

Tehnilised nõuded tarkvaralahendustele

Käesolevad nõuded/põhimõtted spetsifitseerivad millele peaksid loodavad tarkvara komponendid vastama, et nad ei oleks infrastruktuurist jäigalt sõltuvuses, töötaksid SMIT majutusplatvormil, nende skaleerimist ja paigaldamist saaks automatiseerida ning tehnoloogiline võlg on hallatav ja mõõdetav. Dokument on abimaterjal arendajatele ja arhitektidele kes konkreetset tarkvaralahendust disainivad.

Kategooria	Nõue	Laiendatavus	Kommentaar
1. Üldine			
1.1	Rakendused luuakse põhimõttel, kus ühte paigaldatavasse komponenti ei panda kokku sisult väga erinevat funktsionaalsust, vaid pigem jaotatakse vajadusel erinevate komponentide vahel. Komponentideks jaotamine toimub sisulistest, mitte tehnilistest funktsionaalsustest lähtudes v.a kasutajaliideste eraldamine eraldi rakendusteks. (vt. Bounded Context)	KOHUSTUS	Komponentideks jaotamisel mitte lähtuda tehnilistest mõõdikutest nagu koodiridade arv või et tõstame näiteks tehnilise funktsionaalsuse nagu logimise eraldi komponendiks. Kui üks komponent on jäigas sõltuvuses teisest (ei saa tööprotsessi lõpuni viia või ei saa paigaldada, kui teine ei vasta samal ajal), siis on tegemist hajusa monoliidiga ja tuleks need komponendid kokku tõsta koodi tasemel. Täiendavat lugemist: http://martinfowler.com/articles/microservices.html https://martinfowler.com/bliki/BoundedContext.html https://microservices.io/microservices/general/2018/11/04/potholes-in-road-from-monolithic-hell.html
1.2	Komponent peab jooksmas vähemalt 2 instantsi peal, et vältida platvormi muudatuste käigus tekkida võimalik katkestusi.	KOHUSTUS	Konteinerplatvormile paigaldatud rakenduste puhul peaks vähemalt toodangu keskkonnas minimaalselt 2 instantsi jooksmas, et vältida katkestusi. Eelistada ka jõudluse vaates komponentide hulga tõstmist, mitte tõsta olemasolevate komponente ressursikasutust. Näiteks mitte teha VM-e suuremaks (mälu/CPU vaates).
1.3	Rakendust ehitatakse ja pakendatakse SMIT CI/CD lahendusega (Bamboo) ning komponendile kohandatakse automaatset staatilise koodi analüüsi vastavalt SonarQube profiilile, mille tulemused on kättesaadavad SMIT SonarQube keskkonnast. Lähtekood ei tohi sisaldada vigu mis on analüsaatori poolt leitud.	KOHUSTUS	Kehtib nii loogika kui kasutajaliidese komponendi puhul. Minimaalne on vaikeprofiil, kui arendajad ei lepi kokku täiendavates nõuetes. Täiendav info: Tehnoloogilise võla mõõtmine (SonarQube)
1.4	Lähtekood on UTF-8 formaadis ning tekstilised väärtused tuleb liigutada tõlkefailidesse (i8n), lisaks mitte kasutada Deprecated meetodeid.	KOHUSTUS	
1.6	Rakenduse lähtekood on kirjutatud selgusega, mis võimaldab erialast ettevalmistust omaval tarkvaraarendajal süsteemi edasi arendada.	KOHUSTUS	Rakenduse lähtekood ja kommentaarid peavad olema inglise keeles. Rakenduse ärilised muutujad aga eesti keeles, kui neile pole mõlemapoolset loogilist vastet.
