

Soluție tehnică bazată pe arhitectura pe .NET pentru Autoritatea Aeronautică Civilă

Pentru proiectul Achiziționarea serviciilor de realizare și automatizare a fluxurilor de procesare în cadrul procesului de evidență a aeronavelor înmatriculate, optimizarea cooperării dintre autoritățile publice, prin implementarea unei soluții moderne de e-guvernare pentru sporirea transparenței procesului de evidență a aeronavelor înmatriculate în cadrul Registrului Aerian al Republicii, compania noastră va pune la dispoziție o echipă cu expertiză tehnică necesară, capabilă să dezvolte o Sistemul informațional web care să răspundă obiectivelor stabilite de Beneficiar.

Proiectul va fi o soluție bazată pe tehnologii WEB și va oferi o interfață compatibilă cu majoritatea browserelor moderne (Microsoft Edge, Mozilla Firefox, Opera, Google Chrome sau Safari). Din punct de vedere funcțional, se va dezvolta o soluție fiabilă și scalabilă atât în cazul creșterii numărului de utilizatori concurenți, cât și în cazul creșterii volumului de informații gestionate de aceasta. Sistemul informațional va fi dezvoltat ținând cont de tehnologiile moderne, nevoile și perspectivele de utilizare, integrare și scalabilitate.

Indrivo se angajează să presteze serviciile descrise în această ofertă prin limitarea resurselor proprii fără a încheia subcontracte cu alte companii și organizații.

Mai jos, vă rugăm să găsiți toate informațiile necesare despre tehnologia aleasă, soluția tehnică în sine și toate detaliile necesare implementării cu succes a proiectului.

1 Aspecte tehnice ale sistemului

1.1 Stiva de tehnologie

1.1.1 Termeni

- **Arhitectură Monolitică** - Un stil arhitectural în care toate componentele unei aplicații sunt integrate într-un singur bloc sau unitate, care funcționează ca un sistem compact.
- **Modularitate** - Procesul de împărțire a unui sistem software în module independente, care pot fi dezvoltate și întreținute separat.
- **Serviciu** - O componentă funcțională autonomă care îndeplinește o anumită sarcină în cadrul sistemului.
- **API (Interfață de Programare a Aplicațiilor)** - Set de reguli care permite diferitelor componente software să comunice între ele.
- **Interfață** - Punct de interacțiune între diferite module sau între utilizator și sistem.
- **Dependență** - Relația dintre module, în care un modul se bazează pe un altul pentru a-și îndeplini funcționalitatea.
- **Scalabilitate** - Capacitatea sistemului de a-și menține performanța sau de a crește capacitatea în mod eficient atunci când cerințele cresc.

1.1.2 Abrevieri

- **MVC (Model-View-Controller)** - Un model arhitectural care separă logica aplicației în trei componente principale: model (date), vedere (interfață utilizator) și controler (logica aplicației).
- **ORM (Object-Relational Mapping)** - Un mecanism care permite conversia datelor între sisteme de tip obiect și baze de date relaționale.
- **CI/CD (Continuous Integration/Continuous Deployment)** - Procesul automatizat de integrare și livrare continuă a modificărilor de cod în sistemul de producție.
- **REST (Representational State Transfer)** - Un stil arhitectural pentru crearea de servicii web care utilizează HTTP pentru a comunica între module.
- **SOA (Service-Oriented Architecture)** - Un stil arhitectural în care funcționalitatea aplicației este împărțită în servicii care interacționează între ele.
- **ESB (Enterprise Service Bus)** - O arhitectură middleware care facilitează integrarea și comunicarea între diferitele servicii dintr-o aplicație monolitică modulară.
- **DBMS (Database Management System)** - Un sistem software care permite crearea, gestionarea și manipularea bazelor de date.

1.1.3 Dezvoltare

Sistemul informational va fi construit folosind cea mai recentă stivă de tehnologie din ecosistemul ASP.NET Core. Aceasta include pachete și SDK-uri bine stabilite, întreținute și securizate. Un proces automatizat folosind Dependabot va fi configurat pentru întregul sistem pentru a se asigura că pachetele și SDK-urile rămân actualizate și că orice alerte privind vulnerabilitățile nou descoperite sunt abordate cu promptitudine.

Exemple de instrumente:

- Framework ASP.NET Core – Cadrul pentru construirea de API-uri web
- ASP.NET Core Razor/Blazor - Cadrul UI
- Entity Framework – Integrare cu sursele de date pentru cantități moderate de date
- Hangfire / Quartz – Programarea lucrărilor de fundal
- MassTransit – Integrare agnostică cu cozile de mesaje
- Dapper – Integrare cu sursele de cantități mari de date
- Redis – Mecanisme de lucru cu cache
- etc.

Deciziile de utilizare a unui instrument specific vor fi revizuite după analiza cerințelor, deoarece pot influența direct sau indirect deciziile.

Serviciile incluse în Sistemul informational vor fi dezvoltate ca soluții separate și independente din punct de vedere tehnic, cu utilizarea cadrului FOD și aplicarea conceptelor FOD, cum ar fi separarea front-office de back-office, expunerea API, comunicarea bazată pe mesaje etc., pentru a asigura integrarea conceptuală a noilor servicii dezvoltate cu întregul ecosistem digital și pentru a simplifica integrarea cu serviciile periferice la nivelul platformei.

1.1.4 Timp de rulare

Toate componentele sistemului vor fi containerizate și furnizate ca imagini Docker bazate pe Linux. Această abordare va permite sistemului să fie agnostic față de hardware-ul de bază pe care este implementat.

1.2 Considerații arhitecturale

1.2.1 Abordare arhitecturală

Sistemul informational va fi construit folosind un stil de arhitectură orientat spre servicii. Sistemul va fi împărțit în zone logice de funcționalitate (denumite în continuare servicii) care, integrate, vor aborda cazurile de utilizare de bază ale sistemului. Notă: Fiecare serviciu poate fi implementat, ca o instanță fizică separată. Această abordare va aduce următoarele beneficii:

- **Încapsulare:** Fiecare serviciu va încapsula date și comportamente conexe care oferă valoare pentru sistem. Toate modificările sunt limitate în această unitate încapsulată, permițând fiecărui serviciu să fie dezvoltat, implementat și menținut independent.
- **Scalabilitate îmbunătățită:** Arhitectura îmbunătățește scalabilitatea sistemului.
- **Izolarea greselilor :** Eșecurile pot fi izolate la nivelul fiecărui microserviciu, prevenind afectarea întregului sistem.
- **Cicluri de dezvoltare mai rapide:** Sistemul se poate adapta la noile cerințe și situații mai rapid.

1.2.1.1 Convențiile de denumire

Pentru a evita neînțelegerile induse de decalajul model-cod – o situație în care diagramele de arhitectură includ concepte abstracte nereprezentate direct în limbajul de programare (de exemplu, module, straturi, servicii, componente, containere) – va fi adoptat un vocabular comun.

Vocabularul se va baza pe o variație a standardului C4 aprobat la nivel internațional, componentele acestuia fiind:

Diagrama contextului sistemului:

- **Scop:** Oferă o vedere la nivel înalt a sistemului și mediului în care operează.
- **Componente:** Sistemul (software-ul luat în considerare) este reprezentat ca un singur bloc, cu actori externi sau sisteme reprezentate prin blocuri separate..
- **Detalii:** Această diagramă arată modul în care sistemul interacționează cu entitățile externe, dar nu are scopul de a analiza detaliile interne ale sistemului.

