Security Analysis
Vaccination Portal

Jana Farag, Peter Haidt

26.5.2024

MIM

1. Summarized Risk Evaluation

The vaccination portal application will have an impact on millions of patients and should be trusted to handle their sensitive and private patient information with great care. The risk not only comes from attackers wanting to compromise the system by obtaining any of the admin privileges but also an attacker being able to manipulate steal a patient's identity by obtaining their registration code or by making the application completely unavailable for legitimate users who need the vaccination appointments for the sake of their health protection. To ensure this does not happen, centralized role-based access control must be implemented and maintained according to the principle of least privilege. Also, federated identities should be used to save money, time, and ensure higher security of the login data flow and login credential storage. This will be the basis for confidentiality, authentication, and authorization within the application.

As this vaccination can be a matter of life and death it is a necessity to ensure high availability. DDoS protection, Load Balancing and Autoscaling of physical servers must be implemented and guaranteed by the hosting provider. The application must be run on enough resources to withstand the incoming traffic and any DDoS attempts. As this application is accessible to everyone on the internet, it is essential to have an ongoing and monitored unified threat management system, encapsulating all logging, monitoring, content filtering and anomaly detections. This should be supported by active methods to filter out attackers like ACLs and Firewalls configured with whitelists allowing needed protocols e.g. HTTP(S) and TCP.

To ensure the secure transfer of data over the internet (an unsecured network) and to avoid spoofing and tampering of internet traffic, data encryption in transit must be a top priority. Using TLS 1.3. over HTTP, can ensure the confidentiality and integrity of the data flows between the web browser (users / patients) and the webserver.

The basis of any highly available application is also the ability to deal with disasters (e.g. server damage or server outages). The security of the physical hardware and getting it to run again is transferred to the histing provider. However, it is still our responsibility to ensure there is a backup to recover the data from. The backups must be done regularly and multiple times as the to be discussed RPO must be met by ensuring at least one successful backup within the specified time. For integrity protection this backup will also be encrypted with strong encryption algorithms.

Finally, the basis of any anomaly detection, continuous security improvements and fast incident response is the logging and monitoring of the logs. For effective security, the alarm threshold must be reasonable and avoid false positives. Exact incident response plans with dedicated roles are also essential to fast disaster recovery.

As security is an iterative process, regular training and security audits must be set in place to maintain the level of security to a high standard.

All these layers of security follow the strategy of Defense in Depth to ensure that if one of these security approaches fails, that others are set in place to avoid attacks.

2. System Design

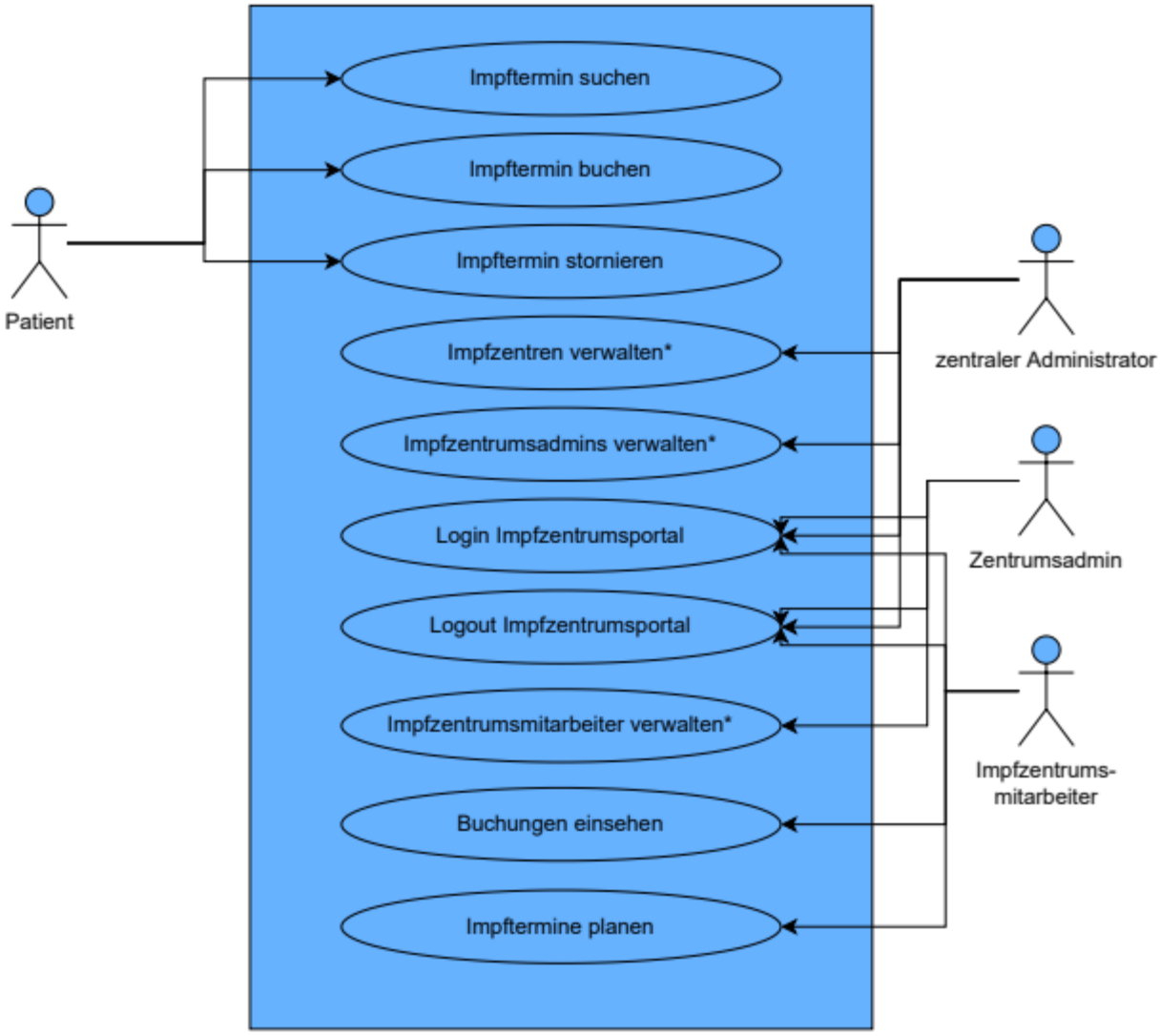
2.1 Summary of the system

An application for booking and managing vaccination appointments is to be developed for the federal state of Baden-Württemberg. A web application is to be provided with which both employees in the vaccination centers and patients can work. Approximately 60 vaccination centers for the 11 million inhabitants of Baden-Württemberg are to be connected

2.2 System requirements

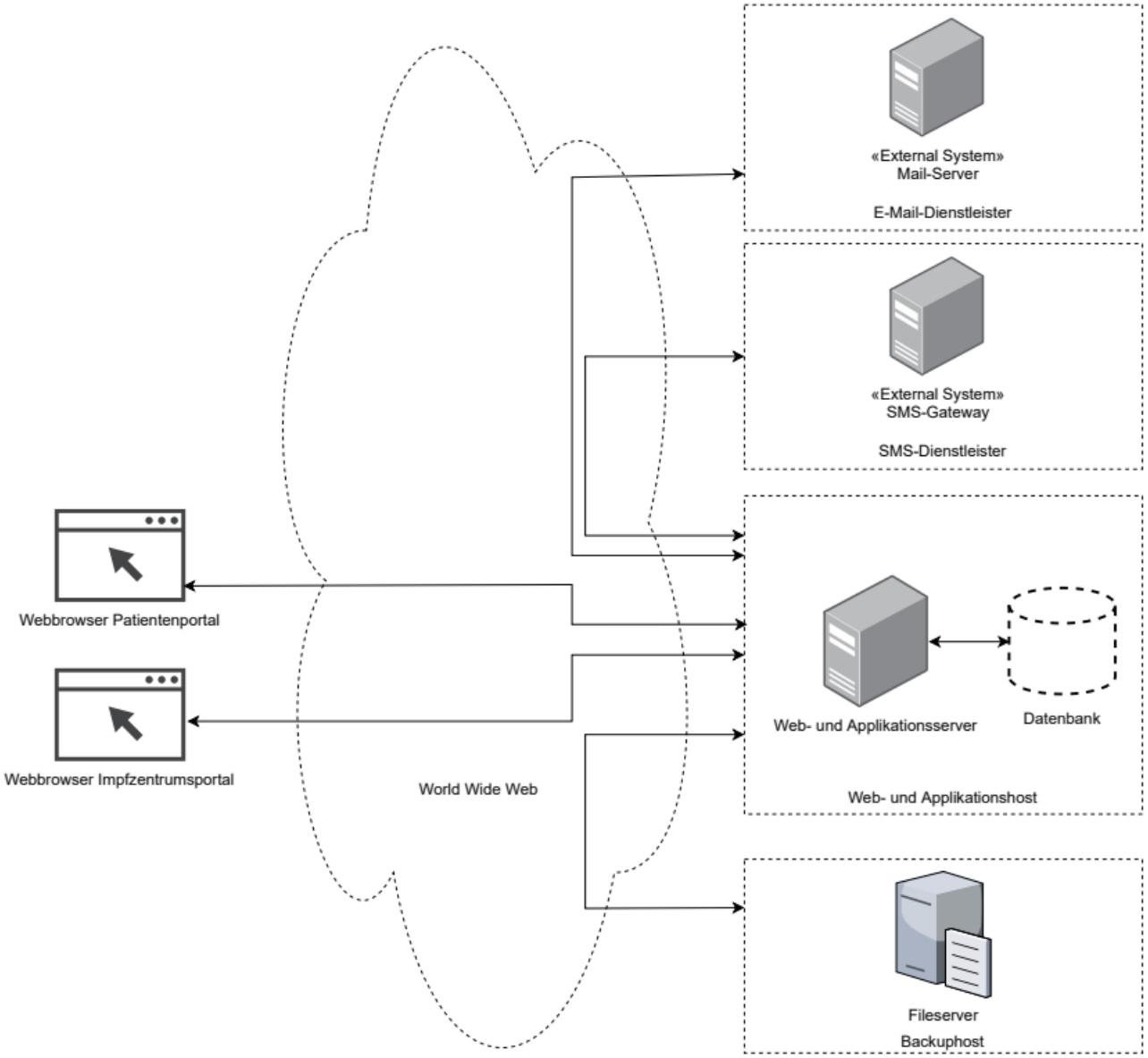
The system requirements can be found in the document "SSWEM vaccination portal description".

2.3 Use-Case Diagram



*Der Einfachheit halber werden hier erstellen, bearbeiten und löschen / deaktivieren zusammengefasst.

2.4 Architecture



3. Security analysis

3.1 Security- und Non-Requirements

Security Requirements

--> patient portal is defined as the patient portal application with its full stack.
--> vaccination center portal is defined as the vaccination center portal application with its full stack.

