, excessive costs or limited return on investment is a waste of the already limited healthcare budget of governments and healthcare providers, which could have been better spent on alternative ways to improve the care that people receive. This idea was central to many design decisions. These considerations are relative to competing products. Other software for health-care infrastructure and processes does exist and if the HealthSuite does not offer a unique selling point that allows it to maintain a reasonable market share, it will be replaced by other products.

Notations of Taking decisions	🟩 Accepted solution 🟨 Main design problem 🟥 Rejected solution 🟧 Sub-problem → Consists of
Taking decisions	See Figure 28.
Design Space	See Figure ?

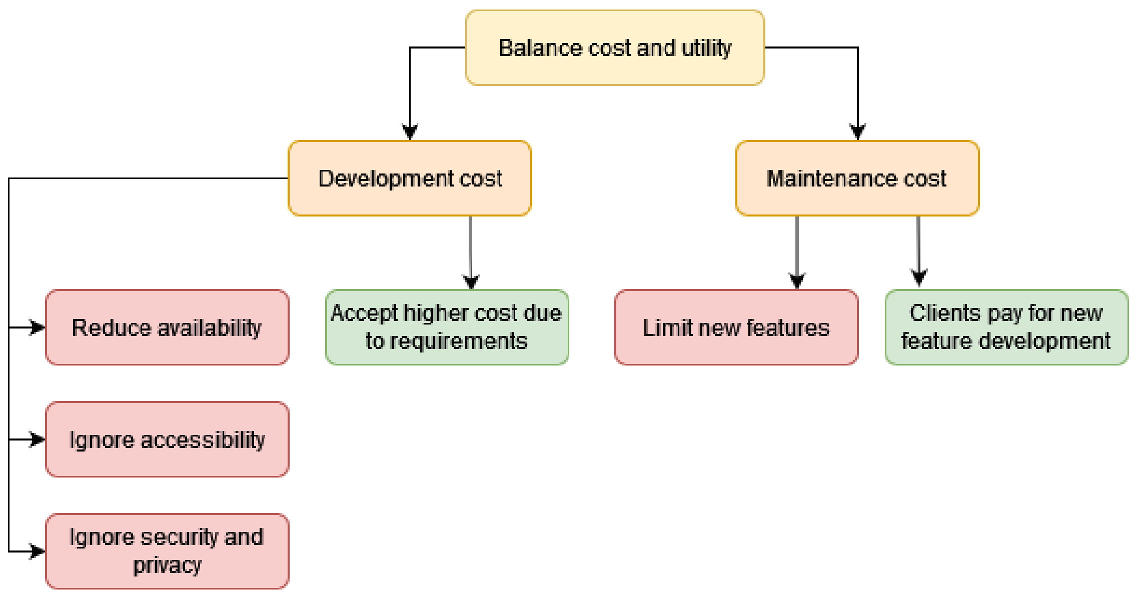


Figure 28: Taking decisions about the cost of the system.

7 Assessment

7.1 Assesment for healthcare providers

Stakeholder	Healthcare providers
Stakeholder ID	SH2-HP
Utility tree	See figure 29.
Rationale accessibility	The system should be accessible for users, in this case the employees of the healthcare provider in particular since the healthcare provider is the stakeholder. Accessibility consists primarily of two aspects: how satisfied users are with the software and how well it scores in standardized usability tests. These tests include metrics like the time before something happens on screen after a user input, and whether all interactive elements are clearly visible and consistent. Based on these two aspects, the QA is split into the attribute refinements user satisfaction and usability. Since they are related (though not entirely the same), the QA scenarios for them are also similar. The value to the healthcare provider and the impact on the system architecture are not absolutely crucial but also not unimportant relative to other QAs.
Rationale modifiability	Modifiability is important to the healthcare provider, because their needs might change in the future. Switching to another software platform like the HealthSuite is a time-consuming and costly affair so it should be avoided even if healthcare providers aren't always able to anticipate their future requirements. Therefore it is essential that the system can be modified if necessary. It is acceptable that this incurs additional costs, because new features have to be developed, but it should not take too long to develop new features. For instance, if new hardware (for instance a new type of MRI scanner) becomes available to a hospital, they want the HealthSuite to support it, and while they can wait for this support to be added, it should happen in a reasonable timeframe. Even if features cannot always be implemented or are implemented in a different way than requested, the healthcare provider always expects to be kept up to date on the status of the request. This is very important to healthcare providers, and the impact on the HealthSuite is large, because the ability to extend the system should be a primary concern while developing the system, and as long as the system is in active use, development time should be spent on adding new features based on requests from healthcare providers and potentially other stakeholders as well. However, responding to requests in time is relatively straightforward to organize, even though it is very important to healthcare providers.
Rationale availability	The availability of the system is also very important since it's an application in the healthcare domain. The uptime of the system must be very high, because failures can cause care processes to be interrupted, delayed or cancelled, resulting in unsatisfied patients or potentially even reduced health outcomes for some patients. It can also result in wasted time of health professionals who are expensive, and lost data depending on how the system works. Healthcare providers would like to avoid this as much as possible, and since healthcare continues 24 hours a day, 7 days a week, the system should also be available at all times. Availability is refined as uptime and the related scenario is quite straightforward: an employee attempts to access the system, which should result in the system responding as expected because it is available. This is important to the healthcare provider, and has a high impact on the architecture, since it requires very careful thought about software, hardware and how it is all connected. Distributed systems can be very complex and making sure it does not break down, even with heavy use or fluctuations in use, is a daunting task.