1.7	Rakenduse lähtekoodi haldus toimib SMITi kesksete versioonihalduspõhimõtete järgi.	KOHUSTUS	Kehtib nii loogika kui kasutajaliidese komponendi puhul. Lisainfo leiab dokumendist: Lähtekoodi halduse ja ehitamise nõuded
1.8	Keskkondadesse rakenduste paigaldamine on automatiseeritud SMIT CI /CD mehhanismidega.	KOHUSTUS	Kasutatakse Bamboo paigaldusplaane: https://confluence.atlassian.com/bamboo/deployment-projects-338363438.html
1.8	Kokku ehitatud rakendust peab saama paigaldada erinevatesse keskkondadesse, ilma et seda peaks uuesti kokku ehitama.	KOHUSTUS	Konfiguratsioon määratakse keskkonna muutujatena või muude rakenduseväliste võimalustega (näiteks Kubernetes configmap)
1.9	Auditlogi tuleb markeerida ära logikirjetes, et oleks pärast otsingus eristatav	KOHUSTUS	
1.10	Rakendustel puudub ligipääs avalikku internetti (sh klientidel). Kui on vajalik ligipääs äriandmetele väljaspool SMIT võrku, tuleb tellida webproxy ligipääs.	KOHUSTUS	
1.11	Rakenduste sõltuvuste (maven,npm jne) allikana tuleb kasutada SMIT sisest kesket repositooriumi (Artifactory) .	KOHUSTUS	Kõik välised sõltuvused peavad olema sealt kättesaadavad.
1.12	Kõik välised sõltuvused tuleb rakenduse ehitamisprotsessikäigus läbi skaneerida Artifactory Xray töövahendiga ja mitte kasutada kõrge kriitilisusega turvanõrkuseid sisaldavaid teke/komponente.		https://jfrog.com/help/r/jfrog-cli-for-jfrog-xray

1.13	Kasutajaliidese sõltuvused/osad nagu JS,CSS,FORMATS,GIF jms, mida kasutatakse kasutajaliidese kuvamisel, peavad rakenduse käivitusel tulema samast allikast.	KOHUSTUS	Välise sõltuvuste ja repode kasutamine ei ole lubatud (nagu välised cdn-id, google fonts, github jne). Arendamisel tuleb kontrollida browseri võrguliikluse lehelt, et välisperimeetrisse ei mindaks päringuid tegema.
1.14	Välise teenuste nagu ReCaptcha, Google Analytics jms kasutamine ei ole lubatud.	KOHUSTUS	Tuleb kasutada SMIT siseseid lahendusi.
1.15	Rakenduse arhitektuuri koostamisel tuleks arvestada, et rakenduse koosseisu kuuluvaid komponente peab saama uuendada iseseisvalt.		Soovituslik kasutada "Blue-Green" paigalduspõhimõtteid.
1.16	Rakenduse loogikakomponentide üldine sisemine arhitektuur peaks järgima MVC mustrit.		
1.17	Rakendustevahelisi integratsioone teostatakse kokkulepitud põhimõtete järgi (võimalikud variandid on XTEE, HTTP otse või MQ).	KOHUSTUS	Integratsioonide disainis lähtuda dokumendist: " Rakendustevahelised integratsioonipõhimõtted "
1.18	Kui loodav lahendus koosneb mitmest komponendist, on need lahus arendatavad, pakendatavad, versioneeritavad ja paigaldatavad.	KOHUSTUS	
1.19	Rakendus on versioneeritud kasutades semantilise versioneerimise põhimõtet.		https://semver.org/ A.B.C kujul, kus C on veaparandus, B on funktsionaalne uuendus, mis töötab ka vanematel integratsioonidel ja A on integratsioone potentsiaalselt lõhkuv uuendus. Versiooni suurt numbrit A kasutatakse ka API versiooni defineerimiseks. <i>Lisasoovitus:</i> Kui major versioon saab uuenduse, siis peavad vanema versiooniga teenused hakkama tagastama päises teavitust, et versioon on <i>deprecated</i> (nt. <i>X-API-Deprecated</i>)
1.20	Toodangu keskkondades on rakendused on automaatselt monitooritud.	KOHUSTUS	
2. Testitavus			
2.1	Komponendil on olemas minimaalne genereeritav testandmete komplekt.		Peamiselt koormustestide või suitsutestide läbiviimise keskkonna jaoks.
2.2	Test- ja toodangukeskkonnad peavad olema üksteisest lahus ning testandmed ei tohi olla toodanguandmed (v.a juhul kui tegemist ei ole avalike andmetega - näiteks aadressid).	KOHUSTUS	
2.3	Loogikakomponendil on olemas integratsioonitestid veebiteenuste või sõnumivahetusteenuste jaoks ning vajadusel täiendavad ühiktestid spetsiifilise äriloogika valideerimiseks.	KOHUSTUS	Ühiktestide maht ja olemasolu on vabatahtlik.