Diagrama de nivel container:

- **Scop:** Se concentrează pe containerele din sistem și pe interacțiunile acestora.
- **Componente:** Containerele reprezintă aplicații sau servicii, cum ar fi servere web, baze de date, cozi, keyvault's, aplicații desktop etc. Fiecare container poate conține mai multe componente.
- **Detalii:** Această diagramă arată containerele majore și relațiile lor, oferind o vedere mai detaliată decât diagrama de context.

Diagrama componentelor:

- **Scop:** Oferă evaluare mai atentă a interiorului fiecărui container, arătând componentele majore și interacțiunile lor.
- **Componente:** Reprezintă componente individuale în cadrul fiecărui container, cum ar fi clase, module, servicii etc.
- **Detalii:** Ilustrează modul în care componentele lucrează împreună pentru a atinge funcționalitatea componentei.

Diagrama codului (claselor):

- **Scop:** Oferă o vedere detaliată a structurii interne a unei componente.
- **Componente:** Reprezintă clase sau module și relațiile lor.
- **Detalii:** Oferă o privire mai atentă asupra structurii interne a unei componente, arătând clasele cheie și interacțiunile acestora.

1.2.1.2 Vedere logică

Sistemul informational va fi împărțit în mai multe zone logice de funcționalitate. Aceste zone vor defini limite clare în care modelele specifice sunt definite și aplicate, asigurându-se că limbajul, regulile și conceptele fiecărui model rămân consecvente și coezive în contextele lor respective.

Aceste zone logice de funcționalitate (servicii) se vor mapa la definiția unei componente în diagramele C4. Detaliile de implementare ale unui anumit serviciu vor fi încapsulate în spatele unei interfețe bine definite, denumite și contracte (descrise mai jos). Fiecare componentă își va expune interfața în scopuri de integrare și va utiliza interfețele altor componente pentru a facilita integrarea. Contractele vor varia în funcție de strategia de integrare aleasă.

1.2.1.3 Vedere fizică

Fiecare componentă a sistemului va avea capacitatea de a fi implementată ca o unitate de sine stătătoare.

Notă: În anumite situații, vizualizarea fizică poate diverge de cea logică pentru un anumit sistem din cauza constrângerilor variate.

Notă: discutat mai detaliat la punctul 2.2 Considerații de aspect fizic.

1.2.2 Structura componentelor

Fiecare componentă va fi proiectată în funcție de cazurile de utilizare specifice pe care le va acoperi. Cu toate acestea, pentru a accelera dezvoltarea, este recomandabil ca aceștia să împărtășească structuri interne similare.

1.2.2.1 Integrare

1.2.2.1.1 Stratul contractual

Fiecare componentă a sistemului va oferi un API public pentru integrare. Aceasta va include structuri precum:

- DTO-uri
- Evenimente
- Validatori
- Interfețe publice

Contractele vor expune funcționalitatea disponibilă a componentei fără a divulga detalii suplimentare de implementare.

1.2.2.1.2 Stratul de rutare

Fiecare componentă va conține un construct care va permite să servească drept punct de mapare între contract și părțile externe. Un exemplu simplu este o componentă care va expune același API public pentru integrare prin stratul de rutare ca:

- gRPC
- API-ul REST
- GraphQL API
- Puncte de intrare asincrone de integrare prin procesoare de mesaje

1.2.2.2 Stratul de logica business

Va implementa contractele definite în stratul de contracte. Acest strat nu ar trebui să fie accesibil din exterior decât prin componenta de găzduire. Stratul de afaceri va încapsula toate datele și aspectul comportamentului oricărei componente.

1.2.2.3 Infrastructura de hosting

Va încapsula toate structurile de cod necesare pentru găzduirea componentei ca unitate autonomă.

1.2.2.4 Stratul de utilități

Va conține structuri de cod reutilizabile care pot fi partajate pe mai multe straturi. Acestea includ lucruri precum:

- Managementul configurației
- Cache
- Logare
- Controale de sănătate
- etc.

1.2.3 Modele de comunicare

Fiecare componentă va expune un contract printr-un strat de rutare. Acest lucru facilitează integrarea altor componente cu acele servicii pentru a interoga sau modifica datele.

Integrarea dintre componentele sistemului și dependențele acestora va urma fie modelul sincron, fie asincron.

1.2.3.1 Comunicare sincronă

Configurare simplă în care o componentă o apelează pe cealaltă prin interfața sa disponibilă public.

Beneficii:

- Ușurință de implementare
- Ușor de urmărit
- Raspuns imediat
- Viteza de raspuns marita

Compensații:

- Cuplarea in timpul de rulare între componenta apelant și dependentă.
- Poate duce la probleme de performanță și întârzieri din cauza latenței rețelei.
- Poate duce la probleme de scalabilitate atunci când numărul de utilizatori crește.

1.2.3.2 Comunicare asincronă printr-o coadă de mesaje

O componentă ar publica un mesaj către un broker/coada de mesaje, iar componentele interesate l-ar prelua de acolo. Părțile interesate sunt componentele care ar putea fi abonate la evenimentele unei componente, acestea fiind definite în stratul de contracte.

Beneficii:

- Permite aplicației sa scalezeprin reducerea impactului blocării apelurilor și operațiunilor I/O pe server.
- Poate oferi toleranță la erori, permițând serviciilor să continue să funcționeze în cazul unor defecțiuni sau întreruperi temporare.

Compensații:

- Un singur punct de eșec sub formă de broker de mesaje.
- Poate introduce o latență suplimentară în cazul în care mesajele sunt întârziate sau dacă suprasolicitarea de așteptare și procesare a mesajelor este semnificativă.
- Complexitate crescută a soluției.

1.2.4 Izolarea datelor

Regula generală: Fiecare componentă va incapsula/izola datele sale respective.

Fiecare componentă din interiorul unui sistem ar trebui să fie autonomă. Menținerea componentelor izolate una de cealaltă ajută la creșterea modularității și a cuplajului liber. Orice mutație sau interogare de date necesară ar trebui să fie disponibilă ca parte a interfeței publice și utilizată în consecință.

Cuplarea la nivel de date va reduce scalabilitatea și rezistența întregului sistem.

1.2.5 Integrare cu sisteme interne și externe

Pentru a facilita integrarea între componentele interne, fiecare componenta va genera un pachet client (adică SDK) în scopul integrării.

Integrarea cu sistemele externe se va face printr-un plan de integrare care va proteja și acționa ca un strat anticorupție pentru sistem de modificările acestor dependențe.

1.2.6 Considerații de performanță

Sistemul va fi să fie capabil de:

- Permitearea unui minim de 200 de utilizatori autorizați să lucreze simultan.
- Asigurarea activității concomitente pentru cel puțin 150 de utilizatori autorizați.
- Răspunsul la cel puțin 100 de interogări simultane de la servicii fără a afecta performanța operațională.
- Servirea a peste 500.000 de utilizatori anonimi pe an.

- Gestionarea a minimum 500 de utilizatori anonimi concurenți și 300 de interogări paralele.
- Primirea, procesarea și stocarea datelor din peste 20.000 de tranzacții legate de evenimentele din profilul Unității Statistice.
- Asigurarea că timpul mediu de răspuns al serverului nu depășește 3 (trei) secunde sub sarcina nominală a sistemului.