Sensitivity points	1. The HealthSuite is reliant on support staff to fulfill their SLA about responding to feature requests in time. 2. Some design decisions may make it harder to make the system modifiable. 3. Modifiability is hard to define: which aspects of the system should be modifiable and to what degree? 4. The healthcare providers may not always have a good grasp of how hard certain features might be to build, resulting in unmet expectations.
Tradeoffs	1. The HealthSuite has a limited development time budget, as they are limited in the number of developers they employ, and they also need to work on other products. On the other hand, the healthcare providers may request a lot of features, all of which need to be done in time. 2. Sometimes, many healthcare providers may request many different features in a short time span, and maybe even very complicated features. Other times, there may not be many feature requests at all. This flexibility is good for the healthcare providers but the developer of the system prefers consistency over flexibility in this case because it makes the workload easier to predict, making it easier to fulfill their promised timeframes and SLAs. 3. Availability often results in significantly higher costs. The healthcare providers do not want the system to be very expensive, but they do want it to be highly available. 4. Modifiability could conflict with availability, since it is hard to make a system that is flexible enough to be easily modifiably but also follows strict constraints that allow it to be resilient and available at all times.
Risks	1. If the system becomes too complex to easily modify, it could become extremely expensive to keep modifying it. 2. If accessibility is not taken into account from the start, the system will not be appreciated by employees as much and the healthcare provider may want to switch to another system.
Non-Risks	1. Limited usability is not a risk, because there are standardized methods for how to achieve better usability and it can be continuously improved after the system is launched. 2. If the time before developer response upon a feature request is too long, it can easily be remediated, so it is not a risk that this would be a structural issue as long as the company actively tries to keep their promises (which can be handled easily by agreeing on an SLA with financial consequences).

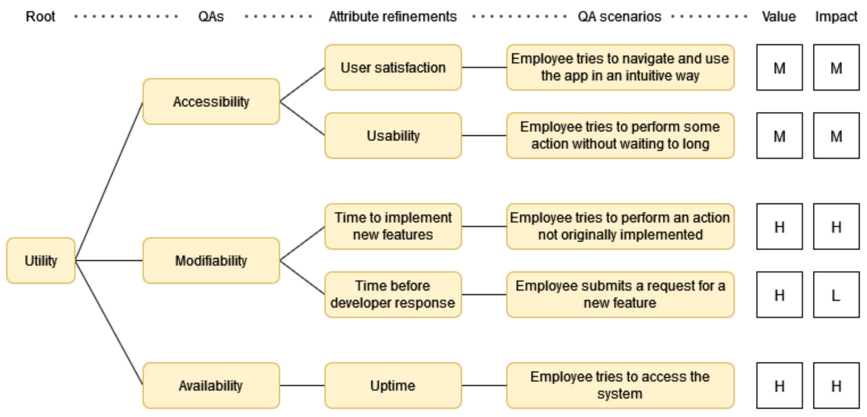

Root
QAs
Attribute refinements
QA scenarios
Value
Impact
Utility
Accessibility
User satisfaction
Employee tries to navigate and use the app in an intuitive way
M
M
Usability
Employee tries to perform some action without waiting to long
M
M
Modifiability
Time to implement new features
Employee tries to perform an action not originally implemented
H
H
Time before developer response
Employee submits a request for a new feature
H
L
Availability
Uptime
Employee tries to access the system
H
H

Figure 29: ASRs for the healthcare providers represented as a utility tree.

7.2 Assesment for Informal caregivers

Stakeholder	Informal caregivers
Stakeholder ID	SH2-IC
First scenario Diagram	See Figure 30
Rationale for Reliability	In this scenario, the focus is upon reliability, and this can be achieved by accomplish two QAs. First, is quality over time by partitioning the code in the software and by stability of the technologies used when developing the software. Second, is availability by implementing the whole infrastructure as a distributed system. Partitioning the code is very important because if informal caregivers report a problem with the software in which a certain step did not work, the developers should be able to find the script responsible with that and fix it. On a business value side, it is on a medium priority because it would be nice to have this, but if this is omitted the software will not fail. On an architectural impact level side, it is on a low priority because partitioning the code inside the software will not affect the architecture of it over-all, it will not introduce new APIs. Also, stability counts, because upon developing the software is important to have stable version of technologies that are being used. Besides that, having technologies that support each other is a must, for example: Google Spanner is not supported by Apache Kudu. On a business value side, it is on a high priority because this is a must, without having technologies working harmonious with each others, will result a reduced number of users. On an architectural impact level side, it is on a medium priority, because some technologies might require additional steps that can modify the architecture. Having a distributed system in order to support availability is essential, because this will offer two things: checkpoint mechanism, fast connectivity. The checkpoint mechanism will be implemented between multiple servers, for example: if a request cannot be processed due to failure from a request server, that request will be transferred to the next available server and so on. On a business value side and on an architectural impact level one, it is on a high priority because it is mandatory to have it, also, it can change the whole architecture.
Second scenario Diagram	See Figure 31