Identification Requirements:

- The **vaccination center portal** shall identify *vaccination center employees* before allowing them to use its functions.
- The **vaccination center portal** shall identify *vaccination center administrators* before allowing them to use its functions.
- The **vaccination center portal** shall identify the *central administrator* before allowing them to use its functions.
- The **patient portal** shall identify *registration codes* before allowing patients to search for appointments.
- The **patient portal** shall identify *patients* before allowing patients to book appointments.
-

Authentication Requirements:

- The **vaccination center portal** shall verify the identity of *vaccination center employees* before allowing them to use its functions.
- The **vaccination center portal** shall verify the identity of *vaccination center administrators* before allowing them to use its functions.
- The **vaccination center portal** shall verify the identity of the *central administrator* before allowing them to use its functions.
- The **patient portal** shall verify the identity of *patients* before allowing them to book appointments.
- The **mail-server** shall verify the identity of *senders* before allowing them to send a verification email.
- The **sms-gateway** shall verify the identity of *senders* before allowing them to send a verification sms.
- The **file-server** shall verify the identity of the *process* before allowing them to use backup functionality.

Authorization Requirements:

- The **vaccination center portal** shall not allow *vaccination center employees* to *change* appointment booking status.
- The **vaccination center portal** shall not allow *vaccination center employees* to *change* appointment booking information.

- The **vaccination center portal** shall allow each *vaccination center employee* to obtain access to appointment bookings.
- The **vaccination center portal** shall allow *vaccination center admins* to obtain access to employee information of the other vaccination centers.
- The **vaccination center portal** shall allow *vaccination center admins* to *change* employee information of their respective vaccination center.
- The **vaccination center portal** shall not allow the *central admin* to *access* vaccination center employee information.
- The **vaccination center portal** shall not allow the *central admin* to change vaccination center employee information.
- The **vaccination center portal** shall allow the *central admin* to *change* vaccination center information.
- The **vaccination center portal** shall allow the *central admin* to *change* the information of vaccination center admins.
- The **patient portal** shall allow *patients* to book appointments.
- The **patient portal** shall allow *patients* to cancel their appointments.
- The **patient portal** shall allow *patients* to access their booking information.
- The **patient portal** shall allow *patients* to access their personal information.
- The **patient portal** shall allow valid registration code holders to view appointments.
- The **email-server** shall allow the *application-server* to send emails.
- The **SMS-gateway** shall allow the *application-server* to send SMS.
- The **file-server** shall allow the *application-server* to use its backup functionality.

Immunity Requirements:

- The **patient portal** shall protect itself by validating the input data.
- The **patient portal** shall notify the *central administrator* if malware is detected during a scan.
- The **vaccination center portal** shall protect itself by validating the input data.
- The **vaccination center portal** shall notify the *central administrator* if malware is detected during a scan.
- The **patient portal** shall protect itself from DoS attacks.
- The **vaccination center portal** shall protect itself from DoS attacks.
- The **file-server** shall protect itself from DoS attacks.
- The **sms-gateway** shall protect itself from DoS attacks.

Integrity Requirements:

- The **vaccination center portal** shall prevent unauthorized corruption of employee data.
- The **vaccination center portal** shall prevent unauthorized corruption of vaccination center data.
- The **patient portal** shall prevent unauthorized corruption of patient data.
- The **patient portal** shall prevent unauthorized corruption of booking data.

- The **patient portal** shall prevent unauthorized corruption of available appointments.

Intrusion Detection Requirements:
- The **vaccination center portal** shall detect attempted accesses that fail identification requirements.
- The **vaccination center portal** shall detect attempted accesses that fail authorization requirements.
- The **vaccination center portal** shall detect attempted accesses that fail authentication requirements.
- The **vaccination center portal** shall notify the *central administrator* of all failed attempted accesses during the previous 24h.
- The **patient portal** shall detect attempted accesses that fail identification requirements.
- The **patient portal** shall detect attempted accesses that fail authorization requirements.
- The **patient portal** shall detect attempted accesses that fail authentication requirements.
- The **patient portal** shall notify the *central administrator* of all failed attempted accesses during the previous 24h.

Nonrepudiation Requirements:
- The **patient portal** shall store tamper-proof records of patient data.
- The **patient portal** shall store tamper-proof records of booking data.
- The **vaccination center portal** shall store tamper-proof records of employee information.

Privacy Requirements:
- The **patient portal** shall not allow unauthorized individuals access to bookings.
- The **patient portal** shall not allow unauthorized programs access to bookings.
- The **patient portal** shall not allow unauthorized individuals access to patient data.
- The **patient portal** shall not allow unauthorized programs access to patient data.
- The **vaccination center portal** should not allow unauthorized individuals access to bookings.
- The **vaccination center portal** shall not allow unauthorized programs access to bookings.
- The **vaccination center portal** shall not allow unauthorized individuals access to patient data.
- The **vaccination center portal** shall not allow unauthorized programs access to patient data.
- The **fileserver** shall not allow unauthorized individuals access to backups.

- The **SMS- Gateway** shall not allow unauthorized individuals access to SMSs.
- The **mail server** shall not allow unauthorized individuals access to emails.

Security Auditing Requirements:

- The **vaccination center portal** shall collect the status of its security mechanisms every Friday at 19:00h.
- The **vaccination center portal** shall organize the status of its security mechanisms every Saturday at 19:00h.
- The **vaccination center portal** shall summarize the status of its security mechanisms every Friday at 19:00h.
- The **vaccination center portal** shall report the status of its security mechanisms every Friday at 19:00h.
- The **patient portal** shall collect the status of its security mechanisms every Friday at 19:00h.
- The **patient portal** shall organize the status of its security mechanisms every Saturday at 19:00h.
- The **patient portal** shall summarize the status of its security mechanisms every Friday at 19:00h.
- The **patient portal** shall report the status of its security mechanisms every Friday at 19:00h.

Survivability Requirements:

- The **patient portal** shall not have a single point of failure.
- The **vaccination center portal** shall not have a single point of failure.

Physical Protection Requirements:

- The **patient portal** shall protect its hardware components from physical damage.
- The **patient portal** shall protect its hardware components from theft.
- The **patient portal** shall protect its hardware components from surreptious replacement.
- The **patient portal** shall protect its backup hardware components from physical damage.
- The **patient portal** shall protect its backup hardware components from theft.
- The **patient portal** shall protect its backup hardware components from surreptious replacement.
- The **patient portal server hardware** shall be physically separated from the **patient portal backup server hardware**.
- The **vaccination center portal** shall protect its hardware components from physical damage.
- The **vaccination center portal** shall protect its hardware components from theft.
- The **vaccination center portal** shall protect its hardware components from surreptious replacement.

- The **vaccination center portal** shall protect its backup hardware components from physical damage.
- The **vaccination center portal** shall protect its backup hardware components from theft.
- The **vaccination center portal** shall protect its backup hardware components from surreptious replacement.
- The **vaccination center portal server hardware** shall be physically separated from the **vaccination center portal backup server hardware**.
- The **file-server** shall protect its hardware components from physical damage.
- The **file-server** shall protect its hardware components from theft.
- The **file-server** shall protect its hardware components from surreptious replacement.
- The **SMS-gateway** shall protect its hardware components from physical damage.
- The **SMS-gateway** shall protect its hardware components from theft.
- The **SMS-gateway** shall protect its hardware components from surreptious replacement.
- The **email-server** shall protect its hardware components from physical damage.
- The **email-server** shall protect its hardware components from theft.
- The **email-server** shall protect its hardware components from surreptious replacement.

System Maintenance Security Requirements:

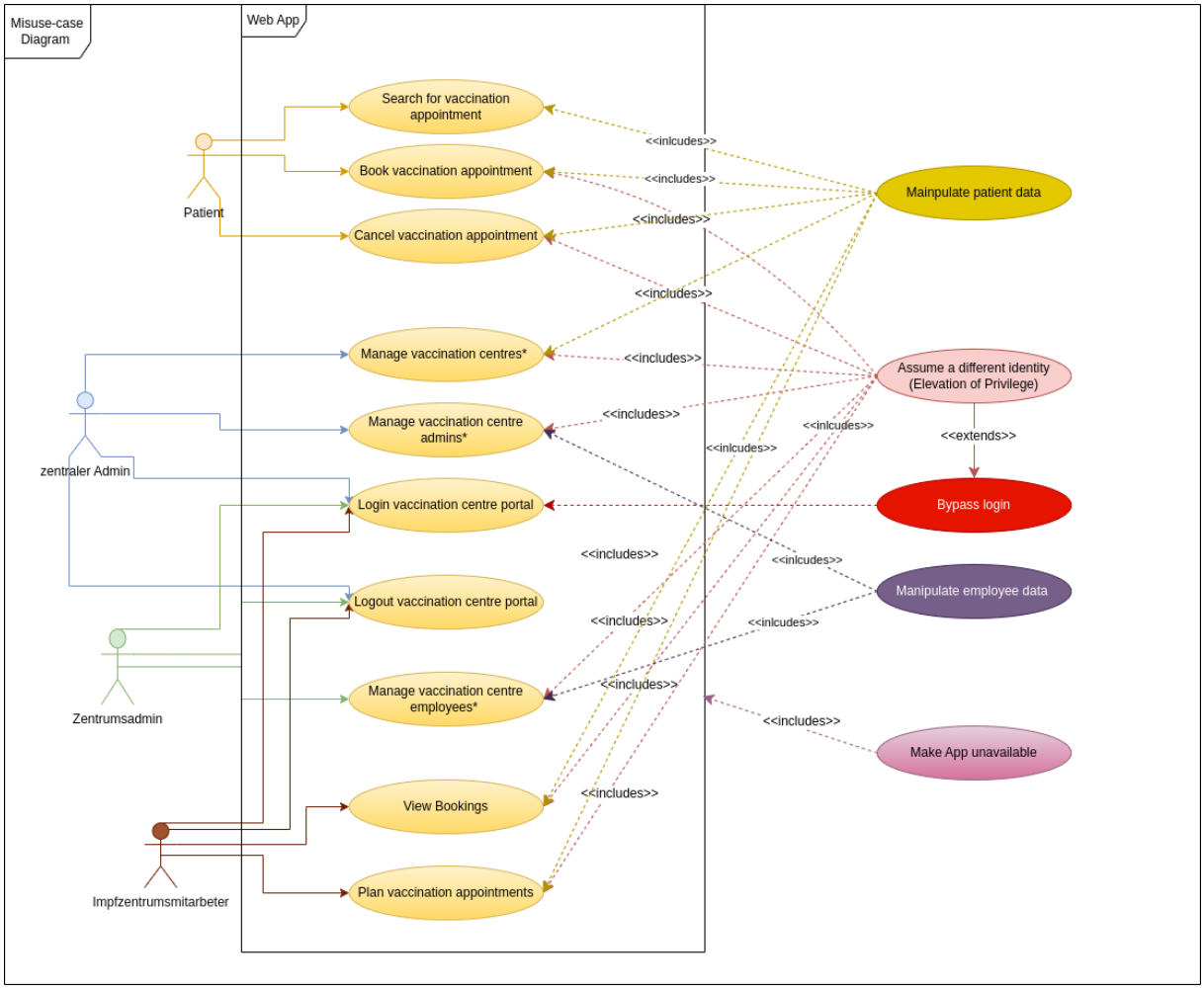
- The **patient portal** shall not violate its security requirements as a result of updating data.
- The **patient portal** shall not violate its security requirements as a result of updating hardware.
- The **patient portal** shall not violate its security requirements as a result of the updating of a software component.
- The **patient portal** shall not violate its security requirements as a result of replacement of data.
- The **patient portal** shall not violate its security requirements as a result of replacement of hardware.
- The **patient portal** shall not violate its security requirements as a result of replacement of a software component.
- The **vaccination center portal** shall not violate its security requirements as a result of updating data.
- The **vaccination center portal** shall not violate its security requirements as a result of updating hardware.
- The **vaccination center portal** shall not violate its security requirements as a result of the updating of a software component.
- The **vaccination center portal** shall not violate its security requirements as a result of replacement of data.