2 Structura sistemului

2.1 Componentele logice și capacitățile acestora

2.1.1 Serviciu de autentificare

Serviciul de autentificare este o componentă vitală pentru securizarea Platformei, însărcinat în primul rând cu verificarea identităților utilizatorilor sau entităților care încearcă să acceseze un sistem sau resursele sale specifice. Acest serviciu va asigura că numai persoanele sau sistemele autorizate au acces la date sau funcționalități protejate. Aspectele și funcționalitățile cheie ale serviciului de autentificare vor include:

- Verificarea identității utilizatorului: confirmă identitățile utilizatorilor prin metode precum nume de utilizator și parole, date biometrice (de exemplu, amprentă digitală sau recunoaștere facială), carduri inteligente sau autentificare cu mai mulți factori (MFA), care combină mai mulți factori de autentificare pentru o securitate sporită.
- Controlul accesului: impune politici pentru a determina ce resurse sau acțiuni poate accesa sau efectua un utilizator pe baza identității sale autentificate și a permisiunilor atribuite.
- Managementul utilizatorilor: oferă instrumente pentru înregistrarea utilizatorilor, crearea contului, gestionarea parolilor (de exemplu, resetări) și dezactivarea sau suspendarea contului în cazul unor încălcări de securitate.
- Managementul sesiunii: Urmărește activitatea utilizatorului în timpul sesiunilor autentificate, asigurând securitatea sesiunii și timeout-uri adecvate pentru a preveni accesul neautorizat.
- Protocoale de securitate: utilizează protocoale precum OAuth, OpenID Connect sau SAML pentru a activa conectarea unică (SSO) și pentru a facilita accesul utilizatorilor la mai multe aplicații sau servicii.
- Autentificare bazată pe jetoane: emite jetoane la autentificarea cu succes pentru a verifica cererile de acces ulterioare.
- Înregistrare și audit: Urmărește încercările de autentificare și menține înregistrări detaliate pentru analiza de securitate și conformitate.
- Politici privind parolele: Permite administratorilor să configureze și să aplice cerințele privind parola, inclusiv complexitatea, expirarea și istoricul, pentru a îmbunătăți securitatea.
- Capabilități de integrare: se integrează cu diverși furnizori de identitate, cum ar fi Active Directory, LDAP sau furnizori terți, pentru a valorifica bazele de date și sursele de identitate existente.
- Securitate API: Include caracteristici pentru autentificarea și autorizarea solicitărilor API de la sisteme sau aplicații externe.

Serviciul de autentificare va fi integrat cu MPass, asigurând verificarea identității perfectă și sigură în întregul sistem. Această integrare va îmbunătăți capacitățile Serviciului de autentificare prin valorificarea caracteristicilor robuste ale MPass pentru gestionarea identităților utilizatorilor și a controlului accesului. Cu MPass, Serviciul de Autentificare va beneficia de măsuri de securitate suplimentare, gestionarea eficientă a utilizatorilor și accesul îmbunătățit la resurse, consolidând și mai mult securitatea generală și eficiența sistemului.

2.1.2 Serviciul de autorizare

Serviciul de autorizare, strâns integrat cu un serviciu de autentificare, este esențial pentru gestionarea a ceea ce utilizatorii sau entitățile autentificate au voie să facă în cadrul unui sistem. Autorizarea definește și impune politicile de control al accesului, specificând acțiuni, date sau resurse accesibile utilizatorilor pe baza identității și a permisiunilor acestora. Aspectele cheie ale unui serviciu de autorizare includ:

- Politici de control al accesului: definește și impune cine poate accesa ce și în ce condiții (de exemplu, oră, locație sau dispozitiv).
- Controlul accesului bazat pe roluri (RBAC): Atribue roluri utilizatorilor, cu permisiuni legate de aceste roluri, simplificând gestionarea accesului.
- Permisuni detaliate: Permite administratorilor să specifice controlul detaliat al accesului la nivelul resurselor individuale sau al câmpurilor de date.
- Autorizare dinamică: ia decizii de acces în timp real pe baza unor factori contextuali precum atributele utilizatorului, caracteristicile dispozitivului și evaluările riscurilor.
- Managementul politicilor: Include instrumente pentru definirea, gestionarea și actualizarea politicilor de control al accesului.
- Protecția resurselor: Se asigură că numai utilizatorii autorizați pot accesa date sensibile, API-uri sau funcționalități.
- Refuzarea accesului neautorizat: Împiedică utilizatorii să acceseze resurse sau să efectueze acțiuni pentru care nu au permisiunea.
- Controlul accesului bazat pe atribute (ABAC): folosește atributele utilizatorului și al resurselor pentru a lua decizii de acces.
- Delegarea de autoritate: Permite administratorilor să delege deciziile de control al accesului pentru anumite resurse sau funcționalități anumitor utilizatori sau roluri.
- Securitate API: Extinde controlul accesului la punctele finale API, asigurându-se că numai clienții API autorizați pot face solicitări.

Serviciul de autorizare va fi integrat cu MPower, sporind capacitatea acestuia de a gestiona permisiunile utilizatorilor și controalele de acces în cadrul sistemului. Această integrare folosește caracteristicile avansate ale MPower pentru a se asigura că numai utilizatorii autorizați pot accesa anumite resurse și pot efectua acțiuni desemnate. Prin combinarea Serviciului de Autorizare cu MPower, sistemul va beneficia de un management al accesului mai robust și mai dinamic, securitate îmbunătățită și o gestionare mai eficientă a permisiunilor pentru toate modulele.

2.1.3 Serviciul de formulare

Serviciul va oferi un mecanism de configurare a formularelor personalizate necesare perfecționării documentelor legate de procesele de afaceri. Cu acest serviciu, formularele și șabloanele personalizate vor fi configurate și implementate pentru a genera toate documentele specifice aplicației.

2.1.4 Serviciul de management al șabloanelor

Serviciul de șabloane va permite utilizatorilor să creeze, să stocheze și să utilizeze șabloane predefinite pentru diferite tipuri de documente, e-mailuri sau comunicări. Acest serviciu simplifică și standardizează procesul de generare a conținutului consistent și cu aspect profesional prin furnizarea de șabloane reutilizabile. Iată care sunt aspectele și funcționalitățile cheie asociate de obicei cu un serviciu de șabloane:

- Crearea șabloanelor pentru documente precum contracte, propuneri, facturi, e-mailuri sau rapoarte. Șabloanele acceptă adesea diverse formate de fișiere, inclusiv documente text, foi de calcul și șabloane de e-mail.
- Personalizare pentru a se potrivi cerințelor specifice de branding, stil și conținut. Utilizatorii pot personaliza șabloanele adăugând sigle ale companiei, anteturi personalizate și substituenți dinamici pentru informații variabile.
- Controlul versiunilor pentru a menține coerența documentului și a urmări modificările. Acest lucru îi ajută pe utilizatori să urmărească revizuirile șablonului și să revină la versiunile anterioare dacă este necesar.

Serviciul de șabloane simplifică crearea documentelor, îmbunătățește coerența și economisește timp, oferind utilizatorilor o bibliotecă de șabloane pre-proiectate care pot fi personalizate, completate cu date relevante și utilizate pentru diferite tipuri de documente și comunicații în cadrul unei organizații.