Rationale for Accessibility	In this scenario, the focus is on accessibility, and this can be achieved by accomplish two QAs. First, is ease of use of the software by taking into account the user's feedback and by having call center operators to help the users if they do not know how to do something in the software. Second, is the modularity of the software and by that it means that the software should be both in a web application form and mobile application one. Users' feedback is important and it is mandatory to be taken into account, because they might make some steps more easier for example: it is too hard to create an account. On a business value side, it is on a high priority. On an architectural impact level side is on a medium priority. Also, in order to achieve ease of use, it is necessarily to have call center operators that help the users when they have trouble using the software. For example: if a user stays more than 5 minute in the software without doing nothing, a message will pop-up and will announce to user that they can enter in a voice call (in the application) with an operator that can help them. On a business value side it is on a low priority but on an achitectural impact level is on a high priority. Having the software modular is a need nowadays because of the vast market in electronic devices such as: laptops, personal computers and smartphones. If the software is in a form of web application, then it can be used by users that might have a less performant device: for example: a smarthphone that was released 5 years ago, not forcing the users to upgrade their devices in order to use the software. Both on a business value side and on an architectural impact level one is on a high priority. The mobile application is important because nowadays people are using more their smarthphones than their computers. Also, it is more easier to access the software via a mobile application by making one tap on an icon than to go to browser and type the desired URL address of the software. Also, not to force users to buy new smarthphones because the app requires a lot of resources, it was decided that the mobile app will be in fact just a webview of the web application. On a business value side is a medium priority as well as for the architectural impact side.
Sensitivity points	1. Developers might not always be looking into the users feedback. 2. Developing a Distributed System requires a high budget. 3. Users might not always provide feedback after ending an action in the software. 4. Call centers operators might not always be available to help the users. 5. Developers might not always partitioning their code when implementing new features or improvements for the application.
Tradeoffs	1. The Mobile Application, because it is implemented as a webview in order to increase the performance on older devices, for some users, the UI/UX elements might not have the proper placement in the app, depending on their devices resolution. 2. Developing a Distributed System in order to increase the availability of the software, might create big expenses from a budget point of view. 3. In order to achieve quality over time by delevoping the software only with technologies that are compatible between them, some of the technologies might be outdated, having better alternatives that are not compatible with the current ecosystem.

Risks	1. If the Distributed System goes down for a maintenance or it has a major fault, the whole software will be unusable. 2. If the developers do not respect the same ”name coding”, the scripts behind the software will be hard to understand, risking that if a problem appears, developers might not know what script is responsible for that problem. 3. Not having all the requested features because some of them might be implemented with a technology that is not compatible with the technologies ecosystem that the software is built on. 4. If a doctor do not use the software, there might be a barrier between the patient that is using it and that doctor.
Non-Risks	1. The software will never be overloaded because behind it there is a whole distributed system responsible with avoiding that. 2. The users do not need to worry that the software will not run on their personal devices. 3. The users do not need to worry that they will not know how to use the software because beside the fact that the software is easy to use, they have the possibility to be helped by call center operators.