- The **vaccination center portal** shall not violate its security requirements as a result of replacement of hardware.
- The **vaccination center portal** shall not violate its security requirements as a result of replacement of a software component.
- The **patient portal** shall backup its data as a batch process every end of the day.
- The **patient portal** shall scan itself every day for known vulnerabilities.
- The **patient portal** shall notify the administrator if any known vulnerabilities are found.
- The **patient portal** shall scan itself every day for available component updates.
- The **patient portal** shall update outdated components every second Friday.
- The **patient portal** shall patch any identified vulnerabilities of its components within 24 hours.
- The **vaccination center portal** shall scan itself every day for known vulnerabilities.
- The **vaccination center portal** shall notify the administrator if any known vulnerabilities are found.
- The **vaccination center portal** shall scan itself every day for available component updates.
- The **vaccination center portal** shall update outdated components every second Friday.
- The **vaccination center portal** shall patch any identified vulnerabilities of its components within 24 hours.

Non-Requirements

- The **vaccination center portal** shall not protect against malicious administrators.
- The **vaccination center portal** shall not protect against malicious employees.
- The **patient portal** shall not protect against malicious administrators.
- The **patient portal** shall not protect against malicious employees.
- The **vaccination center portal** shall not protect against overpowered actors (ie. national intelligence agencies).
- The **patient portal** shall not protect against overpowered actors (ie. national intelligence agencies).
- The **patient portal** shall not protect against possible malware infections.
- The **vaccination center portal** shall not protect against user error (ie. due to phishing or social engineering).
- The **vaccination center portal** shall not protect against possible malware infections.

3.2 Misuse-Cases



*Der Einfachheit halber werden hier erstellen, bearbeiten und löschen / deaktivieren zusammengefasst