2.1.5 Serviciul de notificare

Serviciul de notificări este conceput pentru a oferi utilizatorilor alerte și actualizări în timp real despre evenimente importante, activități sau modificări relevante pentru munca sau interacțiunile lor în cadrul aplicației. Aceste notificări pot fi sub formă de mesaje pop-up, e-mailuri sau notificări în aplicație și servesc pentru a menține utilizatorii informați și implicați. Caracteristicile cheie includ:

- Alerte personalizabile: utilizatorii pot personaliza tipurile de notificări pe care le primesc și modul în care sunt livrate (de exemplu, e-mail, pop-up) în funcție de preferințele și prioritățile lor.
- Actualizări în timp real: notificările sunt declanșate de evenimente sau acțiuni specifice din cadrul aplicației și sunt livrate imediat pentru a se asigura că utilizatorii sunt la curent cu schimbările importante pe măsură ce apar.
- Urmărirea activității: notificările includ informații despre mesaje noi, actualizări ale înregistrărilor, atribuirii de sarcini, întâlniri viitoare sau alte activități relevante.
- Acțiuni pe care se poate face clic: utilizatorii pot lua măsuri direct din notificări, cum ar fi răspunsul la mesaje sau accesarea înregistrărilor sau sarcinilor asociate cu un singur clic.
- Prioritizare: notificările sunt prioritizate în funcție de urgență sau importanță, permițând utilizatorilor să se concentreze asupra sarcinilor și actualizărilor critice.
- Notificări în aplicație: Sistemul informational va avea un centru de notificări centralizat unde utilizatorii își pot vizualiza și gestiona toate notificările într-un singur loc.
- Preferințe utilizator: utilizatorii pot configura setările de notificare, cum ar fi frecvența, sunetele de notificare și dispozitivele sau canalele prin care primesc notificări.
- Modulul de notificări îmbunătățește implicarea utilizatorilor, productivitatea și gradul de conștientizare prin furnizarea de actualizări în timp util și relevante despre evenimente și activități din cadrul unei aplicații software. Ajută utilizatorii să rămână informați și să ia măsuri imediate atunci când este necesar.

2.1.6 Serviciul fluxuri de lucru

Acest serviciu permite crearea fluxurilor de lucru și atribuirea acestora unor entități. Fluxul de lucru va permite utilizatorilor să configureze fluxuri logice pe care trebuie să le urmeze informațiile procesate în cadrul sistemului IT. Administratorul va seta stări și tranziții între stări, astfel încât aplicația să simuleze fluxul de afaceri de care clientul are nevoie. De asemenea, acest serviciu permite definirea acțiunilor pentru tranziții și stări pentru roluri specifice.

Fluxurile de lucru care urmează a fi implementate în cadrul Sistemul informational au o complexitate redusă și presupun în mare măsură acțiuni care vizează pregătirea formularelor, schimbarea etapelor, stărilor, aprobarea/respingerea și procesarea acțiunii formularului. În timpul fazei de analiză a afacerii, toate fluxurile de lucru Sistemul informational vor fi stabilite și documentate.

2.1.7 Serviciu de căutare

Serviciul de căutare va oferi o căutare globală în toate modulele și înregistrările din sistem. Acesta va permite căutarea oricărui atribut de metadate ale documentelor sau informațiilor publicate folosind atât „&&” (și) cât și „||” (sau) operatori. Utilizatorii pot prelua informații după diverse criterii, cum ar fi cuvinte cheie, autori, data creării și conținutul textului. Utilizatorii interni pot efectua căutări simple folosind șiruri de căutare sau pot efectua căutări mai complexe pentru filtrare precisă (QBE). Indiferent de tipul de informații, utilizatorii interni vor utiliza o metodă consecventă pentru interogarea și preluarea datelor din toate partițiile de sistem.

2.1.8 Serviciul de rapoarte

Serviciul de rapoarte este conceput pentru a genera diverse rapoarte și statistici la cerere. În funcție de nivelul de acces și permisiuni, rapoartele vor rezuma informațiile solicitate.

Generatorul de rapoarte este un mecanism universal încorporat bazat pe principiul: utilizatorul formulează condiția în care sistemul generează un raport (pe baza filtrelor puse la dispoziție și identificate în timpul proiectării sistemului). Nivelul de acces pe raport va fi configurat, astfel încât vom evita accesul neautorizat la informațiile de sistem centralizate. Rapoartele pot fi exportate într-un fișier extern editabil (PDF, XLS, XLSX,

CSV, DOC, DOCX). În mod implicit, rapoartele trebuie preluate în format PDF, în timp ce diagramele – în format JPG/PNG.

Serviciul de generare de rapoarte adună informații din sistem, folosind toate datele introduse manual sau automat în baza de date. Serviciul va genera documente standard și rapoarte statistice specifice fluxurilor de lucru implementate și proceselor decizionale. Serviciul va oferi rolurilor administrative un număr standard de șabloane configurabile de documente/rapoarte standardizate, permițând producerea de noi documente/rapoarte atunci când este necesar.

2.1.9 Serviciul REST API

Serviciul REST API stabilește o conexiune directă sigură între Sistemul informational și Aplicația Clientului, oferind un sistem de comunicație bidirecțională în timp real, capabil să gestioneze volume mari de informații.

Serviciul REST API va asigura integrarea și comunicarea bidirecțională cu alte servicii prin Servicii Web Securizate. De asemenea, modulul va folosi RESTful Web Services pentru transfer automat și securizat de date de la celelalte servicii și invers. Serviciul utilizează standarde web bazate pe arhitectura REST și utilizează protocolul HTTPS pentru comunicarea datelor

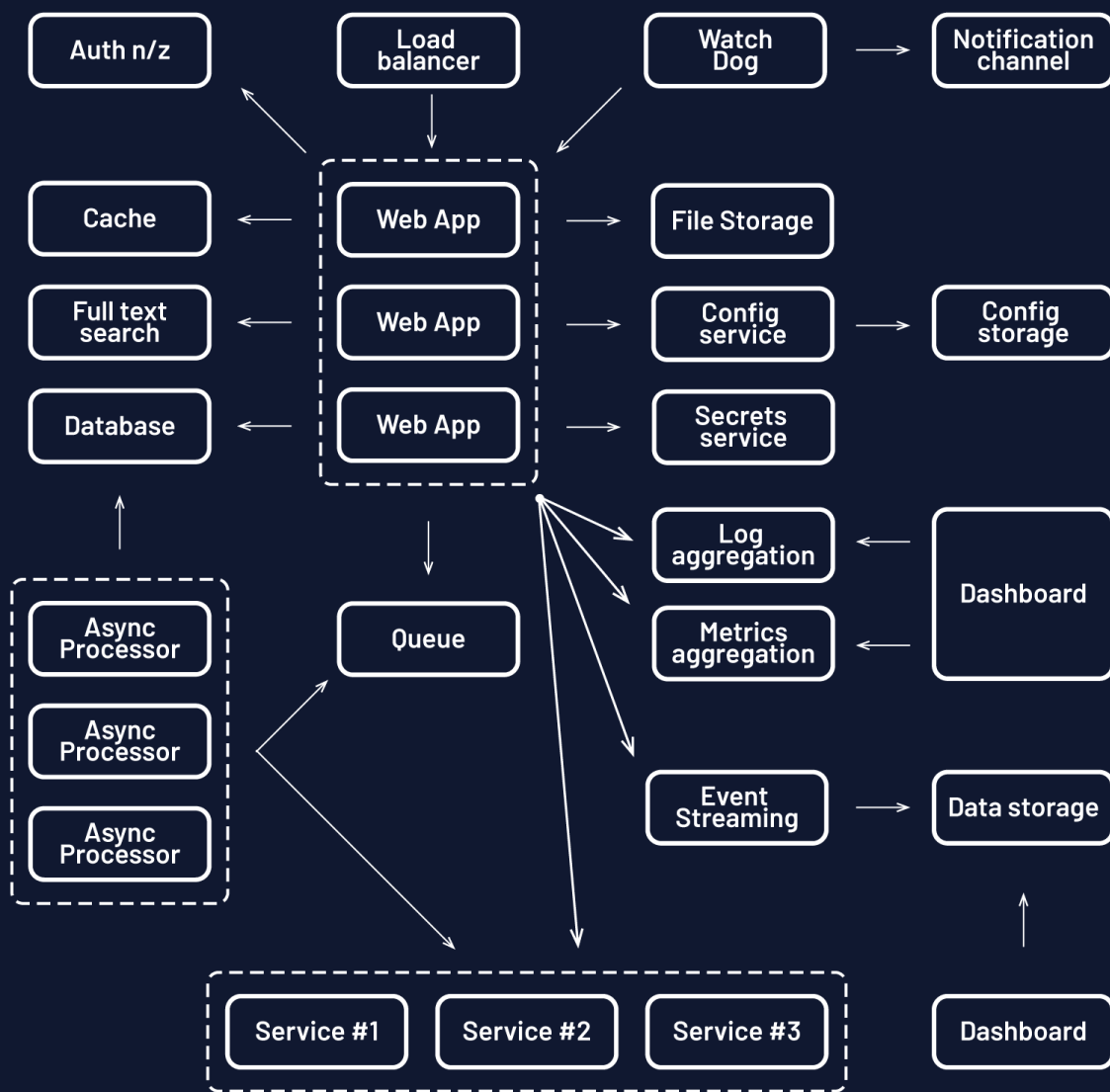
Pentru validarea datelor, sistemul va efectua o interogare Web Method către celelalte servicii, la rândul lor serviciile returnând rezultatul interogării care va fi analizat de sistem și afișat utilizatorului într-un limbaj clar și ușor de utilizat.