7.2.1 ASR 1 as Utility Tree: Reliability

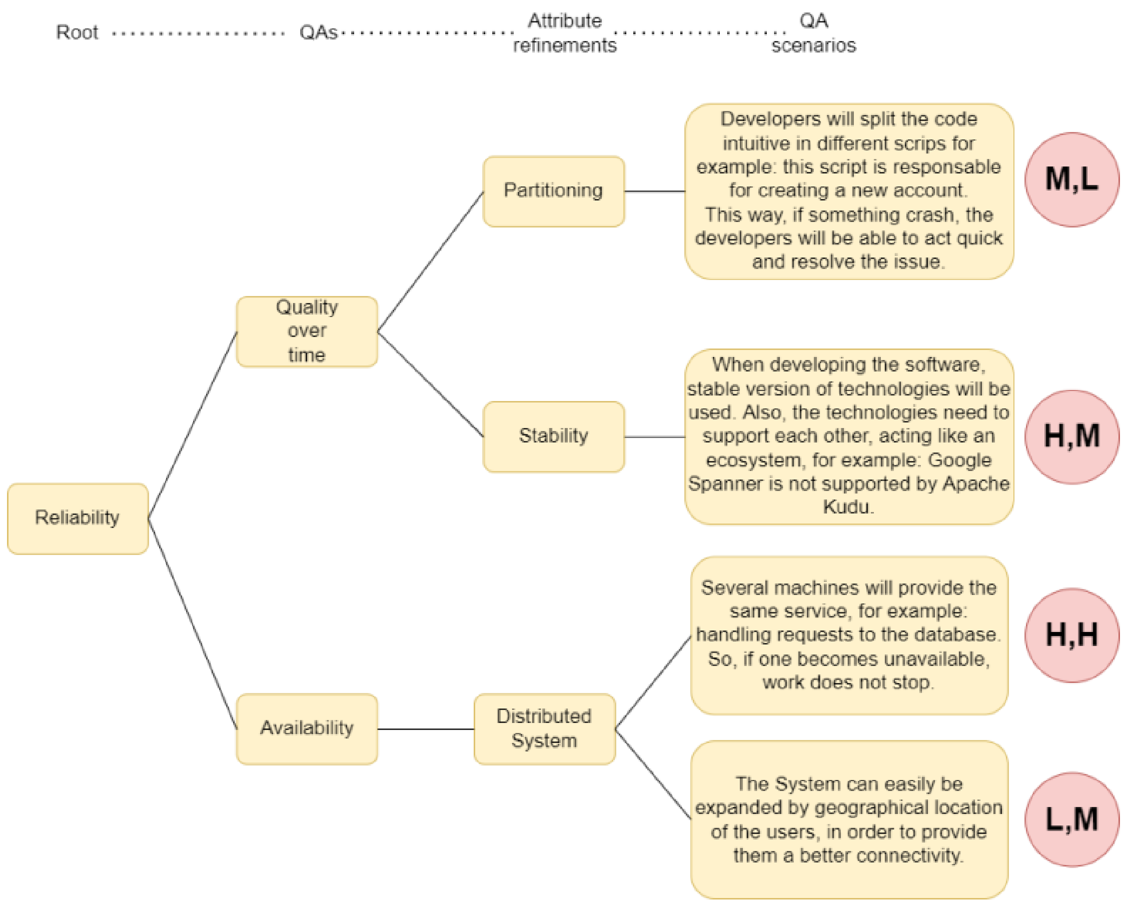

Root
QAs
Attribute refinements
QA scenarios
Reliability
Quality over time
Availability
Partitioning
Stability
Distributed System
Developers will split the code intuitive in different scrips for example: this script is responsable for creating a new account. This way, if something crash, the developers will be able to act quick and resolve the issue.
M,L
When developing the software, stable version of technologies will be used. Also, the technologies need to support each other, acting like an ecosystem, for example: Google Spanner is not supported by Apache Kudu.
H,M
Several machines will provide the same service, for example: handling requests to the database. So, if one becomes unavailable, work does not stop.
H,H
The System can easily be expanded by geographical location of the users, in order to provide them a better connectivity.
L,M