**Der Einfachheit halber werden hier alle Arten von Injection zusammengefasst (XSS, XPath, XXE, SQL)

Figure 1: Misuse Case – Overview

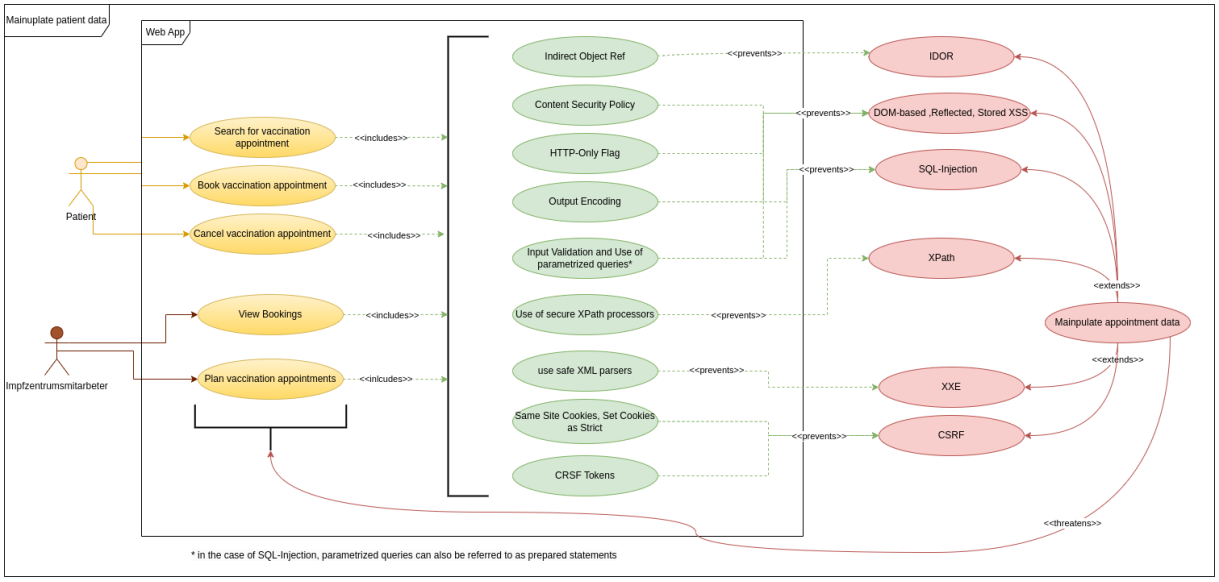


Figure 2: Misuse Case - Manipulate Patient Data

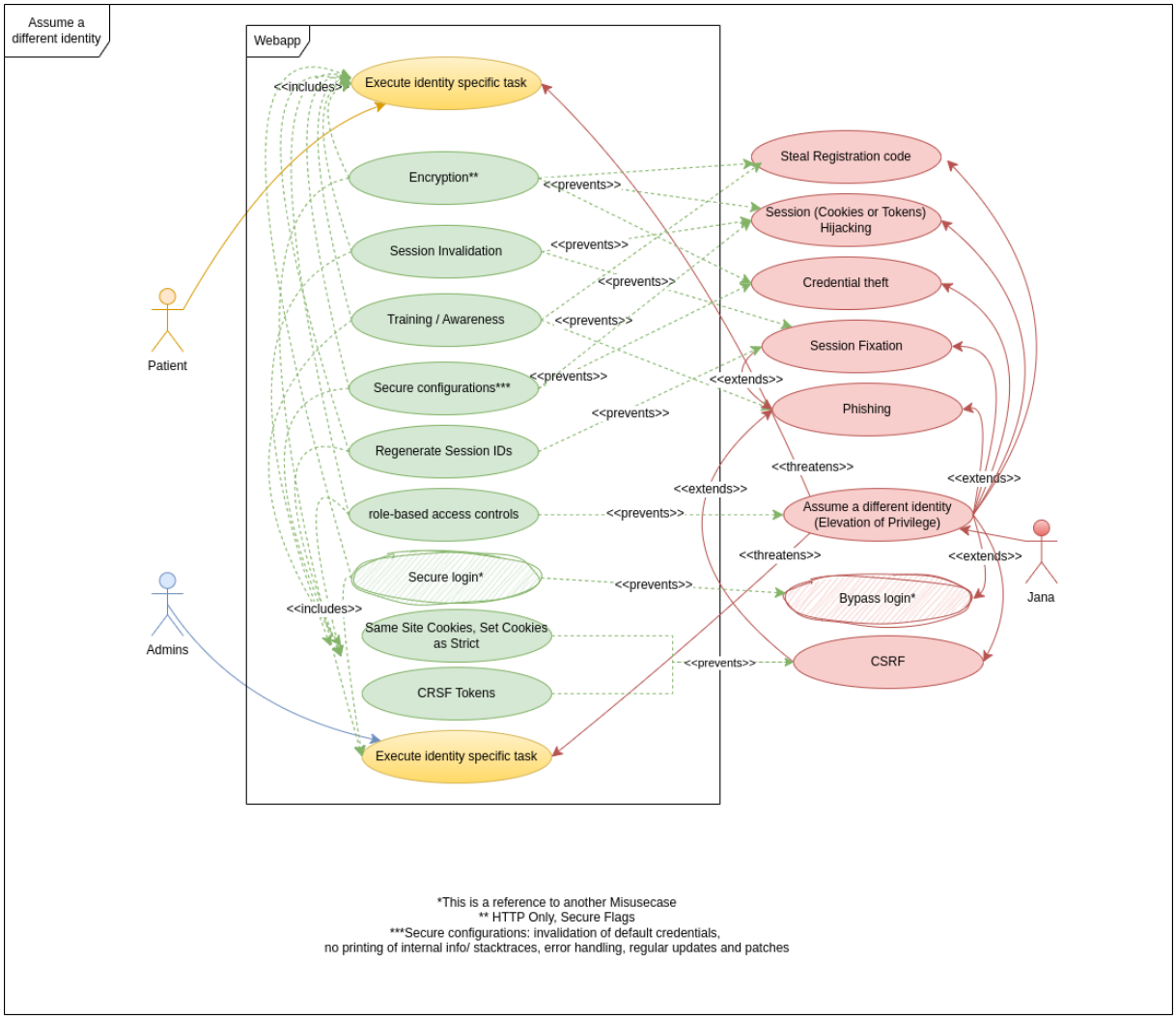


Figure 3: Misuse Case - Assume a different identity

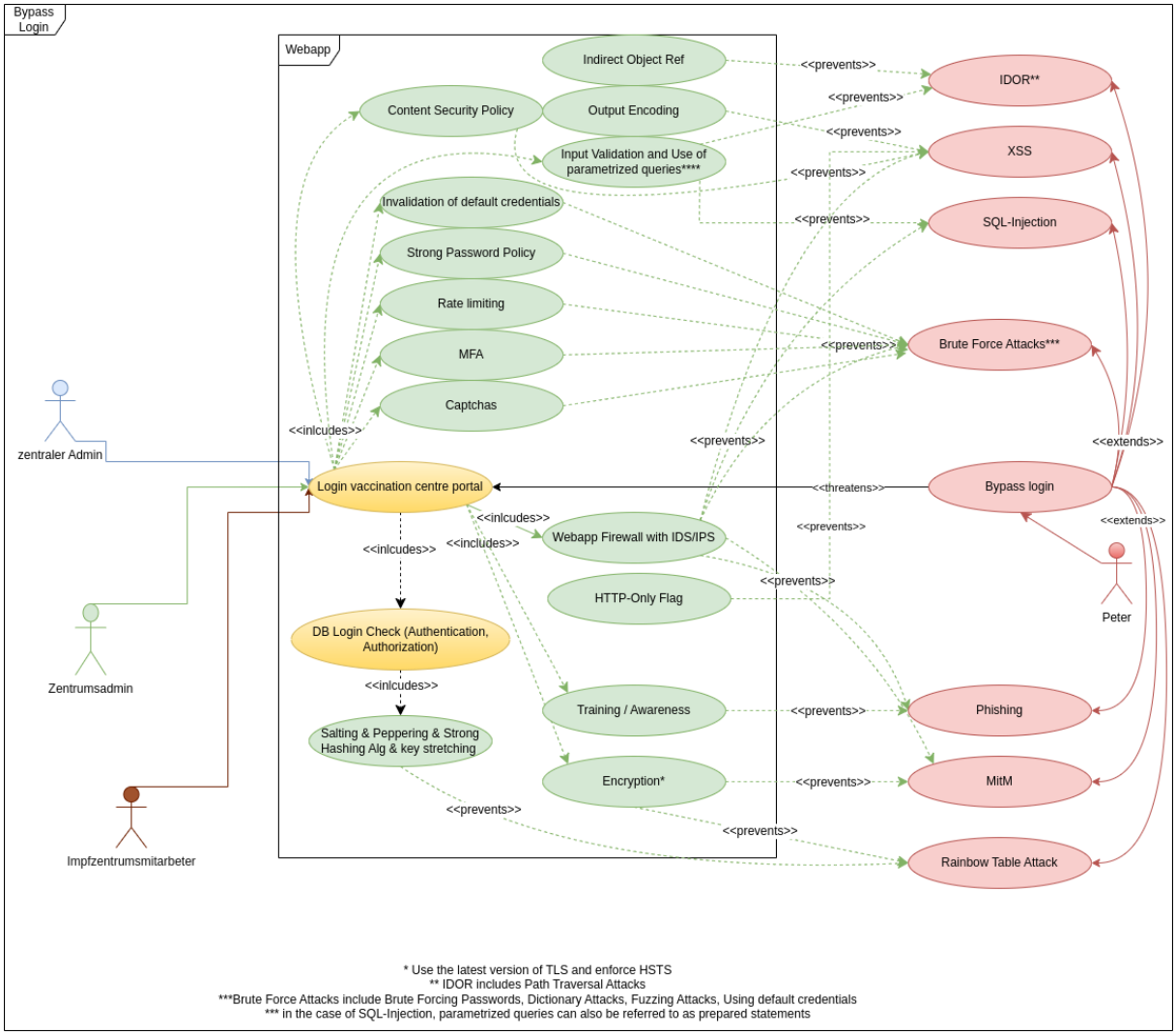


Figure 4: Misuse Case - Bypass Login

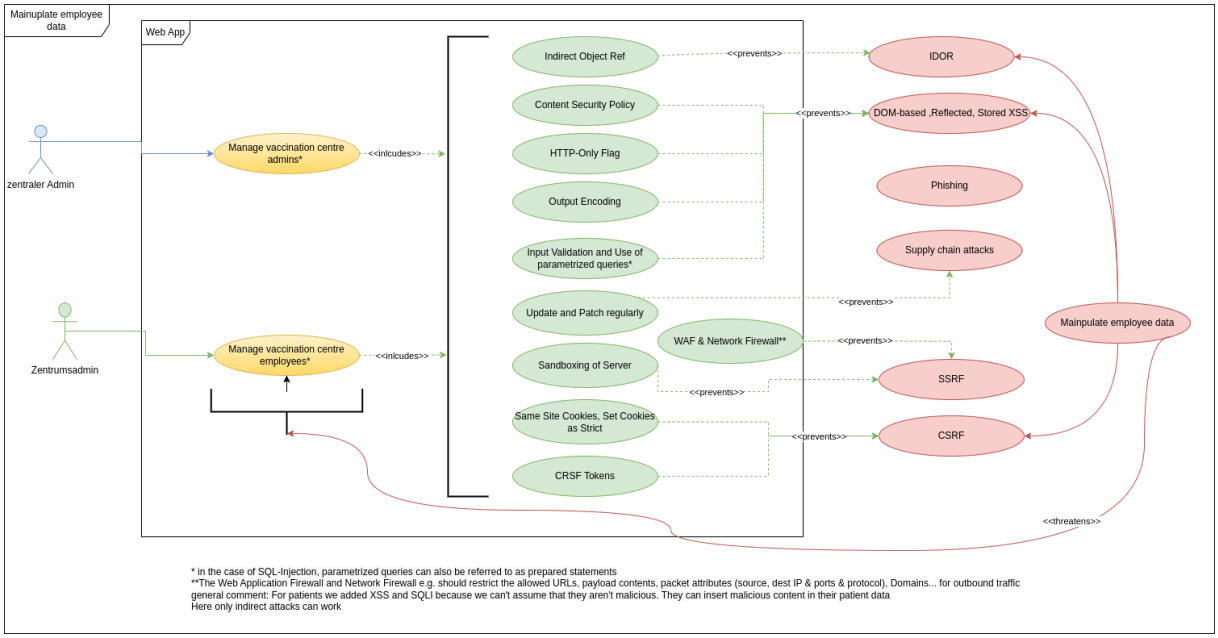


Figure 5: Misuse Case - Manipulate Employee Data

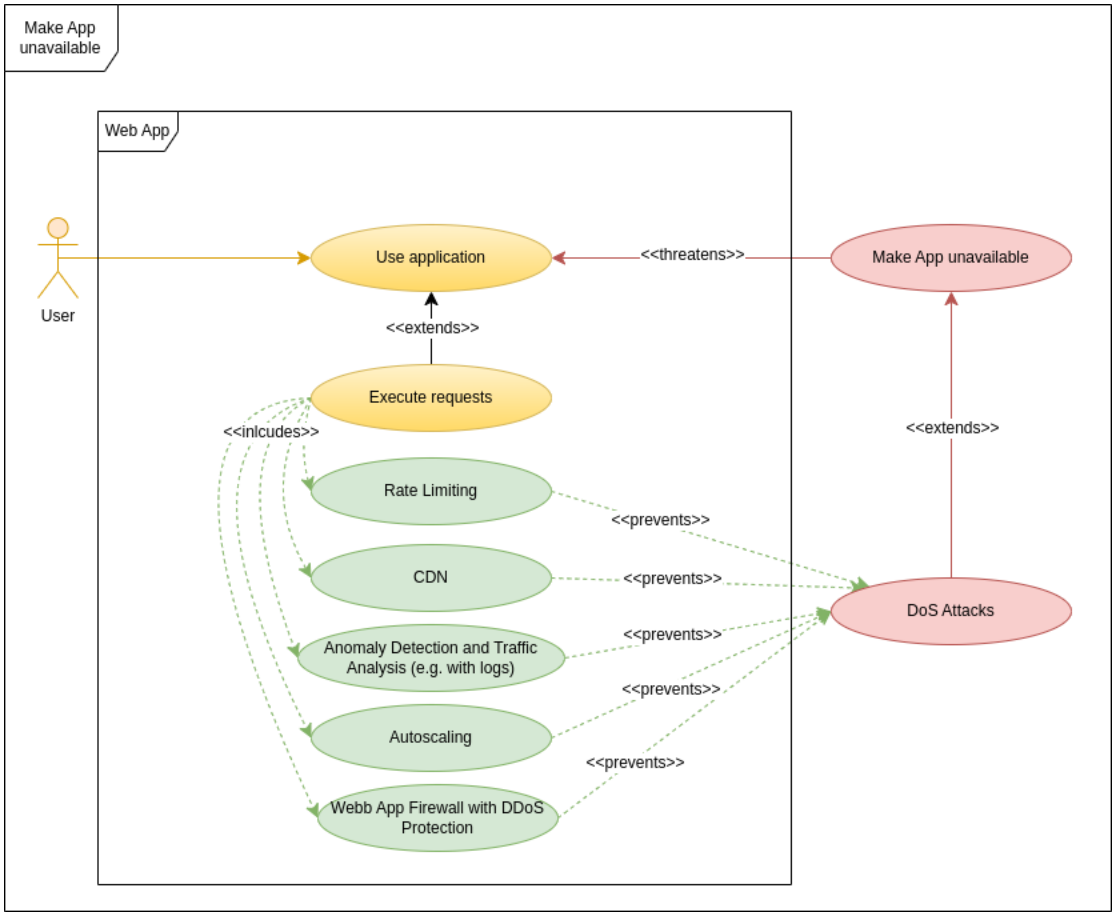


Figure 6: Misuse Case - Make App unavailable

Name	**Bypass Login**
Short Description	Attacker manages to bypass or break the login mechanism and obtain access to application functionalities and resources
Change History	5.5.2024
Actors	Attacker, (Personnel)
Context and Prerequisites	Login inputs are not validated/ whitelisted, or only client-side validated. Data in motion encryption is not enforced. Login attempts are not rate limited. Lack of security awareness in personnel. Vaccination portal login page is not IP/domain restricted. Data at rest is not encrypted.
Normal Procedure - **Brute Force**	1. Attacker has access to the vaccination center portal login page. 2. Attacker performs brute force attacks, including dictionary attacks, fuzzing attacks, and default credential attempts. 3. Attacker obtains authorized access to the Web Application.
Normal Result	The Attacker can perform identity specific tasks, corresponding to the credentials.
Alternative Procedure - **SQLi**	1. Attacker has access to the vaccination center portal login page. 2. Attacker performs SQL injection attacks on the login input form. 3. Attacker receives all usernames and passwords from the database. 4. Attacker can log in using obtained credentials.
Alternative Procedure - **XSS**	1. Attacker has access to the vaccination center portal login page. 2. Attacker sends script-injected link to victim personnel (scam-mail) 3. Victim clicks on the link, requests legitimate login portal and logs in. 4. Victim's browser executes malicious script and sends an HTTP request to attacker's webserver with stolen authentication cookies. 5. Attacker uses cookies to impersonate the user on vaccination center Portal.
Alternative Procedure - **IDOR**	1. Attacker has access to the vaccination center portal login page. 2. Attacker requests information using direct object references.

	3. The valid http request is executed, and the direct object reference entity is revealed. 4. Attacker obtains access to vaccination center portal functionalities.
Alternative Procedure - **Phishing**	1. Attacker sends an email containing a link to the victim personnel. 2. Victim clicks on the email and goes to the phishing login website. 3. Victim enters its credentials on the website. 4. Attacker collects victim's credentials. 5. Attacker logs in to the legitimate vaccination center portal website. 6. Attacker obtains access to the vaccination center portal and with the victim's authorization level.
Alternative Procedure - **MitM**	7. Victim is logged in to the vaccination center portal and is performing tasks on the home network. 8. Attacker accesses the Wi-Fi network by using stock credentials. 9. Attacker intercepts traffic between victim and portal. 10. Attacker captures victim's session cookies. 11. Attacker uses cookies to impersonate victim in the vaccination center portal.
Alternative Procedure - **Rainbow Table Attack**	1. Password Hashes of the vaccination center portal are leaked. 2. Attacker uses precomputed hash table to find a hash-string pair matching a password hash. 3. Attack uses password to log in to the vaccination center portal.
Countermeasures	1. Indirect Object References 2. Output Encoding 3. Content Security Policy CSP 4. Input Validation 5. Parameterized Queries 6. Invalidate Default Credentials 7. Strong Password Policy 8. Rate Limiting 9. Multi Factor Authentication MFA 10. Captchas 11. Web App Firewall WFA 12. HTTP-Only Flag 13. Training / Awareness 14. Encryption 15. Salting & Peppering 16. Strong Hashing Algorithms 17. Key Stretching

Reaction Options (Attacker)	- Distributed Brute Force Attacks (Botnets) - Zero-Day Exploits in underlying dependencies. - Advanced Phishing (Spear Phishing) - Watering Hole Attacks - Supply Chain Attacks
Stakeholders and Risks	Patients, all Personnel, complete takeover of Application, Data manipulation, Data breaches, Targeting of stakeholders, Complete service denial, vaccination center shutdown.