Serviciul REST API va integra Sistemul informational e-guvernare și serviciile orizontale, inclusiv:

- MCloud – cloud guvernamental pentru a găzdui Sistemul informational;
- MConnect – platformă de interoperabilitate guvernamentală care va fi utilizată pentru integrarea și schimbul de date între platformă și sistemele IT externe;
- MPass – serviciu de conectare unică de autentificare guvernamentală prin certificat digital sau identitate mobilă;
- MSign – serviciu guvernamental pentru aplicarea și validarea semnăturilor digitale și mobile;
- MNotify – serviciu guvernamental de notificare a utilizatorilor autorizați cu privire la apariția diferitelor evenimente de afaceri;
- MLog – serviciu guvernamental pentru înregistrarea evenimentelor;
- MPay – platformă guvernamentală de plată electronică;
- MPower – serviciu guvernamental de gestionare a autorizațiilor pentru persoane fizice și juridice;

2.2 Considerații de aspect fizic și tehnologic

Aspectul fizic/vizualizarea arhitecturii se va concentra pe hardware-ul și infrastructura reală a sistemului, mediile și implementarea diferitelor containere/componente. Mai jos puteți vedea un exemplu de arhitectură finală a sistemului:

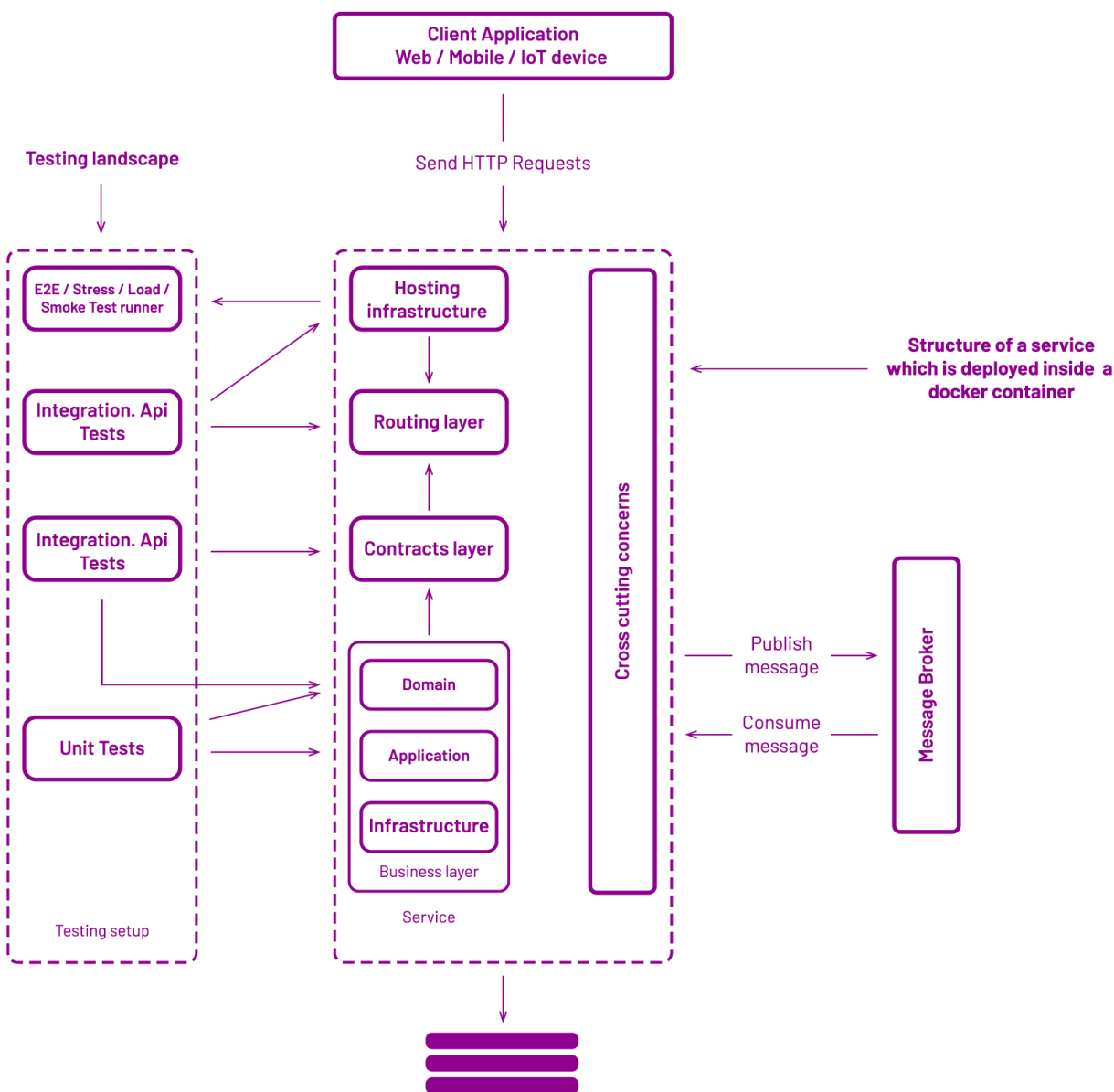


2.2.1 Tipuri de containere

2.2.1.1 Containere de aplicatii

În centrul sistemului, există mai multe containere de aplicații în care se află logica principală a aplicației. Aceste containere pot fi fie containere docker, servere, soluții PAAS care găzduiesc componenta etc.

Containerele specifice pot încapsula una sau mai multe componente în funcție de constrângeri. Vă rugăm să vedeți mai jos un exemplu de arhitectură de serviciu implementată în interiorul unui container Docker ce este consumat de către un client web standard:



2.2.1.2 Servicii auxiliare

Sistemul informational va include servicii auxiliare care acceptă aplicația de bază. Aceste servicii pot îndeplini sarcini precum autentificarea utilizatorului, procesarea plăților sau integrări externe.

Sistemul informational guvernamentală de interoperabilitate MConnect va facilita schimbul de date între autorități pentru a crește eficiența și calitatea furnizării serviciilor publice. Prin intermediul platformei de interoperabilitate, autoritățile publice fac schimb de date în timp real fără a le solicita cetățenilor și mediului de afaceri sub formă de certificate, rapoarte etc. A fi utilizat pentru interoperabilitatea cu sisteme terțe și integrarea cu serviciile M: MPass, MNotify, MLog, MSign etc.

2.2.1.3 Baze de date

Componentele vor interacționa cu una sau mai multe baze de date, de obicei un sistem de management al bazelor de date relaționale (RDBMS) sau alte depozite de date. Bazele de date stochează datele aplicației și pot fi localizate pe același server ca aplicația de serviciu sau pe servere de baze de date separate.