Figure 30: Reliability as an Utility Tree

7.2.2 ASR 2 as Utility Tree: Accessibility

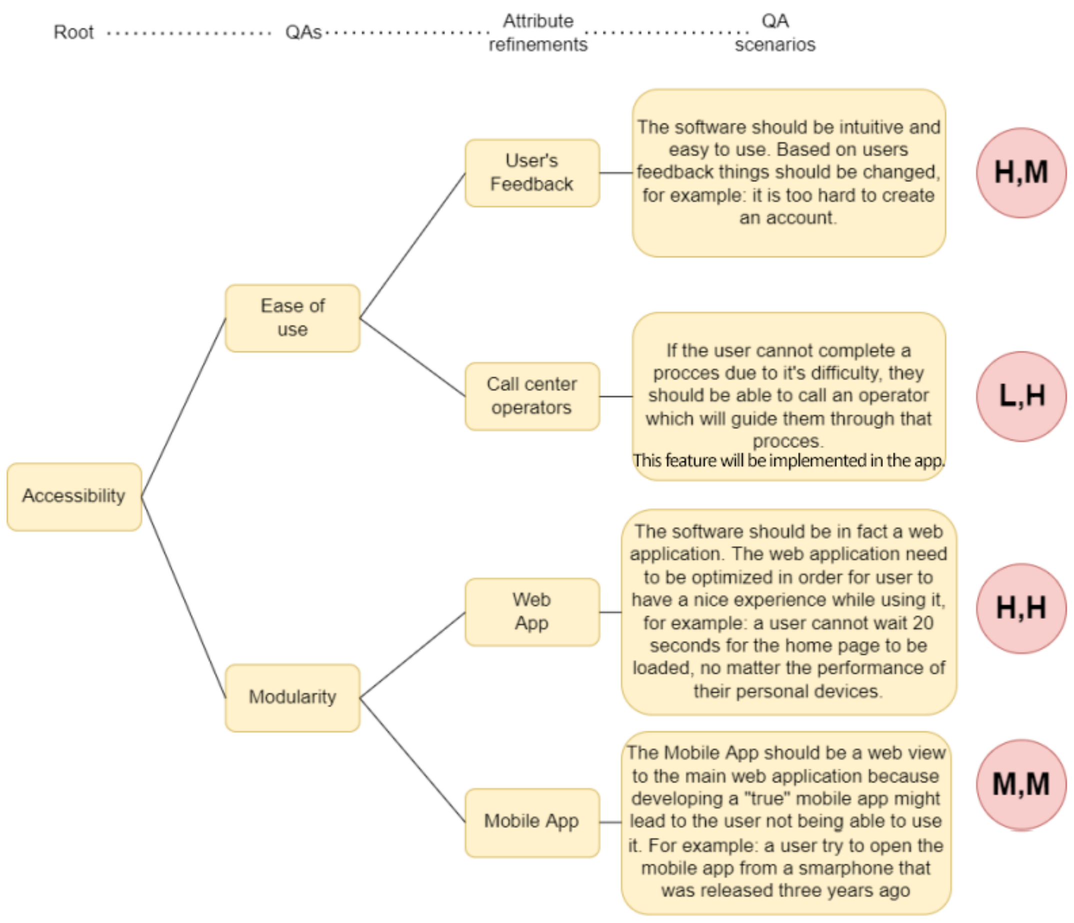

Root
QAs
Attribute refinements
QA scenarios
Accessibility
Ease of use
Modularity
User's Feedback
Call center operators
Web App
Mobile App
The software should be intuitive and easy to use. Based on users feedback things should be changed, for example: it is too hard to create an account.
If the user cannot complete a procces due to it's difficulty, they should be able to call an operator which will guide them through that procces.
This feature will be implemented in the app.
The software should be in fact a web application. The web application need to be optimized in order for user to have a nice experience while using it, for example: a user cannot wait 20 seconds for the home page to be loaded, no matter the performance of their personal devices.
The Mobile App should be a web view to the main web application because developing a "true" mobile app might lead to the user not being able to use it. For example: a user try to open the mobile app from a smarphone that was released three years ago
H,M
L,H
H,H
M,M

Figure 31: Accessibility as an Utility Tree

7.3 Assesment for Patients

Stakeholder	Patients
Stakeholder ID	SH3-P
Utility tree	See Figure 34
Rationale performance	In this scenario, the focus is upon Performance, and this can be achieved by accomplishing two QAs. The first one is complexity. The more one system is complex the more it can have breach in the performance. It is important the system to be designed in such a way that the complexity of the system is its own advantage. Having variety of classes and relationships between them makes it hard to track and here comes the second quality attribute which is modularity. It gives to the system independence and it makes the system easy for maintenance.
Rationale security	In this scenario, the focus is upon Security, and this can be achieved by accomplishing two QAs. First, user security should be available. This can be achieved by designing the system with two-steps of authentication, then the access to the different parts of the system should be strictly separated. Secondly, system security should be present. Storing the data with encryption, securing the REST API and the access of third party systems, as well as using a verified cloud storage provider is crucial part of designing a secure system.
Sensitivity points	1. Users may use unsecured networks. 2. Developers might not always make their code modular when implementing new features or improvements for the application 3. Developers might encrease the complexity of the system with some dependencies between the classes 4. Developers might not always make their code secure enough and cause security breaches 5. Users may use already leaked passwords and data.
Tradeoffs	1. Quality over quantity or the opposite is a well-known question when it comes to building a software. In our particular case the focus os going to be over the quality of the system. 2. Performance over quality sometimes occures as a tradeoff, because it is possible to improve the performance, but worsen the code and introduce other bugs. In our system modularity and handling the complexity is a priority 3. Using third party storage provider can cause security issues as well as performance issues, but it improves the availability for large data.
Risks	1. If the patient is not educated, does not have his own device or just does not want to use the software, the doctor cannot get in touch with the patient. The opposite situation also apply. 2. If the dependence between modules is not managed and one of them fails, the dependent one will fail too. 3. If the developers are not careful regarding the security, it can cause doors for breach 4. The data of the patient, being on a third party cloud provider is always a risk, regarding the unability of the developers of the given system to manage it.
Non-Risks	1. The flow will be provided fast and secure while using the application. 2. The application will be designed to have a secure environmet with a fast performance due to the modularity. 3. Patient data will be stored securely.

7.3.1 Utility Tree: Performance and Security

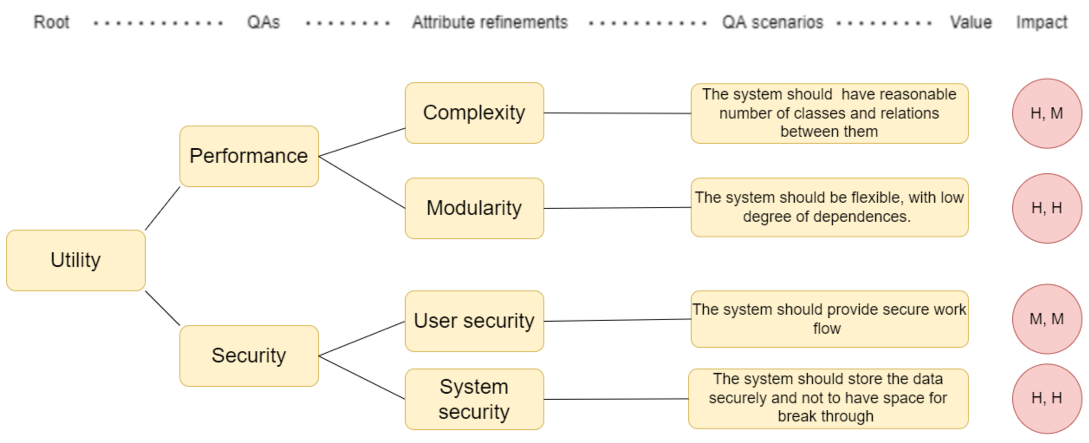

Root
QAs
Attribute refinements
QA scenarios
Value
Impact
Utility
Performance
Security
Complexity
Modularity
User security
System security
The system should have reasonable number of classes and relations between them
The system should be flexible, with low degree of dependences.
The system should provide secure work flow
The system should store the data securely and not to have space for break through
H, M
H, H
M, M
H, H