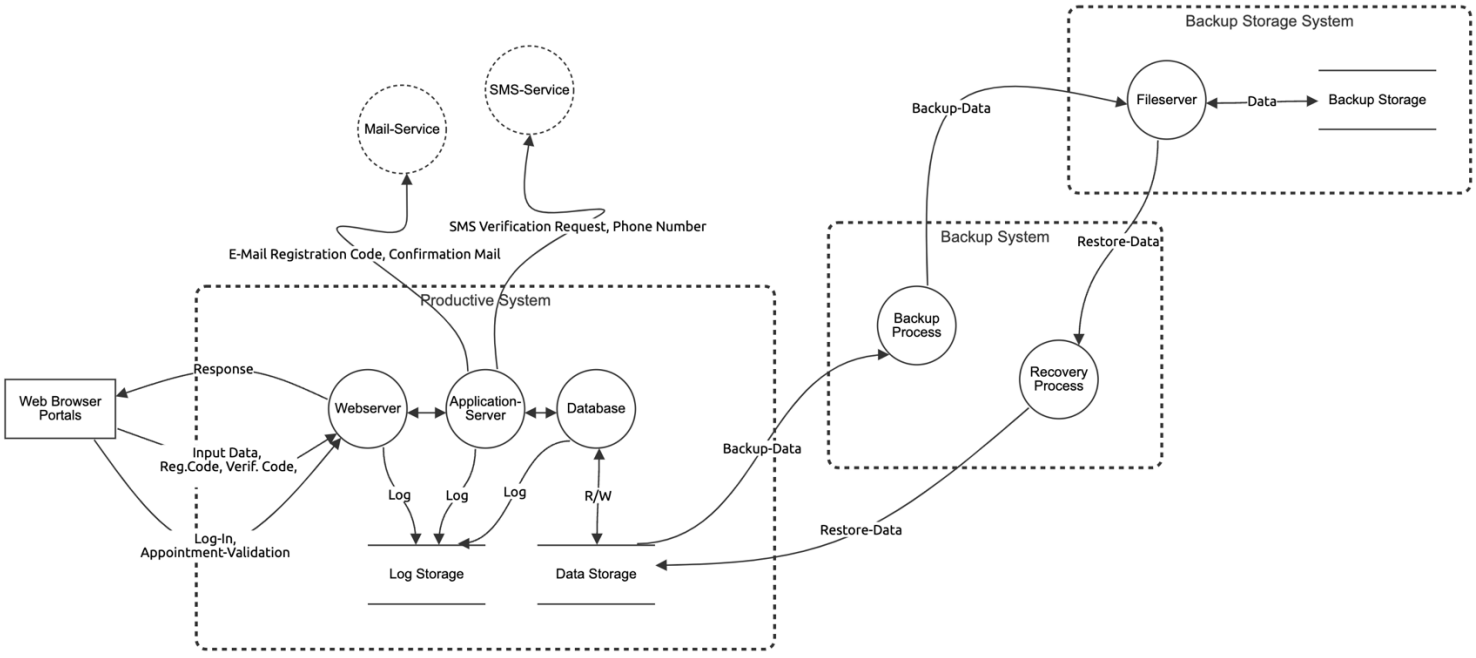
Name	**Assume a different identity**
Short Description	Attacker manages to assume a different identity than his own. This can also be categorized as identity theft. This allows the attacker to execute identity specific tasks as a patient or as an admin / employee.
Change History	5.5.2024
Actors	Attacker, Patient, Employees
Context and Prerequisites	1. Encryption of data in transit is not enforced (TLS, HTTP Only with Secure Flag) 2. Sessions are not being invalidated 3. Employees and Admins don't have enough security awareness against e.g. phishing attacks. 4. Insecure Configurations Reusage of Session IDs 5. No role-based access control 6. Insecure Login 7. No same-site cookies
Normal Procedure - **Steal Registration Code** - **Session Fixation**	**1. Execute patient specific tasks:** 1.1. Attacker steals a patient's registration code 1.2. Attacker deletes the patient's booked appointment to free up a spot for himself **2. Execute admin specific tasks:** 2.1 Attacker performs a Session Fixation Attack (e.g. admin authenticates with the attacker pre generated Session ID through a malicious URL) 2.2 Attacker is authenticated as the central admin 2.3. Attacker now removes all access of the vaccination center admins and can view all personal information of patients and employees
Normal Result	The Attacker can perform identity specific tasks, corresponding to the stolen or compromised identities.

Alternative Procedure - **Phishing**	**2. Execute admin specific tasks:** 2.1 Attacker prepares a phishing link, sending the admin to an exact replica of the vaccination portal login page at his server 2.2 The unaware admin logs into the vulnerable site 2.3 The attacker now has the admins credentials.
Alternative Procedure - **Session Hijacking**	**2. Execute admin specific tasks:** 2.1 Attacker injects script in server (e.g. with stored XSS) 2.2 Admin authenticates on server 2.3 Server returns page code with injected script 2.4. Admins Browser executes the scripts and sends session cookie to attacker 2.5 Attacker hijacks the user's session
Alternative Procedure - **Credential Theft**	**2. Execute admin specific tasks:** 2.1 Admin leaves password on a post it on the computer. 2.2 Attacker sees the credentials and steals them. 2.3 Attacker now has the admin's credentials.
Alternative Procedure - **Bypass Login**	**See Table 1**
Alternative Procedure - **Cross Site Request Forgery**	**2. Execute admin specific tasks:** 2.1 Attacker places malicious HTML onto a website that they control 2.2 The attacker induces admins to visit that site (e.g. complaint, phishing, social media message..) 2.3 The admin vists the malicious site while logged in on the vaccination portal. 2.4. The attacker's page will trigger an HTTP request to the vaccination portal website by the admin to change their email address to the email address of the attacker. 2.5. The admin's session cookie will automatically be included in the request (assuming SameSite cookies are not being used) and the vaccination portal will change the admin's email address to the attacker's email address. 2.6 The attacker can now trigger a password reset and receive the reset email on their email address (as it is registered as the admin's)
Countermeasures	1. Encryption of data in transit (use TLS, HTTP Only Flag with Secure Flag) 2. Use Same-Site Cookies and set cookies as strict

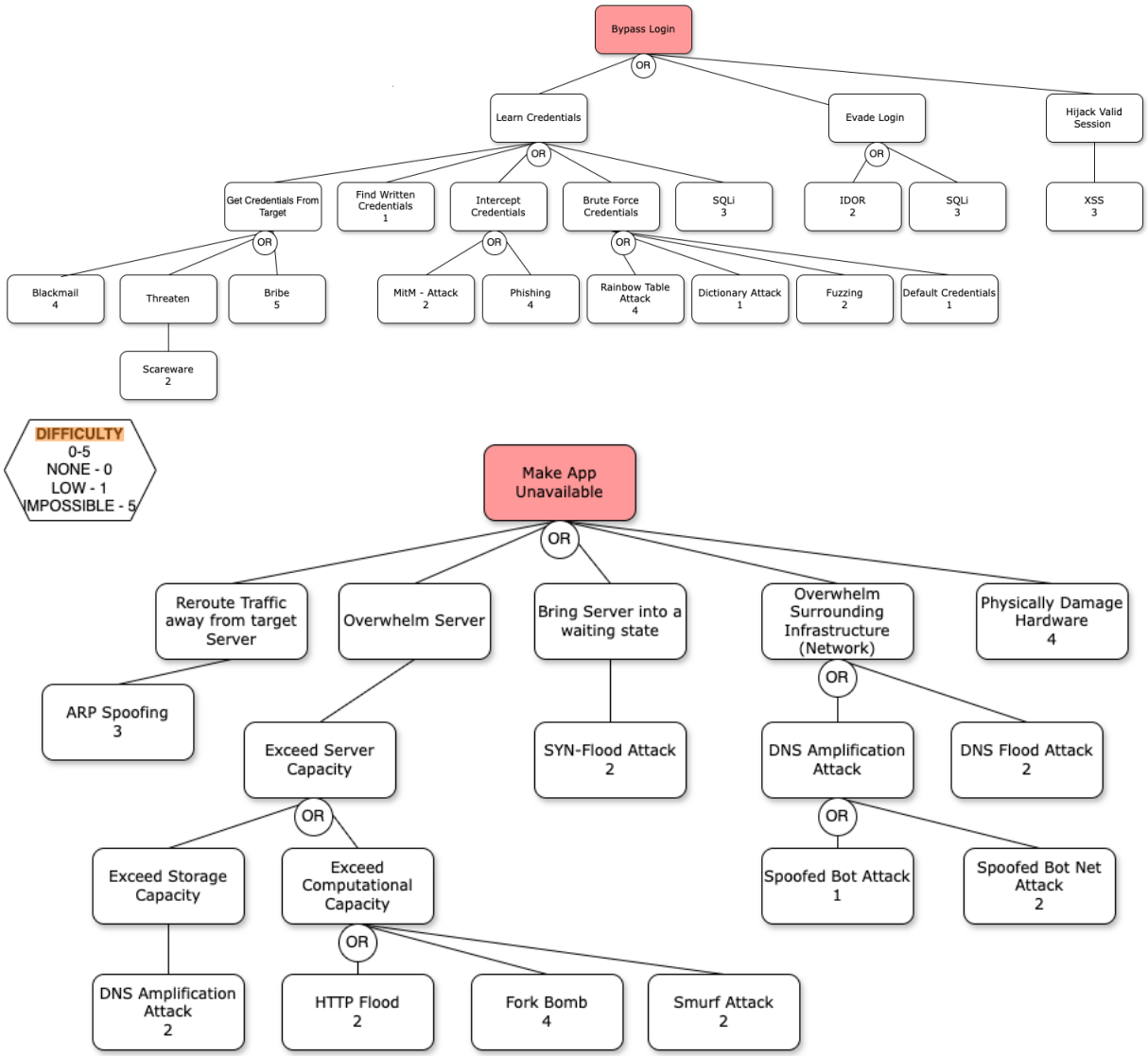
	3. Invalidate Sessions after Logout / after a specific period e.g. 1 hour 4. Regenerate Session IDs for every attempt to access the portal. 5. Employ mandatory training sessions and spread awareness on common attacks through Social Engineering / Phishing and their consequences 6. Use secure configurations and keep error handling and error messages internal 7. Perform Regular Updates of System and Software Components 8. Perform Patches for known vulnerabilities 9. No role-based access control 10. Secure the login 11. Ensure Role Based Access Control 12. Use CRSF Tokens 13. No same-site cookies
Reaction Options (Attacker)	- Social Engineering - Be an undercover vaccination center employee and attack from the inside (non-requirement)
Stakeholders and Risks	**Patient identities** in form of registration codes could be stolen. This would result in the attacker being able to manage their appointments and cancel their needed vaccination appointment bookings (e.g. to free up appointments). This would again result in the patient not being able to get a vaccination, which could result in death of COVID. On the other side this will also make patients lose their trust in the vaccination portal application. --> Attack on Integrity of patient data, Confidentiality of sensitive patient information, Availability of the function of booking, Non-repudiation that a patient was the one cancelling the appointment. Assuming the **central admin identity** can result in the attacker being able to manage and control all other admin accounts and make themselves and admin or employee account. This would result in the compromission of all sensitive data of the patients and employees. The attacker can also shut down the application or make all data public. This renders the whole application then useless, compromised and would ruin the reputation of the business.

