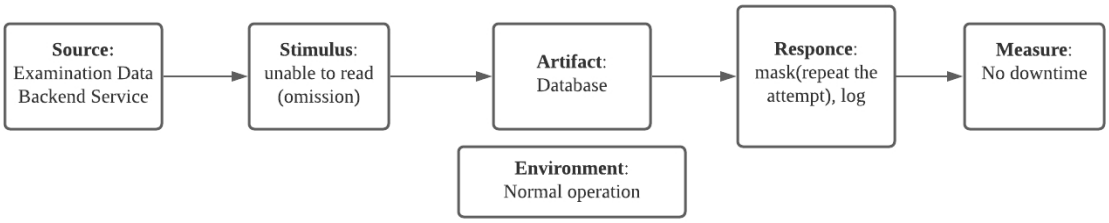
Availability



1. **Source**: The ExaminationData Backend Service which communicates with Database and Public API Gateway.
2. **Stimulus**: When new data is being processed, the ExaminationData Backend is not able to read the data from the Database for the first time.
3. **Artifact**: The main database which saves all the data.
4. **Response**: The system needs to log this fault and repeat the attempt to read data from the Database.
5. **Measure**: The system should not take any downtime to log and do another attempt.

The architecture of the system is able to repeat the reading operation and log the error message.

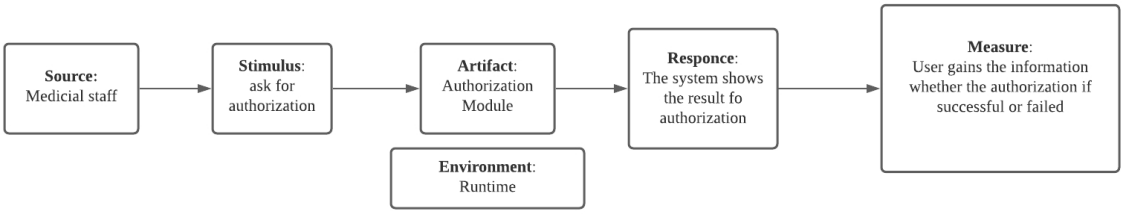
If we consider applying the recovery tactics within no downtime of the system, the system needs some improvements. From the architecture, it only has one Database for saving all of the data, this will be hard to prevent not losing the data when this Database is not readable and needs time for recovery.

The improvement will be adding another Backup Database. This Backup Database will communicate with the Database and ExaminationData Backend Service. The Backup Database always holds the most recent data in the last 24 hours. The Backup Database will not be too large. If the main Database is not allowed for reading, the system will use the Backup Database instead. In this way, we will need no downtime of the system when the Database is not available. After the Database is recovered, the system can still move the previous data from the Backup Database into the Database.

The whole process will be, when the Database is not available for reading, the error message will be logged to the Backup Database, then the system will try to repeat the read operations for two more attempts. If both attempts fail, the system will read the data from the Backup Database until the Database has recovered. After the recovery, the Backup Database will send the data to the Database, so that the Database will not lose data in the recovery process.

One issue will be that, if the Database needs an unexpectedly long time to be recovered, the load balance of the Backup Database needs to be considered more carefully.

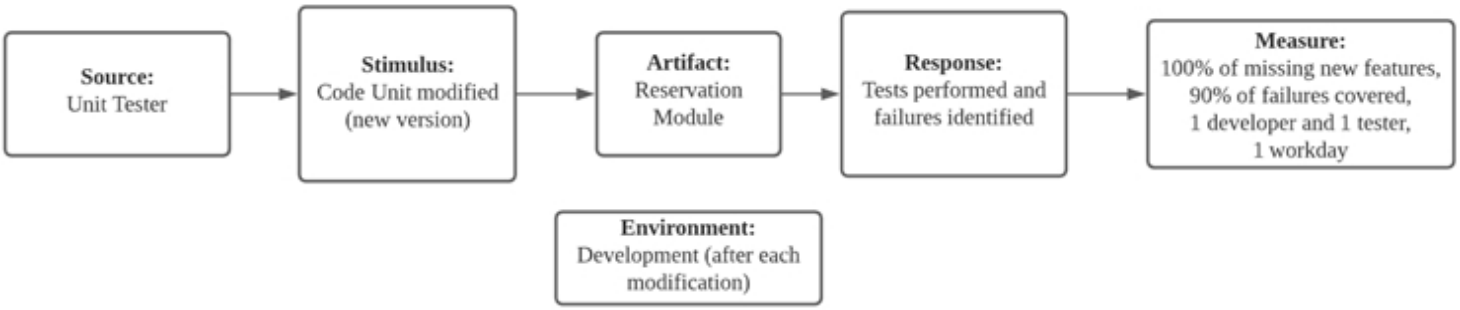
Usability



1. **Source**: The medical staff (doctors and nurses) who want to use the system for equipment reservation.
2. **Stimulus**: The process of authorization of the user.
3. **Artifact**: The authorization module.
4. **Response**: The system needs to show the result of authorization.
5. **Measure**: The user will gain information about the result of the authorization, both in the successful case and failure case.