Figure 32: ASRs for the Patients represented as a utility tree

7.4 Assesment for Payers/Insurers

Stakeholder	Payers and Insurers
Stakeholder ID	SH4-PI
Scenarios Diagram	See Figure 33
Rationale Availability	In this scenario, the focus is on Software uptime. An employee at an insurance company wants to access patient data on their computer. For this the system must be accessible at least 99.99% of the time when such a request would be usually made during normal operation. The priority of this scenario is High because the system must be accessible for every stakeholder when they want to make a request, moreover it has a high impact because many parts of the system contribute to this significant requirement.
Rationale Reliability	In this scenario, the focus is on Recoverability, once in case of an unforseen shutdown, interruption or a failure or other problem the system can recover data affected and re-establish 99.99% of the the desired state of the system. The priority of this scenario is medium because during the normal operation of the system it shouldn't be a constant fear that such interruption would happen, although when it happens it is of high priority to handle the situation with the highest efficiency. This requirement has a high impact because whole system parts must be designed keeping backups, and other security measures in mind.
Sensitivity points	1. If an insurer wants to access the system outside of working ours they should also access everything they need, even if other parts of the system are under maintenance. 2. Insurers might want to access, especially from the border regions Healthcare providers data which is not compatible with the system. In this case this data should also be recovered after a shutdown, which might need to resincronise the systems. 3. The data could take time to fully recover, in case of insurers outside of business hours its low priority however they cannot operate effectively if it happens during worktime. 4. The insurers might use the software making too many requests thus placing high load on the servers

Tradeoffs	1. Using Databases from cloud providers to ensure extra storage safety raises some restrictions which impacts the performance of requests and the interoperability. It also takes the responsibilty out of the hands of the system, for this additional security data storage units should be used. 2. Operating a checkpoint mechanism system requires more maintenance and constant development.
Risks	1. The system is highly dependent on the correct maintenance and high interoperability. 2. In case of a problem when the usual high availabilty of the system goes down the insurers might not be able to do their usual work. 3. Every transaction should be retraceable on the system, in case of a file becomes damaged by a system failure it's possible that lawsuits will be filed.
Non-Risks	1. The system will have a high availability for the insurers in the significant timeframes. 2. The payment data will be safer when secured against system balckouts and unforseen interruption. 3. The users will get notifications about the failure, this way they can use their time for other activites rather than trying to figure out what is wrong with their application.