Sistemul de baze de date relaționale MSSQL va fi utilizat pentru stocarea datelor, cu accent pe extensibilitate și conformarea cu standardele. Baza de date a serverului va rula pe un container separat.

2.2.1.4 Operațiuni de fundal

Sistemele moderne necesită un mecanism pentru timpul de rulare sau pentru a calcula joburi consumatoare la cerere. Pentru a asigura ușurința în utilizare și instrumente de configurare precum Hangfire sau Quartz vor fi utilizate.

2.2.1.5 Căutare elastică

Elastic Search implementează experiențe de căutare puternice și moderne. Permite adăugarea căutării prestabilite pe site-ul web, aplicația sau magazinul de comerț electronic. Sistemul informational va implementa cazuri de utilizare de căutare prin Elastic Search. Serviciul permite indexarea documentelor și rularea interogărilor de căutare rapid și scalabil

2.2.1.6 Cache

Pentru a optimiza viteza de acces la date, va fi utilizat sistemul de cache Redis. Redis este un proiect de structură de date în memorie care implementează o bază de date distribuită, cu valori cheie în memorie și durabilitate opțională.

Utilizarea mecanismelor de cache va îmbunătăți performanța sistemelor, scalabilitatea, realitatea și experiența utilizatorului, oferind totodată și economii de costuri.

2.2.1.7 Cozi de mesaje

Comunicarea asincronă între componente va fi activată prin utilizarea instrumentelor de infrastructură, cum ar fi cozile de mesaje, cum ar fi RabbitMq sau Apache Kafka.

2.2.1.8 Servicii web pentru asimilarea datelor

În aplicațiile web, serverele web (de exemplu, Nginx, Apache) stau adesea în fața serverului de aplicații pentru a gestiona solicitările HTTP primite, pentru a servi active statice și pentru a oferi caracteristici de securitate precum terminarea SSL.

2.2.1.9 Opțional - Load balancer

În funcție de cerințele de trafic și de scalabilitate, echilibratorii de încărcare pot fi folosiți pentru a distribui cererile primite către mai multe instanțe ale serverului de aplicații. Acest lucru ajută la distribuția sarcinii și la toleranța la erori.

În scopul unei gestionări eficiente a traficului, se va folosi un auto scaling load-balancer, în cazul în care găzduirea o permite, care va distribui automat numărul de vizitatori la instanțele disponibile ale aplicației și va crește numărul acestora la nevoie.

2.2.1.10 Opțional - CDN

Rețelele de livrare a conținutului (CDN) pot fi folosite pentru a stoca în cache și pentru a difuza active statice, cum ar fi imagini, foi de stil și fișiere JavaScript, de la servere edge distribuite pentru a reduce latența.

2.2.1.11 Stocarea fișierelor

Pentru stocarea fișierelor, stocarea obiectelor sau alte necesități de stocare a datelor, serviciile de stocare în cloud sau on-premise pot fi integrate în arhitectură. În acest caz specific, MCloud va servi pentru serviciul de stocare.

MinIO va fi folosit pentru a stoca fișiere. MinIO este un server de stocare în cloud ca magazin de obiecte, MinIO poate stoca date nestructurate, cum ar fi fișiere, fotografii, videoclipuri, fișiere jurnal, copii de siguranță și imagini container.

2.2.1.12 Monitorizarea Sistemului

Instrumentele și serviciile pentru monitorizarea, înregistrarea în jurnal și analiza performanței sunt componente esențiale pentru a asigura sănătatea și stabilitatea sistemului.

SEQ este un back-end flexibil auto-găzduit pentru subsistemul de înregistrare ASP.NET Core (Microsoft.Extensions.Logging). Evenimentele de jurnal generate de cadrul și codul aplicației sunt trimise prin HTTP către un server Seq, unde datele structurate asociate fiecărui eveniment sunt utilizate pentru filtrare, corelare și analiză puternice.

Instrumentul Nagios va fi folosit pentru monitorizarea în timp real a infrastructurii, Nagios furnizează grafice și rapoarte, pe măsură ce utilizatorii au nevoie de ele, pe măsură ce datele sunt produse. Raportarea instrumentului oferă detectarea timpurie a întreruperilor, amenințărilor de securitate și erorilor.

2.2.1.13 Subsisteme de configurare

Configurația externă este o arhitectură în care valorile de configurare sunt păstrate într-un depozit central accesibil prin HTTP. Într-o arhitectură de microservicii, cloudul containerelor rulează mai multe instanțe. Este necesar să se asigure că configurațiile sunt consecvente în toate instanțele, precum și atunci când o valoare se modifică, toate instanțele își reîmprospătează configurația respectivă fără timp de nefuncționare.

2.2.2 Medii și rețele

Arhitectura generală a sistemului ar trebui să includă următoarele medii dedicate:

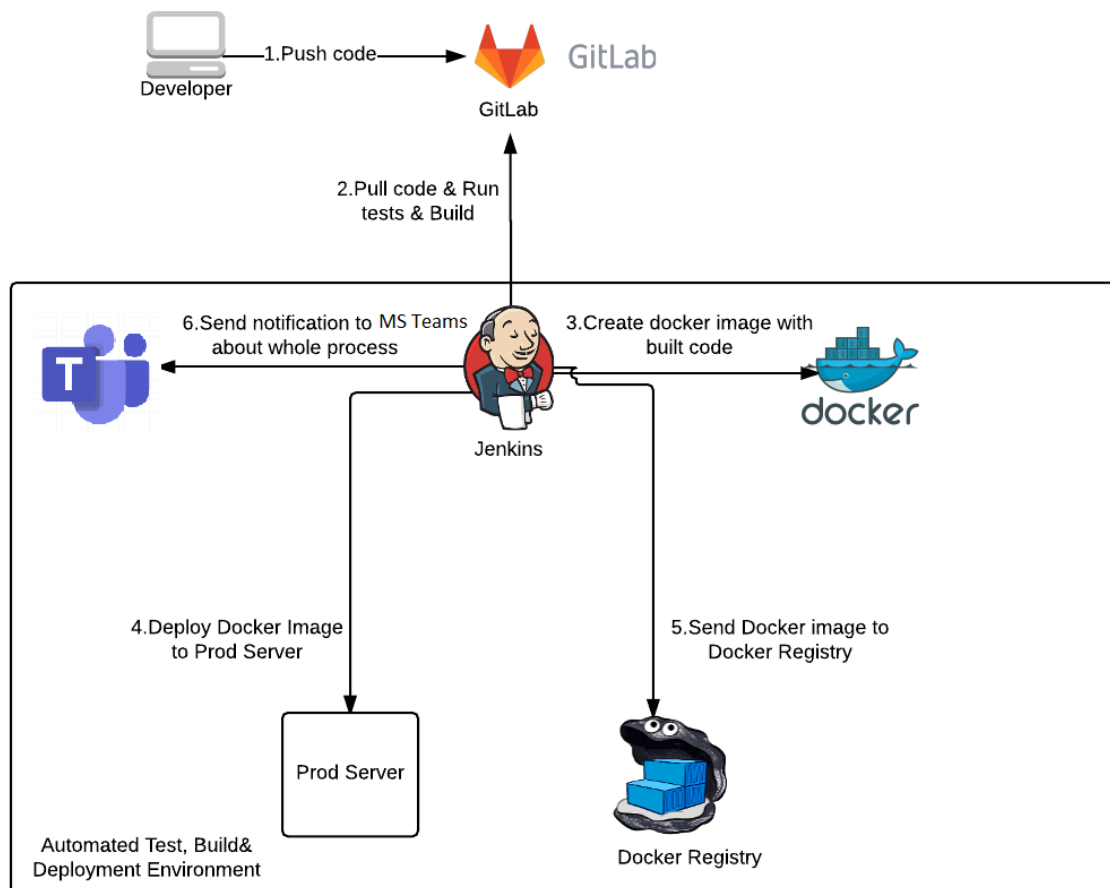
- Mediul de dezvoltare - reprezintă mediul în care echipa de dezvoltare va lucra și va putea susține testele efectuate de dezvoltatori.
- mediu de testare/instruire - reprezintă mediul în care sunt testate toate tipurile de modificări ale sistemului și/sau infrastructurii, anterior implementării lor în mediul de producție;
- Mediul de producție - reprezintă mediul primar al sistemului, responsabil de funcționarea sistemului Sistemul informational.