The system fulfills the requirement, the authorization module can communicate with the API and GUI in both ways, so it will be easy to send the message to the GUI after the authorization finishes.

There is a minor improvement that can be done in the sequence diagram. When we look at the sequence diagram, only the failure message will be sent back to the GUI. The improvement will be simply adding a response message to the GUI if authorization is successful.



Testability

1. **Source:** The source of the stimulus is the Unit Tester in charge of testing the Reservation Module.

2. **Stimulus:** The stimulus is the completion of a new version of the Reservation Module.

3. **Artifact:** The artifact considered is the Reservation Module.

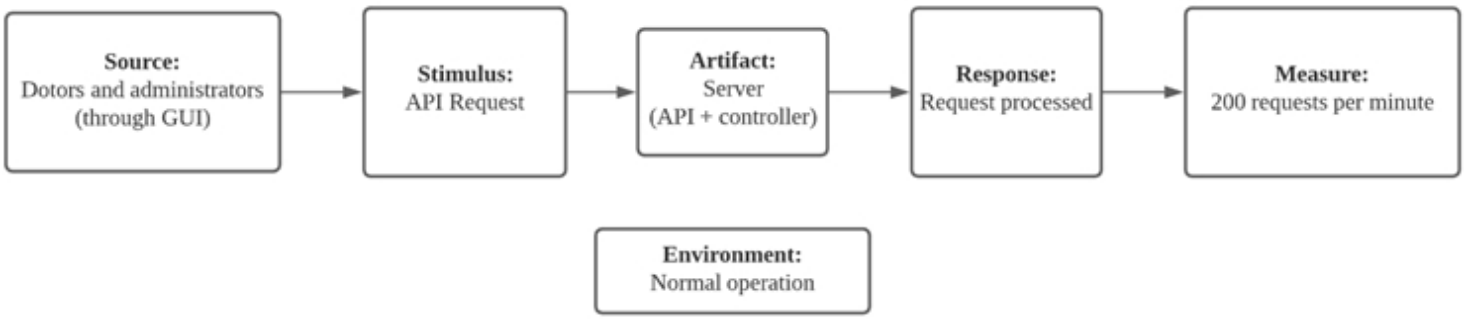
4. **Response:** New tests are written, and all tests are performed. Failures are easily identified.

5. **Measure:** All new features that are not implemented are identified, and 90% of the failures (both in new features and old ones) are identified. This must need the work of 1 developer and 1 tester during 1 workday.

This system architecture meets the requirement, as the Reservation Module is very well isolated, its inputs come exclusively from the GUI, and it only requests the Database for information. This means that the inputs can be easily simulated, and a mock Database can be used to test the different use cases.

Of course, some decisions will have to be made in future stages of the project (design and development) for this requirement to be fulfilled by the system, but this architecture makes those decisions possible.

Performance



1. **Source:** The source of the stimulus is the medical staff and the administrators. They request the information through the GUI module. It is an external source (it is not part of the system).

2. **Stimulus:** The stimulus is all Requests from doctors or administrators. Doctors' requests could be considered periodic, as doctors usually check examination results and make reservations before attending each patient, but as there are potentially many doctors and each of them can spend different amounts of time with each patient, it is more accurate to consider stochastic requests. Administrators' requests can be considered sporadic, as the equipment is added sporadically and statistics are usually checked once or twice a day.

3. **Artifact:** The artifact considered is the Server component (API + Controller).

4. **Response:** 1. The system processes all requests and responds to them.

5. **Measure:** The measure will be the throughput, as per the number of requests processed per minute.

To analyze if the system meets the requirement, we need to focus on the number of requests that can be fired. The Maintenance and the Statistics and Data modules are used sporadically by the system administrators. The Examination Data Module and the Reservation Module are used by doctors, with a stochastic arrival of requests (as mentioned above in the stimulus section).