7.4.1 Utility Tree: Availability, interoperability and reliability

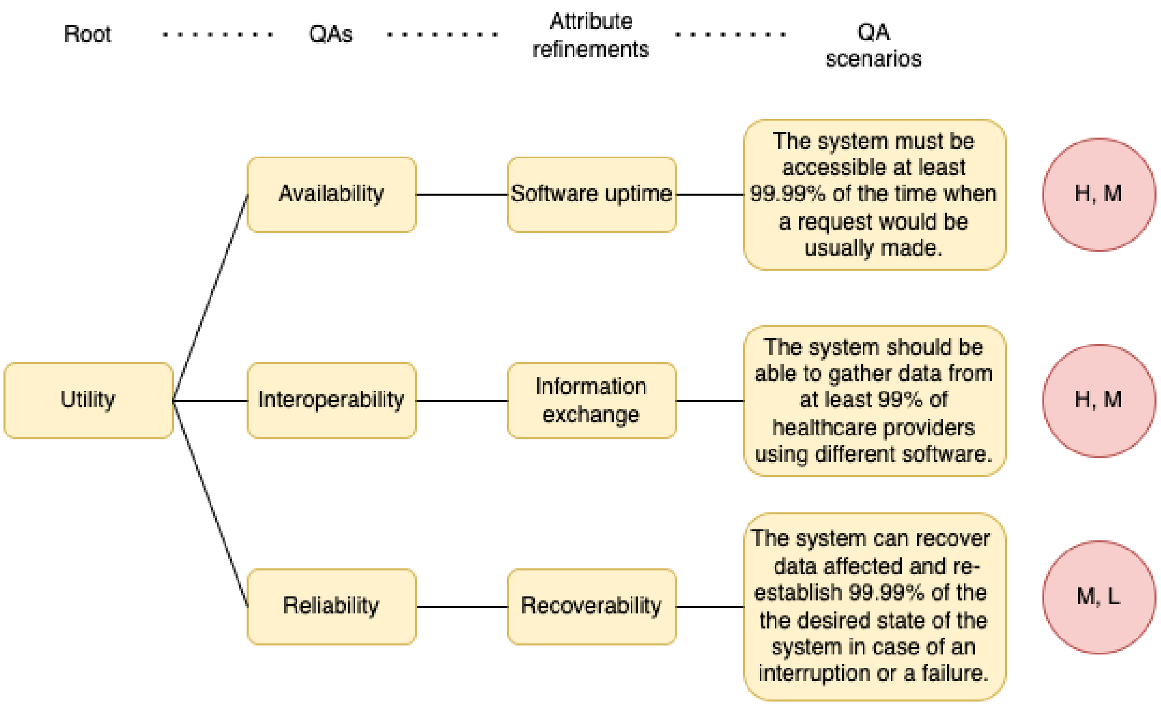

Root
QAs
Attribute refinements
QA scenarios
Utility
Availability
Software uptime
The system must be accessible at least 99.99% of the time when a request would be usually made.
H, M
Interoperability
Information exchange
The system should be able to gather data from at least 99% of healthcare providers using different software.
H, M
Reliability
Recoverability
The system can recover data affected and re-establish 99.99% of the the desired state of the system in case of an interruption or a failure.
M, L

Figure 33: ASRs for the Payers and Insurers represented as a utility tree

7.5 Assesment for Privacy Protection Agency

Stakeholder	Privacy Protection Agency
Stakeholder ID	SH5-PPA
Utility tree	See Figure 34
Rationale relaibility	The system must be reliable for all users and in itself. Reliability is important for users and system operation. It should be designed by system architects for users to go through verifications while logging into the system, and to do this safely and stably while transacting in the system using methods such as authentication, authorization and encryption. In addition, the design and creation of the system as a reliable and stable environment within itself is an important point in terms of reliability.
Rationale privacy	There are two important topics that the system should provide privacy and they are information and general system privacy. Protection of information is an issue that can affect all users in the system from different aspects. Confidentiality must be protected in cases such as entering a patient's information into the system, accessing existing information in the system, or making changes to the data in the system. And privacy must be ensured in the entire system, in data inputs and outputs, and in the communication channels to which the system is connected, and data must not leak.
Sensitivity points	1. While making any use on the system, disconnections may occur and data may be lost. 2. Users may try to log in from unsecured networks. 3. Users may lose app information or not pay attention. 4. Hardware-related problems may occur. 5. The data entered into the system may be lost or leaked by users without attention. 6. Data may not be accessible due to a malfunction in the system.
Tradeoffs	1. The possibility of prioritizing the usability and rapid implementation of the system for a faster transition process, but as a very important and indeed critical point, the possibility of not giving priority to reliability and confidentiality at a sufficient level, taking into account all possible situations and taking the necessary precautions in advance. 2. In-depth studies and restrictions may be required for system reliability and confidentiality, and a detailed examination of the assignment in this regard may be a situation that may affect reliability in the future, but this subject should be skipped because there may be some information restrictions in this area or situations that may make it difficult to use in the application. 3. In cases where the program does not allocate a large budget to each area, but the system architecture encounters unforeseen situations, there may be a possibility that it will need officials with different expertise.
Risks	1. If there is a software or hardware-related deterioration in the system, the whole system becomes inaccessible. 2. If a problem occurs during the actions taken in the system, it causes disruptions in use and reliability problems. 3. If the system grows too much, the possibility of using more hardware and increasing environmentally harmful waste (increased server usage, etc.) 4. Occurrence of privacy problems that may affect all users due to virus or other attacks on the system.

Non-Risks	1. The flow will be provided properly and reliably while using the application. 2. The application will be designed to have a reliable environmet in itself and will continue to provide this except for extreme events. 3. User information, patient information, information entered into the system and information available in the system will be kept confidential and necessary precautions will be taken.