2.4	Kasutajaliidese automaatsete kirjutamine, mis simuleerivad kasutaja käitumist põhivoogude taseme, on arendusmeeskonna otsustada.		Testide loomisel on soovitatav kasutada ilma kasutajaliidestast sirvijat, et oleks võimalik kergesti testida ilma graafilist keskkonda omamata.
2.5	Rakenduse automaatsete mahtu tuleb analüüsida CI/CD mehhanismide abil ning tulemused peab publitseerima SMIT SonarQube keskkonnas.	KOHUSTUS	Testide mahu miinimum määratakse Sonari kvaliteedivärvaga (reeglina 80% kogu koodist).
2.6	Avaklikke teenuseid tuleb täiendavalt turvatestida.	KOHUSTUS	Turvatestide juhendid ja nõuded leiab dokumendist: Turvatestimine
2.7	Rakenduste poolt toodetavale logi väljundile tuleb kirjutada automaattestid	KOHUSTUS	Testid peavad veenduma, et rakenduses genereeritud logi vastab " Logimise nõuded " dokumendis kirjeldatud nõuetele
3. Kasutajaliides			
3.1	Kasutajaliides tuleb täielikult eraldi arendada loogikakomponendist ning ei eelda selle implementatsiooni olemasolu.	KOHUSTUS	
3.2	Kasutajaliides võib korraga suhelda mitme loogikakomponendiga.		
3.3	Kasutajaliides ja loogikakomponent suhtlevad omavahel üle HTTP /WEBSOCKET protokollini ning ainult läbi eeldefineeritud liidestuse (vahetatakse ainult andmeid).	KOHUSTUS	Võimalikud integratsioonitehnoloogiad: REST, GraphQL, STOMP.
3.4	Kasutajaliideses hoitakse kasutaja sessiooni HttpOnly küpsises (Samesite, Secure), nii minimeeritakse XSS rünnakuid.	KOHUSTUS	JWT-d brauseris mitte hoida, erilahendusena tuleb see krüpteerida.
3.5	Kasutajaliides vahetab loogikakomponendiga ainult andmeid, visuaalset sisu (nn. html-i javascripti) sellest komponendist ei laeta.	KOHUSTUS	Vaikimisi on andmete formaat JSON
3.6	Kasutajaliides suhtleb loogikakomponentidega üle SSL kanali (SSL termineeritakse koormusjaoturis, kust kõik kasutaja päringud läbi lastakse).	KOHUSTUS	
3.7	Kasutajaliides on soovitatav arendada õhukese kliendina.		
3.8	Avalikud kasutajaliidesed peavad vajadusel järgima VEERA disainistandardit ning omama WCAG tuge.		

4. Äriloogika ja õigused			
4.1	Komponentide vaheline andmevahetus peab olema turvaline või kaitstud kasutades TLS-i, mille sertifikaate verifitseeritakse. Autentimata ja/või krüpteerimata protokollide kasutamisel rakendatakse täiendavaid konfidentsiaalsust ja terviklust tagavaid turvameetmeid.		
4.2	Nii sisemised kui välimised süsteemid peavad kasutaja tuvastamiseks kasutama SMIT kesksest tuvastamise teenust (UAA).	KOHUSTUS	UAA on tuvastamislahendus, mis on integreeritud lisaks sisemisele parooliga tuvastamisele ka RIA Taraga, et saaks kasutada ID-kaart, Mobiil-ID või SmartID tuvastamisvõimalusi. Rakendustega integratsiooniks kasutame Open-ID protokoll. UAA liidestamise dokumentatsioon: https://docs.cloudfoundry.org/api/uaa/version/76.13.0/index.html#overview
4.3	Kasutajaid ja nende grupi või rollipõhiseid õiguseid tuvastatakse keskst Active Directory andmebaasist. Andmetepõhised õigused (ACL) asuvad rakenduse juures andmebaasis.	KOHUSTUS	Vähemalt üks roll peab asuma AD-s (ja selle küljes vastav UAA skoop), mis määratleb kas rakendusele saab ligi. Kehtib kõigi uute arenduste puhul. AD gruppidest võetakse info ja lisatakse see UAA tokeni skoopi. Eeldus on et AD gruppide külge lisatakse skooibi info ning rakendusel ei ole vaja minna AD-sse eraldi grupinfot küsima.