	Assuming the **center admin identity** will result in the attacker manipulating employee data and being able to create employee accounts for themselves. Assuming the **employee identity** would result in the attacker being able to change bookings, mess up all plans for vaccination appointments. This not only impacts patients but also the employees working at the centers being thrown into chaos and losing valuable working time. --> Attack on.. ..Integrity of patient data and employee data, ..Confidentiality of sensitive patient and employee information, ..Availability of the whole application, ..Non-repudiation, ..Authorization mechanisms

3.3 Data flow diagram



3.4 Attack-Trees



3.5 Threats and Mitigations

Threat ID	Threat	Description	Stride (Threat Type)	Mitigation	DFD Element	Risk Rating	OWASP Top 10	Handling
T001	Non-Canonical Names	An attacker can bypass access permission checks because of the use of non-canonical (relative path) for log access checks. This leads to Information Disclosure, Confidentiality Violation negatively impacting logs.	T- Tampering with Data I - Information Disclosure E – Elevation of Privilege	Use canonical Names and Paths	Log Storage	Likelihood: Medium Technical Impact: Medium Business Impact: Medium Overall Risk Severity: Medium	**A01: 2021- Broken Access Control**	Address
T002	No encryption of data in transit	An attacker can read and manipulate because there's no integrity protection for data on the network / data in transit. This leads to the corruption of integrity, confidentiality and negatively impacts all data flows over an unsecured network (the internet).	R- Repudiation T I	Encrypt with TLS	ALL	LH: Medium TI: HIGH BI: HIGH Overall Risk Severity: HIGH	**A02: 2021- Cryptographic Failures**	Address
T003	Inadequate Access Control	An attacker can write to some resource because permissions are granted to the world or there are no ACLs. This leads to the corruption of integrity and negatively affects the data flow between Web Browser and Webserver (malicious input data from the Browser and the (corrupted) responses that Web Browsers receive).	T E	Principle of Least Privilege, MFA, Secure Configuration Management	Web Server	LH: MEDIUM TI: MEDIUM BI: MEDIUM Overall Risk Severity: MEDIUM	**A01: 2021- Broken Access Control** **A04: 2021- Insecure Design** **A05: 2021- Security Misconfiguration**	Address
T004	Missing Rate Limits	An attacker can make the authentication system unusable or unavailable through overloading the application with authentication requests. This compromises the availability of the application and negatively impacts the Login Data Flow of the application.	D – Denial of Service E	Rate Limiting, Captcha, Fail2ban, IP Blocking, Load Balancing, Scaling, Monitor & Alert, DDoS Protection Services	Webserver	LH: HIGH TI: HIGH BI: HIGH Overall Risk Severity: HIGH	**A07:2021 - Identification and Authentication Failures** **A05:2021- Security Misconfiguration**	Address
T005	Missing Load Balancing	An attacker can make your application features unusable or unavailable through overloading the application with information retrieval or writing requests utilizing common DDoS attacks (e.g. Billion Laughs). This compromises the availability of the application and negatively impacts the Database, the Application Server,	D	Firewalls, IDPS, Load Balancers, Captcha, Rate Limit, Incident Response Plan	Webserver, Application Server, Database	LH: MEDIUM TI: HIGH BI: HIGH Overall Risk Severity: HIGH	**A04:2021- Insecure Design**	Transfer

		the Webserver, and the Response Data Flow.						
T006	Syn Flooding	An attacker can make a client unavailable or unusable and the problem persists after the attacker goes away by doing a SYN Flood attack and thereby misusing all the application's connections. This leads to the persistent unavailability of the application negatively impacting all data flows that require a connection (all data flows between Web Browser and Webserver).	D	Syn Cookies, Increase Connection Queue Size, Rate Limiting, IDS	Webserver	LH: HIGH TI: MEDIUM BI: LOW Overall Risk Severity: MEDIUM	**A05:2021- Security Misconfiguration**	Address
T007	Exploiting Data Flows for Unauthorized Access	An attacker can enter data that is checked while still under their control e.g. with a stored XSS attack through the Web Browser and then use it later to render or execute the malicious stored	T E	Input Validation, Whitelist, Output Encoding, Content Security Policy, WAF	Webserver, Application Server, Database, Backup Server	LH: HIGH TI: HIGH BI: HIGH Overall Risk Severity: HIGH	**A03: 2021 -Injection**	Address
		content on the other side of a trust boundary with the Database and Application server processes. This compromises integrity, non-repudiation and negatively impacts the security of the Application Server and Database and potentially its data flows.						
T008	Weak Password	An attacker can find out the plain text credentials with guessing or using password lists or default credentials using brute force attacks, dictionary attacks. This leads to the compromission of non-repudiation, confidentiality and integrity of data which negatively affects all role specific functions depending on the compromised credentials.	S	Force Password Changes, Strong Password Policy, Password Guidelines and Best Practices, MFA	Webserver, Application Server, Database	LH: MEDIUM TI: HIGH BI: HIGH Overall Risk Severity: HIGH	**A07:2021- Identification and Authentication Failures**	Address

T009	Weak Hashing Functions	An attacker can find out the plain text credentials with rainbow table attacks on hashed data. The attacker can crack the hashes faster if multiple users have the same password (e.g. hash). This leads to the compromission of non-repudiation, confidentiality and integrity of data which negatively affects all role specific functions depending on the compromised credentials.	S T	Strong Hashing Algorithms, Strong Password Policies, Rotating Hashing Algorithms, Rate Limit Login Attempts, Monitoring, MFA	Webserver, Application Server, Database	LH: HIGH TI: HIGH BI: HIGH Overall Risk Severity: HIGH	**A02:2021-Cryptographic Failures**	Address
T010	Insufficient Credential Hashing (Salt & Pepper)	An attacker can find out the plain text credentials from hashed compromised credentials (hashed passwords and user specific salt) utilizing brute force, rainbow table attacks or hash cracking sites or applications. This leads to the compromission of non-repudiation, confidentiality and integrity and negatively affects all role specific functions depending on the compromised credentials.	S T	Strong Hashing Algorithms (SOTA), Salting, Peppering, Enforce Strong Password Policies, Log and Monitor, MFA, Response Plans in case of compromise, Security Audits	Webserver, Application Server, Database	LH: HIGH TI: HIGH BI: HIGH Overall Risk Severity: HIGH	**A02:2021-Cryptographic Failures**	Address
T011	Unencrypted Backup Data (Data at Rest)	This assumes an attacker has compromised the Backup Storage Data (e.g. with EoP on the Backup Storage System File Server). The attacker can read all backup data. This leads to the break of confidentiality and negatively impacts the Backup Storage System.	I T	Strong Encryption of Data at Rest, Access Controls, Monitoring and Auditing, Redundant Backups (in case of Tampering), Principle of Least Privilege ACL, Vendor Security Assurance (External Entity)	Log Storage	LH: HIGH TI: HIGH BI: HIGH Overall Risk Severity: CRITICAL	**A05:2021-Security Misconfiguration**	Address
T012	Pointer Manipulation and Buffer Overflows	The attacker can access an unintended memory location through an application vulnerability or buffer overflow attacks (e.g. OpenSSL Heartbleed or Insecure usage of parameters). This can result in the crashing of the application or, for certain pointer values, access to data that would not normally be possible	T I E D	Input Validation, Boundary Checks, Memory Management, Stack Canaries, Address Space Layout Randomization (ASLR), Data Execution Prevention, Sade Programming Languages (Rust,Swift), SCA, DCA, Awareness and Training	Webserver, Application server, Backupserver	LH: MEDIUM TI: MEDIUM BI: HIGH Overall Risk Severity: MEDIUM	**A03:2021-Injection**	Address

		or the execution of arbitrary code, thereby compromising confidentiality and availability. This negatively impacts the Data Storage by accessing unauthorized memory locations.						
T013	Injection	An attacker can write to the data storage that the application code relies on through Injection. Thereby the attacker compromises data integrity. This negatively affects the Data Storage and Backup Storage and their write data flow. (HTML parameter injection, command injection, XML external entity, XPath injection)	T E	Input Validation, Parameterized Queries, WAFs, IDS, IPS		LH: HIGH TI: MEDIUM BI: HIGH Overall Risk Severity: HIGH	**A03:2021-Injection**	
T014	DoS Amplification	An attacker can amplify a Denial-of-Service attack through the embedding of malware in clients to create a botnet with amplification on the order of 10 to 1, resulting in the compromission of	D	Rate Limiting, DDoS Protection Services, Attack Surface Reduction via Load Balancer, Blocking Communication to unused Ports, Anycast network diffusion, Caching	Webserver, Backup System	LH: HIGH TI: MEDIUM BI: LOW Overall Risk Severity: MEDIUM	**A05:2021-Security Misconfiguration**	Transfer
		availability. This negatively impacts the availability of the Webserver, Application Server, and Database.		and CDNs, Rate Limiting				
T015	Downgrade Attack	The attacker can manipulate the security of the TLS/SSL encryption by downgrading the available cipher suites on the victim's client side and choosing a weaker Cypher Suite in the TLS/SSL handshake being then able to decrypt the agreed upon cipher suites. This leads to the compromission of the confidentiality and integrity of data in transit negatively affecting all encrypted data flows between Web Browser and Webserver.	T S I	Forcing Strong Cypher Suites, Removing SSL Support, Use latest TLS version, Perfect Forward Secrecy, HSTS, Strong Key Exchange Algorithms (ECDH)/ RSA	Webserver	LH: HIGH TI: MEDIUM BI: HIGH Overall Risk Severity: HIGH	**A02:2021-Cryptographic Failures**	Address
T016	Insecure Password Recovery	An attacker can utilize the password recovery or password updating mechanisms to compromise a victim's credentials. (e.g. account recovery doesn't	E S	MFA, Confirmation Emails, SMS codes, Monitor Account Activity	Webserver, Application Server	LH: HIGH TI: MEDIUM BI: HIGH Overall Risk Severity: HIGH	**A04:2021-Insecure Design**	Address

		require authentication or doesn't require disclosing old password). This compromises authentication, authorization mechanisms and does not account for fault tolerance in a secure way. This negatively impacts the data flow of the application login.						
T017	Bad Error Handling	An attacker can access internal information (also about inner workings of the application) through purposely triggering errors because the errors or stack trace or other internal information is presented to the Web Browser in a Debugging manner. This leads to the compromission of confidentiality and negatively impacts the Application Server Code information or even Database / Data Storage specific internal information.	I E	Disable Debug Mode for Logs in Production, Never Present Internal Information to the UI in Production, Implement Robust Error Handling Mechanisms with generic error messages	Web portal, Webserver	LH: HIGH TI: HIGH BI: HIGH Overall Risk Severity: CRITICAL	**A05:2021-Security Misconfiguration**	Address