Sugestiile noastre cu privire la rețea:

- Clientul ar trebui să ne furnizeze un IP extern;
- Dacă cloud-ul/ Sistemul informational are firewall sau anumite restricții, atunci clientul ar trebui să ne furnizeze specificațiile necesare poate fi un ghid rapid pentru Sistemul informational lor;
- Furnizați-ne conexiunea la platformă (utilizator administrator la VM).

3 Flux DevOps (CI/CD)

Se vor folosi tehnologii pentru serverul de aplicații, după cum urmează:



4 Back-up și recuperare

Pentru fiecare componentă a sistemului (server web, bază de date) se fac copii de rezervă și pot fi reconstruite în caz de deteriorare. Soluția va fi tolerantă la erori. Arhitectura propusă oferă clustering și failover pentru întregul Sistem informational și componentele proprii. Tehnici care vor fi folosite:

- Replicarea sau oglindirea datelor;
- Copiere de rezervă a datelor.

Prevenirea distrugerii, modificării datelor cu caracter personal sau a disfuncționalităților în funcționarea software-ului destinat prelucrării datelor cu caracter personal este asigurată prin metoda utilizării mijloacelor speciale tehnice și de protecție a programelor, inclusiv programe licențiate, programe antivirus, organizarea sistemului de control al securității software și backup-uri regulate.

- Backup-ul în baza de date se va efectua zilnic prin sistemul intern al MSSQL, de asemenea aceste copii vor fi testate de sistemul intern MSSQL pentru integritate copile vor fi păstrate nu mai puțin de 7 zile, dar nu mai mult de 30 de zile;
- Backup-ul la fișierele statice se va efectua zilnic prin sistemul rsync, salvarea progresivă a copiilor noilor fișiere va fi păstrată nu mai puțin de 7 zile, dar nu mai mult de 30 de zile;
- Backup-ul pe întregul server va fi efectuat lunar prin copilele gazdei fizice vor fi păstrate timp de 6 luni.

PLAN DE IMPLEMENTARE

Descriere	Durata	Resursele alocate
Sistemul Informațional „Registrul Aerian al Republicii Moldova” (SI RA)	59.5 days	
Etapă 1: Inițierea Proiectului	2.5 days	
Analiza cerințelor din Caietul de sarcini, stabilirea obiectivelor și scopurilor proiectului, identificarea resurselor necesare.	0.5 days	Manager de proiect
Elaborarea planului de proiect detaliat, definirea graficului de implementare, inclusiv milestone-uri și termene limită.	1 day	Manager de proiect
Elaborarea planului de comunicare și gestionarea relațiilor cu toate părțile implicate în proiect	1 day	Manager de proiect
Etapă 2: Analiza și Designul Sistemului	17.5 days	
Analiza detaliată a cerințelor: Revizuirea cerințelor funcționale și nefuncționale, consultarea cu părțile interesate pentru clarificarea cerințelor.	5 days	Analist de business
Designul arhitecturii sistemului: Elaborarea designului arhitectural al sistemului SI RA, definirea componentelor principale și a interfețelor între acestea, designul bazelor de date și al fluxurilor de date.	5 days	Arhitect de sistem
Elaborarea specificațiilor tehnice: Documentarea specificațiilor tehnice detaliate, revizuirea și aprobarea specificațiilor de către părțile interesate.	5 days	Arhitect de sistem
Elaborarea și proiectarea Proiectul Tehnic (SRS+SDD)	10 days	Analist de business, Arhitect de sistem
Coordonarea și aprobarea sarcinii tehnice și a elementelor de design	2 days	Manager de proiect
Etapă 3: Dezvoltarea Sistemului	47 days	Analist de business [10%], Manager de proiect [10%]
Configurarea aplicației	3 days	
Instalare și configurare mașini virtuale: - Mediul de dezvoltare - Mediul de producție	2 days	DevOps
Instalare și configurare bază de date Microsoft SQL Server	0.5 days	DevOps
Instalarea componentelor software și configurarea proiectului	0.5 days	DevOps
Implementarea functionalului de autentificare și User Management	6 days	
Definirea rolurilor utilizatorilor și a drepturilor de acces	1 day	Dezvoltator software 1
Implementarea securității prin ID unic și parolă	1 day	Dezvoltator software 1
Crearea audit trail pentru acțiuni	1 day	Dezvoltator software 1

Implementarea functionalului de autentificare prin mPass	1 day	Dezvoltator software 1
Implementarea functionalului de autorizare bazat pe drepturi de acces	1 day	Dezvoltator software 1
Elaborarea profilul utilizatorului si interfetei de gestionare	1 day	Dezvoltator software 1
Implementarea Sistemelor Guvernamentale (MPass, MSign, MLog, MNotify)	4 days	
Implementarea MPass – pentru autentificarea utilizatorilor autorizati SI RA;	1 day	Dezvoltator software 1
Implementarea MSign – pentru semnarea formularelor electronice completate și documentelor/rapoartelor	1 day	Dezvoltator software 1
Implementarea MLog – pentru a înregistra evenimentele critice în timpul funcționării SI RA;	1 day	Dezvoltator software 1
Implementarea MNotify – pentru a notifica utilizatorii autorizati SI RA	1 day	Dezvoltator software 1
Implementarea Registrului Aerian	45 days	
a) Furnizarea informațiilor despre aeronave;	5 days	Dezvoltator software 2 ,Dezvoltator software 3
b) Imprimarea certificatelor de înmatriculare;	3 days	Dezvoltator software 2 ,Dezvoltator software 3
c) Imprimarea certificatelor de radiere;	3 days	Dezvoltator software 2 ,Dezvoltator software 3
d) Introducerea modelelor de aeronave	3 days	Dezvoltator software 2 ,Dezvoltator software 3
e) Introducerea tipurilor de aeronave	3 days	Dezvoltator software 2 ,Dezvoltator software 3
f) Introducerea producătorilor de aeronave,	2 days	Dezvoltator software 2 ,Dezvoltator software 3
g) Introducerea aeronavelor	2 days	Dezvoltator software 2 ,Dezvoltator software 3
h) Introducerea deținătorilor de aeronave	2 days	Dezvoltator software 2 ,Dezvoltator software 3
i) Introducerea proprietarilor de aeronave	2 days	Dezvoltator software 2 ,Dezvoltator software 3
j) Introducerea numerelor de înmatriculare	2 days	Dezvoltator software 2 ,Dezvoltator software 3
k) Introducerea înmatriculărilor	3 days	Dezvoltator software 2 ,Dezvoltator software 3
l) Introducerea înmatriculărilor	3 days	Dezvoltator software 2 ,Dezvoltator software 3
m) Vizualizarea istoricului înmatriculărilor	3 days	Dezvoltator software 2 ,Dezvoltator software 3
n) Vizualizarea înmatriculării la nivel de proprietar de aeronavă	4 days	Dezvoltator software 2 ,Dezvoltator software 3
o). Vizualizarea înmatriculării la nivel de deținător de aeronavă Audit/inspectii AAC	4 days	Dezvoltator software 2 ,Dezvoltator software 3
Rezultatele auditurilor și inspecțiilor de supraveghere efectuate de AAC	10 days	
a) Introducere manuală a inspecțiilor de audit	5 days	Dezvoltator software 1