4.4	Komponendid suhtlevad omavahel ainult üle HTTP või JMS/AMQP protokoll. Suurema jõudluse ja sõltumatuse saavutamiseks on eelistatud sõnumivahetus.	KOHUSTUS	Soovitus on kasutada protokolle üle turvalise kanali (HTTPS/AMQPS).
4.5	Tuvastatud kasutajal peab eksisteerima serveripoolne sessioon, millega seotud infot tuleb hoida keskses hoidlas. Kasutaja tuvastamiseks tohib kasutada ainult "Authorization Code" OAuth2 mehhanismi.	KOHUSTUS	Keskse seansi info hoidlana võib kasutada Redis või PostgreSQL-i andmebaasi. Kliendil või teenuse haldajal peab olema võimalus keskest hoidlast tokeniga seotud info kustutada (nõ välja logimine). Keskne hoidla välistab mitmekordse sisselogimise ja aeguma vastavalt kokkulepitud nõutele.
4.6	Komponendid peavad ka omavahel saama autoriseerituna (kasutades tokenit) andmeid vahetada analoogselt kasutajaliidesele ilma sessioone tekitamata.	KOHUSTUS	Kahe komponendi vaheliseks andmevahetuseks, mis ei toimu lõppkasutaja kontekstis, tuleks kasutada SVC tüüpi süsteemikontosid, millele on ADs lisatud vajalikud skooibid.
4.7	Komponentide omavahelises integratsioonis peab iga komponent omama oma kontot, ei tohi taaskasutada kontosid, mis on väljastatud teistele rakendustele.	KOHUSTUS	
4.8	Igal komponendil on oma andmebaas mille skeemi ja süsteemsete andmete muudatuse hallatakse komponendiga koos, kui komponent vajab andmete salvestamise võimekust.	KOHUSTUS	Haldab kas rakendus ise või hallatakse eraldi automaatikaga. Soovitus on kasutada Bamboo plaane selleks.
4.9	Komponendi versiooniuuendusi teostatakse reeglina ilma katkestusteta teenuse töös (tehakse nn. instantsi haaval), andmebaasi muudatuste tegemisel tuleb tagada, et muudatus töötaks ka eelmise komponentide versioonidega (kohustuslike väljade mitmeetapiline sisseviimine jms)		Üks võimalikke lahendusi on näiteks "Blue-Green deployment" mehhanism. Lisainfo: https://martinfowler.com/bliki/BlueGreenDeployment.html
4.10	Komponent käivitub ka ilma ühenduseta liidestetavate süsteemidega ehk on nn. nõrgalt liidestatud (v.a andmebaasid)	KOHUSTUS	
4.11	Komponent töötab osaliselt edasi ka liidestuste katkestuste puhul ja taastab töö peale katkestuste lõppemist. Stateful ühendused nagu AMQP peavad automaatselt taastama oma ühendused.	KOHUSTUS	Kui rakendusel mingi väline liides on maas, siis kas kasutatakse "Circuit Breakerit" või näiteks "retry" mehhanismi teatud pikkusega. Kui võrguühendus taastub, taastab komponent oma töö. Kui ootamine ei ole võimalik, peaks kindlasti "healthcheck" teada andma probleemist, mis võimaldab antud rakenduse vajadusel eemaldada teenindavate komponentide hulgast ja hiljem taastada. Spring ja Micronaut rakendustes saab kasutada @Retryable annotatsiooni.
4.12	Komponendil puudub eraldi väline konfiguratsioonifail - vajalik konfiguratsioon määratakse kas keskkonna muutujatena või kasutatakse muid platvormi pakutavaid võimalusi (Kubernetesi <i>secrets/configmap</i> näiteks).	KOHUSTUS	Rakenduse saladused, mida ei genereerita teenuste poolt automaatselt (näiteks andmebaas või s3) tuleb panna saladuste hoidlasse. Saladuste hoidlast saab saladus liikuda rakendusse ainult läbi CI/CD mehhanismi. Saladuste hoidla jaoks on SMIT-is olemas vastav teenus.
4.13	Suurte koormuste teenindamiseks vähemuutuvate andmete puhul kasutatakse rakendusserveris vajadusel vähemälusid mis on kesksed, kiired, kõrgkäideldavad ning ei sõltu konkreetse rakendusserveri instantsist. Ei kasuta mitte cachede replitseerimist vaid distributeeritud lahendust.	KOHUSTUS	Võimalik kasutada näiteks Redis või olemasolevat andmebaasi selleks ettenähtud tüüpi tabelite abil.