Threat ID	Action	Reason	First Order Mitigation (for first order attack stated in the threat table)	Second Order Mitigation	Third Order Mitigation
T001 (Non-canonical names)	Address	The threat must be eliminated as it has a great impact on elevation of privilege. Threat is easy to eliminate	Use a SCA Tool with a rule to throw an error in case a developer uses non-canonical names for auth checks. Training and Awareness for Developers.	2nd Order attack: Path Traversal Mitigation: Input Validation, surround input with internal path that user can't know, use hash maps to encode or decode path values, so the internals remain unknown to user.	
T002 (No encryption of data in transit)	Address	All DFD elements can be affected by this threat, and it is within our responsibility to secure them and the data flow between different trust boundaries over an unsecured network.	Use TLS or IPSEC to secure and encrypt data in transit. Use Digital Signatures for confidential or critical or sensitive information before sending to the receiver.	2nd Order attack: TLS Downgrade attack Mitigation: Use the latest version of TLS to ensure support is only for safe cipher suites.	3rd Order Attack: MitM Attack with 2 connections Mitigation: (proof of work) Authentication phase in TLS: Client validates cert (not expired or revoked), client has to trust CA, client
					validates that DNS name queried matches the one on the trusted cert, client validates the server has the private key with encrypting a message to the server with its stated public key
T003 (Inadequate Access Control)	Address	Securing write access to the data storage is our responsibility. (authorization)	Implement Access Controls and ACLs to the data stating explicit denies and explicit allows for authorized users to write data.	2nd Order attack: an authorized user writes malicious data. Mitigation [non-requirement]: Use Digital Signatures to ensure non-repudiation and be able to find the responsible person	3rd Order Attack: an authorized user changes security mechanisms within the application Mitigation: non requirement (malicious admin or employee)
T004 (Missing Rate Limits)	Address	Rate Limiting has to be implemented in the Application Server (business logic) of the application.	Rate Limiting, Captcha, Fail2ban, IP Blocking,	2nd Order attack: DDoS: Botnet Attacks Mitigation (Transferred to hosting partner): Load Balancing, Scaling, CDNs, Monitor & Alert, DDoS Protection Services e.g. in Threat Intelligence Systems	3rd Order Attack: Physical Damage of the hardware Mitigation: Transfer the threat to the hosting partner to ensure physical security of assets.
T005 (Missing Load Balancing)	Transfer	Hosting Provider should ensure the availability of the application with enough and secured resources, as "we do not have the	-	-	-
		necessary experience to operate the application"			
T006 (SYN Flooding)	Address	TCP connections are the base for reliable data transfer within our application. It is important to ensure these connections can be created and deleted when (not) needed.	Syn Cookies, Increase Connection Queue Size, Rate Limiting, IDS		
T007 (Exploiting Data Flows for Unauthorized Access)	Address	We have to secure all data flows within our internal system communication	Input Validation, Whitelist, Output Encoding, Content Security Policy, WAF	2nd Order attack: Cross-Site Request Forgery (CSRF) Mitigation: CSRF Tokens Same Site Cookies Set Cookies as Strict	3rd Order Attack: (also, in combination with any of the previous threats and attacks) Mitigation: (Stated as a non-requirement), Training and Awareness of Admins and Warning of App users.
T008 (Weak Password)	Address	As we have a Login, we have to secure the usage of passwords	1st Order attack: Dictionary Attack / authenticate as an admin with Default Credentials Force Password Changes, Strong Password Policy to ensure use of secure passwords, Password Guidelines and Best Practices,	2nd Order attack: Brute Force Attacks: Mitigation: Multi- Factor Authentication (MFA), Limit on the number of attempts, Log & trigger alarm notification	

T009 (Weak Hashing Functions)	Address	Responsibility of securing user credentials in the database (data at rest), even if they get compromised or leaked.	Strong Hashing Algorithms, Rotating Hashing Algorithms	2nd Order attack: Rainbow Table attacks: Mitigation: Strong Password Policies, Salting	3rd Order Attack: Compromised hashed user credentials & salt Mitigation: Peppering
T010 (Insufficient Credential Hashing (Salt & Pepper))	Address	Responsibility of securing user credentials in the database (data at rest).	Strong Hashing Algorithms (SOTA), Salting, Peppering, Enforce Strong Password Policies, Log and Monitor, MFA	2nd Order attack: (highly unlikely) All credentials, salt, pepper get compromised Mitigation: Response Plans in case of compromise, Regular Security Audits	
T011 (Unencrypted Backup Data (Data at Rest))	Address	Backup File Server is under the responsibility and is essential for ensuring high availability and for recovery options like Backup & Recovery. It allows us to meet RPO objectives.	Strong Encryption of Data at Rest, Immutable Backups, Vendor Security Assurance (from external hosting provider)	2nd order attack: Data Storage has been tampered with before backup Mitigation: Access Controls and Role Based access with authentication and authorization to data storage, Principle of Least Privilege, Monitoring and Auditing,	3rd Order attack: Backup gets corrupted / tampered with Mitigation: Redundant Backups (to ensure successful backups for RPO)
T012 (Pointer Manipulation and Buffer Overflows)	Address	Secure design and security bets practices are part of mainlining a secure application and the basis of securing sensitive data.	Programming: Input Validation, Boundary Checks, Memory Management, Stack Canaries, Address Space Layout Randomization	2nd order attack: Attack on vulnerable 3rd party libraries Mitigation: Use the latest version, regularly update all dependencies and	
			(ASLR), Data Execution Prevention, Sade Programming Languages (Rust,Swift), SCA, DCA, Awareness and Training	check for vulnerabilities, patch any vulnerabilities	
T013 (Injection)	Address	Data Integrity, Information Protection and Ensuring Availability is crucial for the application (CIA), Regulatory Compliance	1st order attack: HTML Parameter Injection, Command Injection, XXE, XPath Mitigation: Input Validation, Output Encoding, CSP, Input Sanitization, Disable External Entity Processing, Parameterized XPath queries	2nd order attack: Attacker utilizes injection through insecure or vulnerable libraries (3rd parties) Mitigation: Use Secure Libraries	
T014 (DoS Amplification)	Transfer	This attack pertains to the network infrastructure and can be transferred to the network administrator parties.	Rate Limiting, DDoS Protection Services, Attack Surface Reduction via Load Balancer, Blocking Communication to unused Ports, Anycast network diffusion, Caching and CDNs, Rate Limiting, IP Blocking	2nd order attack: Attacker uses Botnets to launch Distributed Reflective Denial of Service (DRDoS) attacks, where source IP are spoofed and reflectors such as DNS Services , NTP Protocols, Memcached are used to amplify over 50000 fold. Mitigation: source IP Filtering, Configure Firewalls and IPS to block known attack signatures	

T015 (Downgrade Attack)	Address	Exposure of sensitive information eavesdropping and modification has to be addressed with adequate methods in order to ensure a standard level of security,	Use Strong Cypher Suites, enforce newest TLS Versions, Implement HSTS		
T016 (Insecure Password Recovery)	Address	Direct unauthorized access to user account can be achieved easily allowing for further malicious actions.	Implement MFA, SMS, Email Verification and use Captchas		
T017 (Bad Error Handling)	Address	This needs to be addressed to prevent information leakage, protect confidentiality, and comply with regulations for sensitive data in first line.	Custom Error Handling to control information presented to users, Disable Debugging Mode, Configure Web Server and Application to minimize exposure of detailed stack traces in production		

3.6 Considered Security Design Principles

Secure The Weakest Link: This principle is considered with awareness training for personnel regarding social engineering attacks and phishing, as well as in coding principles regarding buffer overflow vulnerabilities.

Practice Defense in Depth: This principle is considered with secure implementation and using secure design at all stages of the system. Data in motion is encrypted, WAFs protect the system as a first layer defense, alongside content delivery networks and load balancers. Strong input validation, complex hashing of credentials and strong encryption of data at rest complement this at every layer. Enforcing of strong password policies and MFA further add to defense in depth.

Fail Securely: This principle is considered with regular and continuous Backups of all Data on an external system and implemented restoration mechanisms. Comprehensive incident response plans secure and guarantee procedures after a catastrophic event.

Principle of Least Privilege: This principle is considered in the authentication and authorization through strict access control and ACL granting only necessary access to a user to the web application. This also goes for database access and application server access. The processes pertaining to the application are also granted only fine-grained access to the system and other processes.

Compartmentalize: This principle is considered in the separation of processes such as backup, application, database and mail/sms system. As such breaking one process does not compromise the other processes.

Keep it simple: This principle is considered in the implementation of the application. The singular interface to access the application via a HTTPS API from the WAN, as well as using strong encryption libraries and the overall simple design, makes the system secure through simplicity, as no hidden backdoor or plentiful bugs are to be expected.

Promote Privacy: This principle is considered in the usage of firewalls to block unnecessary services so they cannot be used for attacker reconnaissance. It is also considered by using tried and trusted encryption and hashing for data at rest and in motion to protect sensitive patient data and by using fine grained privilege for access to patient data while also using strong authorization/ authentication.

Be Reluctant to Trust: This principle is considered in the usage of authentication mechanisms. Users are not trusted to be benevolent. MFA and access minimization help add an extra layer of security and grant the minimum level of access to perform a task. Every critical request from the Browser to the Web Server is authenticated, encrypted, and authorized. Secure protocols (TLS) are used. External system suppliers are vetted for compliance with security guidelines.

Use Your Community Resources: This principle is considered in the usage of tried and tested security libraries which have been patched for security issues. This also applies to the usage of other system components such as Web Server, Database, and Application Server.