7.5.1 Utility Tree: Reliability and Privacy

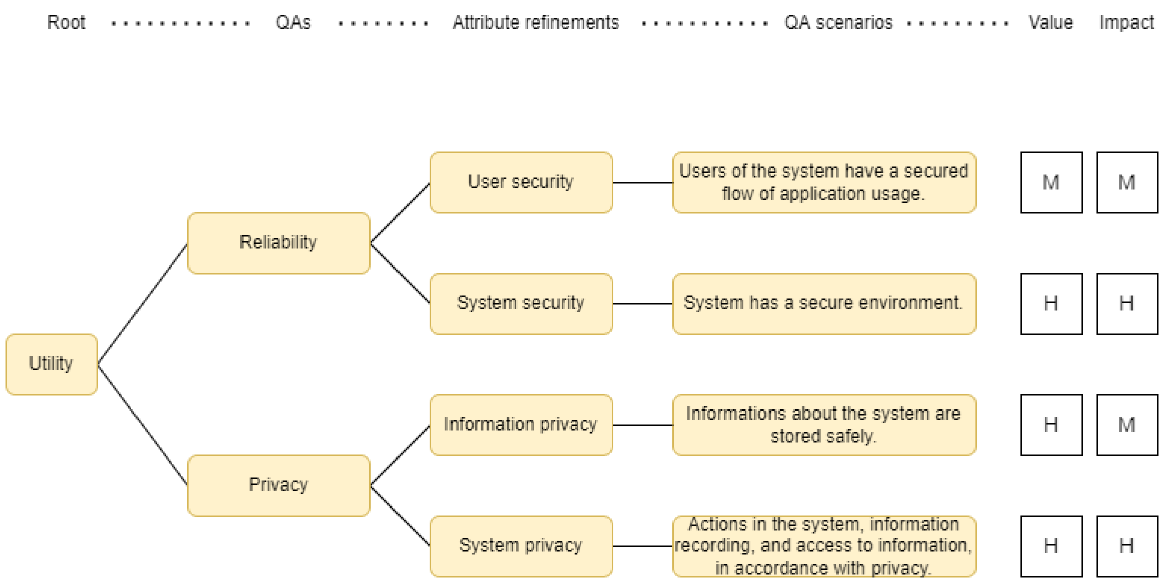

Root
QAs
Attribute refinements
QA scenarios
Value
Impact
Utility
Reliability
Privacy
User security
System security
Information privacy
System privacy
Users of the system have a secured flow of application usage.
System has a secure environment.
Informations about the system are stored safely.
Actions in the system, information recording, and access to information, in accordance with privacy.
M
M
H
H
H
M
H
H

Figure 34: ASRs for the Privacy Protected Agency represented as a utility tree

8 Glossary

API Application Programmable Interface. APIs are the main method of communication between different applications or between parts of the same application. This communication can take place over the internet or on the same device.

ASR Architecturally Significant Requirement. An ASR is a functional or non-functional requirement for the HealthSuite that is of particular importance to a stakeholder.

build time When the system is being developed, this is typically used in contrast to *configuration time* and *runtime*.

business goal A specific goal for a particular stakeholder, summarizing what they want to achieve using this system, formulated in a detailed manner that leaves minimal room for interpretation.

carbon footprint A metric to measure greenhouse gas (CO2 equivalent) emissions, which gives an indication about the environmental impact.

care path A process of healthcare activities, consisting of interrelated steps that often occur together because they are a healthcare providers' typical way of handling a specific situation. For instance, the process of getting diagnosed and treated for a certain type of cancer, including all scans, appointments, doctor's visits, regular check-ups afterwards, etc.

configuration time When the system is being configured, this is typically used in contrast to *build time* and *runtime*.

E-Smart Sensor A wearable device that can monitor heart rate and inform the health care provider of potential health problems via the HealthSuite.

encryption/decryption Using mathematical algorithms to make sure data is unusable unless the key to unlock it is also known. This is recommended for sensitive data such as passwords or medical records.

GDPR General Data Protection Regulation. A regulation specifiying data protection and privacy rules for all countries in the EU.

goal-oriented requirements graph A type of graph that can be used to intuitively visualize the relations between different requirements and goals for an application from a specific stakeholders' point of view.

information advantage A competitive advantage of a company that is directly caused by the information available to them which is not available to the competition.

metamodel A diagram indicating the various elements in a viewpoint and the relations between them. This serves as a point of reference for understanding the views based on the given viewpoint.

native application An application that is written specifically for the device it will be running on. This is in contrast to cross-platform applications, that are written in a general way so they can run on multiple platforms.

penetration test A common method to test software security: an authorized hacker tries to gain access to the system in order to identify vulnerabilities that need to be fixed.

QA Quality Attribute. A non-functional requirement of the system that is important to a stakeholder.

QA scenario Quality Attribute scenario. Quality attribute scenarios are a standardized format to clarify the context and properties of an *ASR* and make it more specific.

rationale The reasoning behind a decision taken as part of the design process. Since the requirements and the environment may change over time, it is important to keep track of the ideas behind certain decisions, so decisions can be retaken while also taking the initial reasoning into account.

runtime When the system is in active use, this is typically used in contrast to *build time* and *configuration time*.

SLA Service Level Agreement. A contract between the provider of a service, including non-functional requirements such as availability and performance specifications. Usually there are (financial) consequences to not delivering the promised service level.

stakeholder A third party that has a stake in the development or use of the system. This can be a corporation, a government, a non-profit foundation, any other kind of legal entity or even an abstract group of people, represented by a spokesperson or committee.

time to first paint The time it takes a web application to show something on the screen, a metric typically used to measure how responsive an application feels to users.

utility tree A diagram that illustrates how the QA scenarios contribute to the utility of the application from a stakeholder's perspective.

viewpoint An abstract description of a view, providing all the necessary information to explain the view and define its rules and uses.

view Views are visualizations that aim to illustrate one or more primary concerns of stakeholders in a more intuitive way, typically in the form of a diagram. They are always based on a viewpoint that explains what it means and why it's useful.

E-Smart Sensor A mechanism that is capable of assign an exact number of machines for required workload/requests.

ID Unique naming or number for identifing a process. For example: a view, a stakeholder, a viewpoint, etc.

Software A program used to operate computers and execute specific tasks.