4.14	Komponendil on tööks kõik vajalikud teegid kaasa pakendatud, all olevast operatsioonisüsteemis mingite teekide olemasolu eeldada ei tohi.		Soovituslik komponendi pakendamisformaad on Docker konteiner, mille ehitamisel tohib kasutada ainult SMIT dockerhubis asuvad baasimageid: https://source.smit.sise/projects/dockerhub

4.15	Komponent on olekuta ehk kõik mis vaja hoida kauem kui üks süsteemiväline päring (request), salvestatakse kas andmebaasi või mõnda teise hoidlasse.	KOHUSTUS	Oleku hoidmiseks kasutatakse väliseid teenuseid.
4.16	Komponent ei tohi eeldada failisüsteemi olemasolu, kus andmed säilitatakse. Mälus võib hoida ühe päringu sees opereeritavate andmete olekuid või andmebaasist taastoodavaid cachesid.	KOHUSTUS	
4.17	Talletamiseks mõeldud binaarfailide jaoks tuleb kasutada eraldi failide hoidmise teenust, mis pakub vastavat veebipõhist teenust.	KOHUSTUS	SMIT-is on selleks olemas eraldi sisemine S3 objektihoidla.
4.18	Komponentide poolt publitseeritavad REST teenused on versioneeritud, dokumenteeritud ning veahaldust tuleb teostada HTTP veakoodidega. Teenused on peavad olema dokumenteeritud OpenAPI spetsifikatsioonile vastavalt ning spetsifikatsioon peab olema vajadusel eraldi kättesaadav kolmandatele osapooltele.	KOHUSTUS	Versiooninumber võib olla kas päises või URI-is. Teenuse tehnilises dokumentatsioonis peab olema viide OpenAPI spetsifikatsioonile (SwaggerUI). API disainimisel tuleb lähtuda dokumendist: " Rakendustevahelised integratsioonipõhimõtted ". API teenused peavad toetama JSON andmeformaati. Accept: application/json
4.19	URL ei tohi sisaldada isikuandmeid või sessioonivõtit.		Kasutada isikuandmete puhul HTTP POST ning sessioonivõtmete puhul sessiooni küpsised.
5. Andmebaas			
5.1	Andmebaasi ühenduse probleemide puhul tuleb andmete muutmise /lugemise päringutele rakendada kordust (kirjutamisel vähemalt 30s), et päringud õnnestuks, kui baasi funktsionaalsus taastub.	KOHUSTUS	Spring ja Micronaut rakendustes saab kasutada @Retryable annotatsiooni andmebaasi teenusklassides. Täpsemad näidised: Andmebaasiga suhtlemise kordamine
5.2	Andmebaaside vahelised integratsioonid ei ole lubatud.	KOHUSTUS	
5.3	Andmebaase komponentide integratsioonivahendina ei tohi kasutada (mitu erinevat komponenti ühe andmebaasi poole pöörduda ei ole lubatud).	KOHUSTUS	See põhimõte ei tähenda et ühte sama loogika komponenti jõudluse mõistes skaleerides ei tohiks kasutada sama andmebaasi.
5.4	Andmeobjektide muutmisel tuleb luua ka migratsiooniskriptid mis teisendavad automaatselt olemasolevad andmed uuele kujule. Migratsiooniskripte on soovitatav käivitada paigaldusprotsessi ühe osana ja mitte määrata neid käivitatavaks iga rakenduse restardiga (see võib tekitada ebavajalikke lukke rakenduste restartimisel, kui neid on mitu instantsi)		Kõige lihtsam variant on määrata CI/CD keskkonna muutujaga, kas migratsioon käivitada. Mõistlik on migratsiooni teha ühe intsantsiga, kui see tehtud, siis teha vajadusel uuesti paigaldus mitme instantsiga. Alternatiiv on kasutada "job /task" tüüpi paigaldusmehhanisme tavalise rakenduse paigaldamise asemel.
5.5	Andmebaasi äriloogikat vaikimisi ei kirjutata (protseduurid ja triggerid).	KOHUSTUS	Kõrvalekalde tuleb defineerida ja põhjendada konkreetse komponendi arhitektuuridokumendis.