The fulfillment of this requirement will be finally determined by the decisions made in the design and development phases. However, given that there are typically no more than several hundred doctors in the hospital, and that each doctor typically only requests information once or twice for every patient they dispatch (it takes 5 to 10 minutes per patient). This will mean that the system must process at most 200 requests per minute. If

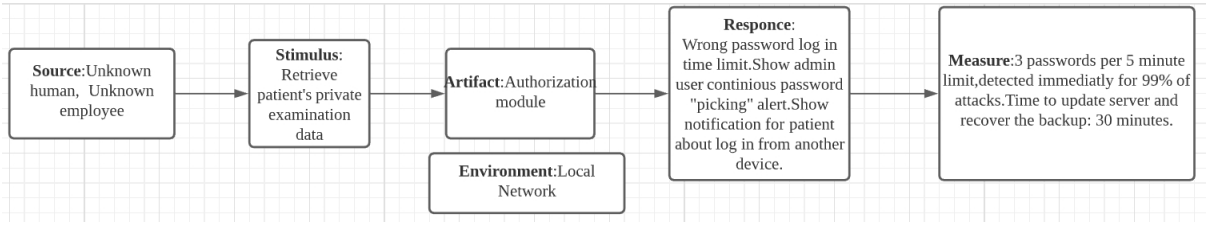
the right design and development decisions are made, this architecture allows for more than 200 requests to be processed per minute, so the requirement is fulfilled.

However, if the hospital grows and there are too many doctors, the server can be replicated, and a load balancer can be added to spread the requests across the multiple instances. This will only be useful if the Database module can process the requests from all server instances. Otherwise, similar tactics would need to be applied for the Database module.

If the system administrators need to make more requests (these requests take more time to be processed), and the system gets overloaded, priority could be given to doctors' requests, to ensure that their requests are responded in time.

These last appreciations, however, are tightly coupled with the modifiability attributes.

Security



Source: Unknown human(hacker) or unknown employee

Stimulus: Retrieve patient's private data in user database.

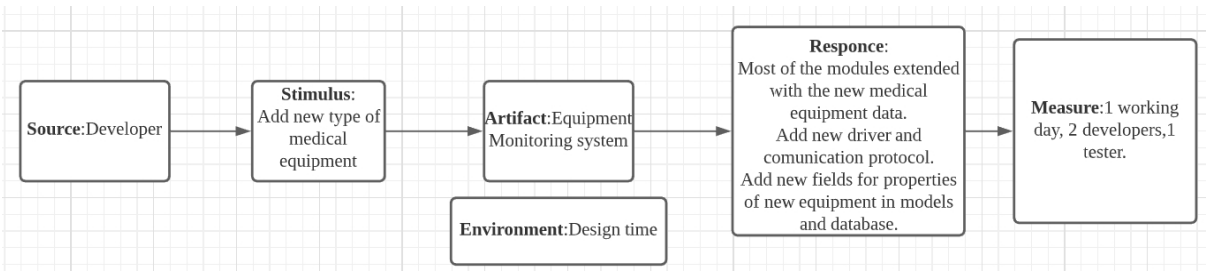
Artifact: The authorization module where the hacker will try to pick the right password for login to someone's patient account.

Response: First of all there is a limit for repetitive password entering. After exceeding the limit the admin user will get the alert showing that someone is trying to log in to the system exceeding the "password limit" and even if the hacker managed to get a login, the real patient will get notification by mail with the message that someone logged into their account from another device.

Measure: 3 passwords per 5-minute limit.

The system fulfills the requirement because it's easy to communicate from the Authorization module to user interface and admin interface.
The system should have some new functionality inside the authorization module such as check for the number of login attempts and notification module for sending the users alert about compromised data retrieving and suspicious login from the non-users device.

Modifiability



Source: Developer team is requested to add a new type of medical equipment.

Stimulus: Add new medical equipment(for example MRI)

Artifact: Equipment Monitoring System

Environment: The changes are made during the design time

Response: Add new device-specific communication protocol for the newly added type of medical equipment, new device driver. Also adding new examination data type in examination model and new properties in the equipment model.

Measure: Adding new medical equipment in all listed modules will take 1 working day. The work will be done by a team of software developers with 2 developers and 1 tester for testing the new features.

The system satisfies the quality attribute for adding new medical equipment since the list of main functionalities will stay unchanged and adding new features for new devices is completely optimized for this system.