3.7 Further Safety Measures to protect the Application

To mitigate the risk of attackers injecting commands that persist in the system and affect future operations such as backup routines or data alterations:

- Implement immutable backups.
- Establish comprehensive logging and monitoring.
- Conduct regular security audits.

To counteract malicious activities carried out while services are unavailable, such as data exfiltration or ransom demands:

- Ensure all sensitive data is encrypted.
- Implement network segmentation.
- Develop a comprehensive incident response plan.

To prevent targeted exploitation of other systems, compromising overall security:

- Implement continuous monitoring.
- Develop and regularly update incident response plans.

To prevent broader attacks using obtained accounts, such as phishing campaigns or targeted attacks on users and systems:

- Continuously monitor and analyze account activities.
- Apply the principle of least privilege to user accounts.
- Use adaptive authentication methods.
- Implement a zero-trust architecture with micro-segmentation.

To prevent the misuse of obtained database credentials to access databases, systems, and networks:

- Implement strong authentication mechanisms.
- Apply the principle of least privilege.
- Segment networks to prevent lateral movement.
- Continuously monitor access and activities within the system.

4. Recommendations for the Implementation

1. Federated identity management

To ensure the login is secured it is recommended to use a third-party identity provider (e.g. OAuth or Google) to authenticate admins and employees. This will hold the login to the highest standard and will alleviate the expenses and time it would take to implement our own solution (that would most probably not be as secure). This will also ensure strong password policies; enablement of MFA and secure account recovery elevate methodologies.
Added to that a huge plus is that federated identity management can allow us to store credentials with a trusted third party that implements all the stated best practices when it comes to password hashing, salting, peppering and storing.

2. Session Management

Session management as a secure design pattern ensures robust user authentication and data protection. Key best practices include using secure, random, and unique session IDs, and employing HTTPS to safeguard data in transit. When using JSON Web Tokens (JWT), encrypt sensitive data and validate tokens server-side to prevent tampering. Implement short-lived JWTs with refresh tokens to balance security and usability. Securely store JWTs in HttpOnly cookies to mitigate XSS risks and set the Secure and SameSite attributes. Regenerate session tokens after authentication events to prevent fixation attacks, and implement session timeout and invalidation policies to mitigate risks from inactive sessions. Regularly audit and monitor session activities to detect anomalies. Educate users on secure session practices, such as logging out from public devices, to further enhance security. These practices collectively ensure a secure, efficient, and user-friendly session management system.

3. Role Based Access Controls

Role-Based Access Control (RBAC) assigns permissions to users based on their roles within an organization, ensuring that users only have access to resources necessary for their job functions. It should be utilized to grant the general admin, the center admin and the employee's role-based access to data and functions specific to their job functions as listed in the user requirements diagram. Central to RBAC is the principle of least privilege, which states that users should be granted the minimum level of access—or permissions—necessary to perform their tasks, minimizing potential damage from accidental or malicious actions. This principle is vital for managing access control because it reduces the risk of unauthorized access to

sensitive information and should be followed and regularly maintained in the security lifecycle and with employee changes.
With Access Control Lists (ACLs) administrators can also define exactly what resources users or roles can access and what operations they can perform on those resources or explicitly deny access.
Effective RBAC implementation involves detailed authorization processes to verify that users have the correct permissions for their roles. Best practices for RBAC include regular review and updating of roles and permissions, leveraging tools like Microsoft Azure Active Directory, AWS IAM, or OpenLDAP for managing roles and permissions, and ensuring compliance with security policies. Properly implemented RBAC not only enhances security but also simplifies management and auditing of user permissions and minimizes the attack surface to role specific functions.

4. **Single Access Point**

The Single Access Point (SAP) design pattern centralizes all user authentication and authorization processes through a single-entry point, typically an authentication server. This approach simplifies security audits and monitoring of access by providing a single location for logging and controlling user sessions. Best practices include implementing strong authentication mechanisms, such as multi-factor authentication (MFA), and using secure communication protocols like TLS. Tools like OAuth, OpenID Connect, and SAML facilitate SAP implementation, ensuring robust and standardized authentication. Centralizing access control helps maintain consistent security policies, enhances threat detection as data is concentrated in one place, and facilitates finding problems or sources when in the process of incident responses as it will also have centralized logs.

5. **Encryption of Data in Transit**

Encryption of data in transit is a fundamental secure design pattern that ensures the confidentiality, integrity, and authenticity (CIA) of data transmitted over unsecure networks, such as the internet. The primary method for securing data in transit between the web browser running the vaccination application and the web server is through Transport Layer Security (TLS). It is always recommended to use the latest version of TLS to prevent Downgrade attacks. The latest version, TLS 1.3, offers enhanced security and performance features, including the elimination of outdated cryptographic algorithms and a simplified handshake process. Best practices include enforcing the use of TLS 1.3, configuring strong cipher suites, and using certificates from trusted Certificate Authorities (CAs) for authentication.
Additionally, IPsec (Internet Protocol Security) can be used to secure data at the IP layer, providing end-to-end encryption and integrity for network traffic.
Implementing Perfect Forward Secrecy (PFS) ensures that session keys are not compromised even if long-term keys are compromised, adding an extra layer of

security when talking about defense in depth. This ensures that session keys are rotated automatically to ensure that previously encrypted data cannot be decrypted at a later point in time if it is compromised.
Tools such as TLS and Let's Encrypt can facilitate the deployment and management of TLS certificates. Regular updates and patches, thorough testing, and adherence to the latest security guidelines are crucial to maintaining robust encryption and protecting data in transit against potential eavesdropping and tampering attacks.

6. Encryption of Data at Rest

Utilizing strong encryption standards, such as AES-256, ensures robust protection against sophisticated attacks. Best practices for implementing encryption of data at rest include employing full-disk encryption for entire storage devices and database encryption for sensitive patient data within databases. It is crucial to manage encryption keys securely using hardware security modules (HSMs) or key management services (KMS) to prevent key exposure. This can be used by a third-party provider instead of having a higher risk of implementing it ourselves (e.g. AWS for Cloud based KMS and HSMs).
Regular backups are an essential part to recovery plans. It is recommended to ensure backups are encrypted, so that they cannot be tampered with. They have to also be protected by access controls and accessible for disaster recovery without compromising security. Tools like BitLocker, VeraCrypt, and AWS KMS provide effective solutions for managing encryption at rest. Implementing these measures not only safeguards sensitive information from theft and unauthorized access but also helps comply with regulatory requirements such as GDPR and HIPAA, thereby maintaining the integrity and confidentiality of critical data while also ensuring ease of access for disaster recovery.

7. Regular Security audits

Security is not something that is achieved once. It has to regularly be maintained and improved iteratively. Security is a process. Regular security audits, a key secure design pattern, involve systematically reviewing and assessing an application’s security posture to identify vulnerabilities and ensure compliance with security policies. Best practices include conducting audits at regular intervals, after major updates, and post-security incidents. Utilize automated tools like OWASP ZAP, Burp Suite, and Nessus for vulnerability scanning, and pair these with manual reviews for comprehensive coverage. Audits should also cover code reviews, configuration checks, and network security assessments. Findings should be documented as well as remediation or recovery plans. Threats should be addressed as soon as possible. Regular audits help maintain robust security, mitigate risks, and adapt to evolving threats and compliance requirements.

8. Active Monitoring and Logging with Alarms

As stated above in the table active monitoring and logging with alarms is essential to being able to detect threats and alarm or notify personnel in case of incidents. It allows fast responding to incidents affecting availability or compromising the application in another way. It can also allow tracing back the person responsible or to find the issue quicker so it can be addressed correctly in a fast manner. This best practice involves continuously tracking system activities, logging critical events (e.g. like 3 failed login attempts or critically high request), and generating real-time alerts for suspicious behavior or anomalies. These are the basis for intrusion detection and prevention systems which can then form threat intelligence systems.
Best practices include implementing centralized logging using tools like ELK Stack (Elasticsearch, Logstash, Kibana) or Splunk, ensuring logs are tamper-proof and retained according to compliance requirements. Real-time monitoring can be achieved with SIEM (Security Information and Event Management) solutions such as Splunk, which analyzes logs and triggers alerts for predefined security events. Regularly review and update alert rules to reflect evolving threats, and ensure the system is configured to avoid false positives (balance between security and usability is finding the optimal threshold). Effective incident response plans should complement this setup to promptly address and mitigate identified threats.

9. Incident Response Planning

Best practices include establishing a clear incident response policy, forming a dedicated incident response team, and conducting regular training and simulations. Every employee should know their responsibilities, what to do and how to contact other responsible employees. Rapid and back-up communication channels should be ensured so that incident response can be fast to ensure our high availability requirement. Post-incident reviews should be done to refine response strategies
incident response team, and conducting regular training and simulations

10. Threat Intelligence Systems (Firewalls, whitelisting, DDoS, IDPS, Content Filters)

Threat Intelligence Systems (TIS) can serve as a focused security point by proactively identifying, analyzing, and mitigating potential security threats to portals network and webservers. These systems aggregate data from various sources, including internal logs, external feeds, and security research, to provide real-time insights into emerging threats and attack trends based on known attack strategies. They should include an infrastructure of Firewalls implemented with traffic while lists, DDoS mitigation systems, Intrusion Detection and Prevention Systems (IDPS), and content filters. Many providers Unified Threat Management Systems for centralized monitoring and threat detection and Web Filtering and more features. The use of such a System will allow much easier monitoring and incident response and can also be used to filter contents that employees can access through the internal network.

11. Principle of least privilege

By adhering to PoLP, this application can minimize the attack surface, mitigate the risk of privilege escalation, and enhance overall security. This principle should be not only followed for the role-based access control but also for developers with database access rights, for processes being triggered like the backup process (e.g. not to run with root permissions) and limited access to production account... This principle should start with an “all deny” rule and only add the minimum permission needed to perform the needed task at hand. We emphasize here again that security is an iterative process, therefore the access right should also always be improved and updated to fit the PoLP when roles change, more rights are needed, employees leave the company etc.

12. Defense in Depth

The stated best practices to include (role-based) access controls, Strong encryption of data in transit and data at rest, intrusion detection/ prevention systems, firewalls, unified threat management systems, continuous logging, monitoring and alarms, and regular security assessments are all part of a bigger Security Design principle which is the concept of “Defense in Depth”. It can never be ensured that an application is completely secure. This strategy acknowledges that no single security measure is foolproof, so a combination of measures provides a more robust defense. It involves employing multiple layers of security controls throughout an information system to protect against various types of attacks with different measures providing different security methods and disaster recovery options.
More things considered here are MFA for authentication, hashing & salting & peppering of credentials (even better: maintained securely by a federated identity provider). These practices ensure that if one defense strategy fails, there will still be others to soften the fall and be in the way of attackers. Essentially this strategy makes the attack harder and thereby filters out a lot of attacks or makes them unattractive for attackers.