5.6	Andmebaasi pöördutakse ainult rakenduse jaoks eraldatud süsteemsete kasutajatena.	KOHUSTUS	Mõeldud on, et andmebaasiga suheldakse läbi rakendusele eraldatud kasutaja. SYS ja POSTGRES tüüpi superuser kontodega baasis käia ei tohi.
5.7	Operatiiv- ja arhiivi andmebaasid on eraldi lahendused, kasutatakse kas eraldi arhiivibaase või mõnda muud spetsiaallahendust.	KOHUSTUS	
5.8	Tekstiotsingute jaoks kasutatakse ainult täisteksti indekseid (Lucene või andmebaasi sisemine täistekstiotsing).	KOHUSTUS	
5.9	Objektid identifitseerida registrikoodide abil.		Riiklikesse registritesse kantavad objektid (isikud, katastriüksused jne) kantakse andmebaasi nende registrikoodiga, mida täiendab riigiprefiks vastavalt ISO3166-1 Alpha 2 standardile. Näiteks isikute sidumiseks süsteemi kasutajakontoga peab kasutama isikukoodi rahvastikuregistrist. Eesti Vabariigi kodanik identifitseeritakse Eesti Vabariigi poolt väljastatud eIDga. Igasuguse muu identifitseerimisevahendi kasutamine peab olema selgelt põhjendatud. Mittekodanike isikuidentifikaator saadakse järgmisel viisil: riigikood + sookood + sünniaeg + [dok_nr id_riigis], kus riigikood - kolmekohaline ISO 3166-1 Alpha-3 standardile vastav riigi kood sookood - soo identifikaator nii nagu Eesti Vabariigi isikukoodis sünniaeg - sünniaeg formaadis YYYYMMDD id_riigis - kui see on olemas, tuleb kasutada isiku koduriigi isikuidentifikaatorit. 16 kohta, 0-polsterdatud vasakult dok_nr - kui isiku koduriigis isikuidentifikaatorit ei ole, siis kasutatakse isiku dokumendi numbrit. Dokumendi number, 16 kohta, 0-polsterdatud vasakult.
6. Jõudlus			

6.1	Komponendi sisemised alamosad suhtlevad omavahel võimalusel sõnumivahetuse või muude asünkroonsete meetodite abil, et vältida blokeeruvaid lõimesid ja tagada et iga alamtöö töötab eraldi lõimes, kasutades efektiivselt virtuaalmasinate mitut protsessorituumat - see soovitus ei kehti kui komponentide suhtlus peab toimuma ühe kasutaja transaktsiooni sees (mitte segamini ajada andmebaasi transaktsiooniga).		Lisainfo: http://12factor.net/concurrency , http://www.reactivemanifesto.org/
6.2	Pikalt töötavad operatsioonid tuleb viia eraldi taustaprotsessideks, mis toimivad ka mitmete instantside puhul, ehk taustatööde info peab olema salvestatud.	KOHUSTUS	
6.3	Komponent peab peatumisel lõpetama käimasolevad protsessid ning pikad taustaprotsessid peavad pooleli jääma ja käimasolev töö tuleb tagastada tööde nimekirja (näiteks "queue").	KOHUSTUS	Lisainfo: http://12factor.net/disposability
6.4	Komponendis loodud protsessid peavad arvestama, et võib toimuda ootamatu rakenduse seiskumine ning selle tulemusel peavad samuti käimasolevad tööd minema järjekorda tagasi.	KOHUSTUS	
6.5	Jõudluse kasvamisel saab automaatselt komponentide instantside hulka tõsta (horisontaalne skaleerimine) ning koormus jaotatakse instantside vahel laiali.		Ei tohiks kasutada nõ. "scale-up" lahendust, kui seda ei määra konkreetne toote pakkuja.
6.6	Komponent peaks käima minema (olema valmis esimeste päringut teenindamiseks) kuni 60 sekundi jooksul		
7. Monitooritavus			
7.1	Komponentide poolt publitseeritavad teenuste monitoorimiseks vajalik info on kättesaadav kokkulepitud formaadis ja protokolliga - soovitatavalt REST formaadis.	KOHUSTUS	Koormusjaotur peab aru saama, kas suunata liiklus konkreetsele nodele või mitte. Kõige lihtsam näide või näidis selleks on Spring Boot Actuator endpoint /health mis annab infot, kas rakendus on korras või mitte. Seda saab vastavalt täiendada ka lisainfoga. https://docs.spring.io/spring-boot/api/rest/actuator/health.html
7.2	Komponenti on võimalik monitoorida APM tarkvara agendiga, mis oskab komponendi seest kõikide lähtekoodis realiseeritud protsesside kohta statistikat (sh. kasutamise sagedus, töötuse aeg) väljastada.		Toetatud tehnoloogiad leiab siit: https://docs.appdynamics.com/display/PRO45/App+Server+Agents+Supported+Environments
7.3	Komponent logib enamus tegevused erinevatel tasemetel ning suunab need "stdouti" või sõnumitena sõnumiserverisse (auditlogi). Komponent ei tegele logifailide haldamisega.	KOHUSTUS	Lisainfo: http://12factor.net/logs
8. Failihaldus			
8.1	Kui teenus võtab vastu faili, siis tuleb seda skaneerida enne baasi või objektihoidlasse salvestamist.	KOHUSTUS	Kasutada tuleb SMIT-i poolt pakutavat MetaVault teenust.
9. Logimine			
9.1	Erindite (Exception) kinnipüüdmisel tuleb logisse salvestada kogu stacktrace, mitte ainult veateade.	KOHUSTUS	Lisainfot leiab dokumendist " Logimise nõuded "
	Logimisel tuleb arvesse võtta Infoturbeosakonna poolt kehtestatud logimise nõudeid.	KOHUSTUS	
9.2	Rakenduse tehnilised komponendid logivad korrelatsiooni ID'd või genereerivad selle ise. Korrelatsiooni ID'd saadetakse iga edasise päringuga kaasa.		
9.3	Logi peab olema JSON formaadis.	KOHUSTUS	Võib kasutada näiteks: net.logstash.logback.encoder.LogstashEncoder
10. Taaskasutus			
10.1	Uut komponenti arendades, tuleks eelnevalt kontrollida, ega sellist pole juba eelnevalt SMIT-is välja töötatud. Eelistatud on kasutada olemasolevat lahendust teenusena ning äärmisel juhul paigaldada endale toode.	KOHUSTUS	
10.2	Olemasoleva teenuse koodi eraldamine eraldi projektiks ja selle iseseisev edasiarendus peaks saama SMIT arhitektuuripaneeli heakskiidu, sellele võiks eelneda diskussioon olemasoleva tiimiga, et vajalikud muudatused nemad viiks sisse või aksepteeriks muudatusettepanekuid.	KOHUSTUS	
10.3	Kui arendatakse teenust, millele hakkavad olema kliendid väljaspool konkreetset infosüsteemi/lahendust, on soovitatav luua sellele iseseisev tehnilise liidestuse võimekus.		Üks võimalik tehniline lahendus on Kuberneteses Operator (https://operatorframework.io)
11. Arvutitöökoht			
11.1	Uued seadmed/lahendused peavad olema Windows või IOS põhised	KOHUSTUS	
11.2	Rakendus/lahendus peab olema suuteline vähemalt minor versioone ise uuendama.	KOHUSTUS	

11.3	Rakenduse paigalduse või uuendamise käigus EI TOHI rakendus arvuti SERVICEID kinni "Disabled" panna (Näide... DocCheck pani ID kaardi serti edastuse teenuse kinni arvutites)	KOHUSTUS	
11.4	Rakendus/lahendus ei tohi olla seotud ühe konkreetse lehitseja või selle versiooniga. Kõik Tehnoloogia valikute lehel olevad lehitsejad peavad olema toetatud ja testitud.	KOHUSTUS	
11.5	Kui soovitakse arvutitöökohta paigaldada mingit rakendust/lahendust, siis tarkvara koos konfiguratsiooniga tuleb arenduselt ATKH-le	KOHUSTUS	
11.6	Valdkond peab kontrollima enda lahendust standard ATK profiilis või eriprofiilis SMIT tehnikupoolt väljastatud seadmes ja kinnitama kas paigaldus on edukas ja vastab teenuse pakkumise nõuetele. Kui ei ole, siis annab infot mida on vaja konfiguratsioonis muuta.	KOHUSTUS	
11.7	Rakendus peab olema käsurealt paigaldatav.	KOHUSTUS	
11.8	Rakendus ei tohi nõuda administreerimise õigust kasutamiseks	KOHUSTUS	