b) Vizualizarea istoricului inspecțiilor	5 days	Dezvoltator software 1
Integrare cu Alte Sisteme	12 days	
Integrarea Registrul de stat al unităților de drept – care furnizează date oficiale de înregistrare a persoanelor juridice	12 days	Dezvoltator software 1
Funcționalități de Administrare și Configurare	15 days	
Definirea utilizatorilor și rolurilor	5 days	Dezvoltator software 1
Administrarea politicilor de securitate și configurarea restricțiilor de acces.	5 days	Dezvoltator software 1
Gestionarea permisiunilor la nivel de document și funcționalități .	5 days	Dezvoltator software 1
Audit și monitorizare:	20 days	
Înregistrarea tuturor acțiunilor utilizatorilor în jurnale electronice .	5 days	Dezvoltator software 4
Implementarea functionalului de cautare informatie bazat pe criterii	5 days	Dezvoltator software 4
Implementarea functionalului de stocarea logurilor catre Sistemul informational Mconect	5 days	Dezvoltator software 4
Implementarea functionalului de vizualizarea actiunilor utilizatorului	5 days	Dezvoltator software 4
Implementarea modului de Rapoarte	40 days	
Implementarea rapoartelor statice	20 days	Dezvoltator software 5
Implementarea rapoartelor dinamice	20 days	Dezvoltator software 5
Funcționalități de Back-up și Restaurare	15 days	
Sistem de back-up și stocare:Realizarea de copii de rezervă incrementale și depline.	7 days	Dezvoltator software 4
Restaurarea datelor în caz de incidente .	5 days	Dezvoltator software 4
Etapa 4: Testarea sistemului	18.5 days	
Elaborarea planului de testare și rezultatele testării interne (funcționale, de integrare, de performanță, de incarcare si securitate)	3 days	Analist de business
Efectuarea testarii functionale	3 days	Tester 1
Efectuarea testarii de integrare	3 days	Tester 1
Efectuarea testarii de performanta	2 days	Tester 1
Efectuarea testarii de incarcare	2 days	Tester 2
Efectuarea testarii de securitate	3 days	Tester 2
Realizarea testării de acceptanță de către utilizator (UAT)	3 days	Tester 2
Elaborarea și prezentarea raportului de testare	0.5 days	Tester 2
Efectuarea sedintei de prezentare a sistemului catre beneficiar	0.5 days	Manager de proiect
Etapa 5: Documentare și instruire	9.5 days	
Descrierea cursurilor și a rezultatelor așteptate	1 day	Trainer
Documentul privind configurarea si implementarea Sistemului instructiunea pentru deployment)	1 day	Trainer
Elaborarea raportului de instalare și configurare	1 day	DevOps
Elaborarea manualelor de utilizare	1 day	Tester 1

Elaborarea manualului Administratorului (inclusiv planul de contingență)	1 day	DevOps
Procese verbale de recepție cantitativă și calitativă	1 day	Manager de proiect
Instruirea directă a administratorilor sistemului	1 day	DevOps
Instruirea directă a utilizatorilor	1 day	Manager de proiect
Pregătirea materialelor electronice tip eLearning pentru utilizatori	1 day	Trainer
Etapa 6: Lansare în exploatare industrială	2.5 days	
Configurarea sistemului, pregătirea mediului de producție și realizarea testelor finale	0.5 days	DevOps
Prezentare documentație tehnică a sistemului către personalul administrativ	0.5 days	DevOps
Lansarea oficială a sistemului	0.5 days	DevOps
Metenanta	240 days	

Metodologia, Abordarea

1. Această secțiune ar trebui să demonstreze receptivitatea ofertantului la TOR prin identificarea componentelor specifice propuse, abordând cerințele, oferind o descriere detaliată a caracteristicilor esențiale de performanță propuse și demonstrând modul în care abordarea și metodologia propuse îndeplinesc sau depășesc cerințele. Toate aspectele importante ar trebui abordate cu suficient detaliu, iar diferitele componente ale proiectului ar trebui să fie ponderate adecvat unele în raport cu altele. O descriere detaliată a abordării și metodologiei pentru modul în care Ofertantul va îndeplini Termenii de referință ai proiectului, ținând cont de adecvarea la condițiile locale și mediul proiectului. Detalii despre modul în care diferitele elemente ale serviciului vor fi organizate, controlate și furnizate.
2. Metodologia va include, de asemenea, detalii cu privire la mecanismele interne de evaluare tehnică și de asigurare a calității ale Ofertantului.
3. Explicați dacă vreo lucrare ar fi subcontractată, cui, cât procent din lucrare, motivul pentru aceasta și rolurile subcontractanților propuși și modul în care toată lumea va funcționa ca o echipă.
4. Descrierea mecanismelor și instrumentelor de monitorizare și evaluare a performanței disponibile; cum vor fi adoptate și utilizate pentru o cerință specifică.
5. Planul de implementare care include o diagramă Gantt sau un program de proiect care indică secvența detaliată a activităților care vor fi întreprinse și calendarul lor corespunzător.
6. Demonstrați modul în care intenționați să integrați măsurile de sustenabilitate în execuția contractului.
7. Orice alte comentarii sau informații referitoare la abordarea și metodologia proiectului care vor fi adoptate.

Definiii

Sprintul – o perioadă în care echipa Scrum lucrează la sarcini specifice. Echipele noastre Agile estimează misiunile pentru 22 de zile lucrătoare. Fiecare sprint încorporează faze tradiționale de dezvoltare a software-ului: cerințe, evaluări, proiectare, evoluție, livrare. Echipa Agile își concentrează eforturile pe generarea unei noi unități funcționale pentru produs la sfârșitul sprintului. Beneficiul cheie al împărțirii proiectelor în sprinturi este faptul că sistemul se adaptează mai bine la schimbare. Sprinturile au următoarea structură:

- Planificarea sprintului,
- Întâlniri zilnice (Scrums),
- Dezvoltarea produsului,
- Sprint Review,
- Retrospectivă.Proiect

Project/Product Backlog – o listă de sarcini pe care echipa Agile intenționează să le îndeplinească pentru a atinge anumite obiective. Backlogul include descrieri ale funcționalităților și serviciilor dorite necesare pentru atingerea obiectivului final predefinit. Sprint Backlog este flexibil pentru a se modifica atâta timp cât modificările

nu interferează cu Scopul Sprintului. Toate sarcinile au diverse priorități atribuite de Product Owner, în funcție de valoarea lor de afaceri (care este stabilită de Product Owner) și de efortul necesar pentru a le dezvolta (stabilit de Echipa de Dezvoltare cu Scrum Master/Team Leader).). Backlog-ul este actualizat regulat prin adăugarea sau modificarea articolelor, realizarea de specificații, eliminarea și stabilirea priorităților cu privire la elementele incluse. Unele dintre elementele care pot face parte din produsul nerezolvat sunt implementarea anumitor funcții, remedierea erorilor, rezolvarea problemelor, îmbunătățirea diferitelor componente. Printre cei care pot participa la procesul de construire a acestui produs se numără clienții, Echipa de Dezvoltare, echipa de marketing și de vânzări. Product Owner-ul este responsabil de gestionarea Product Backlog-ului.

Sprint Backlog – este un plan detaliat bazat pe elementele produsului nerezolvat, care va fi abordat pe parcursul următorului sprint; conține informații cu privire la modul în care Echipa va implementa cerințele stabilite? Sarcinile sunt împărțite în ore și fiecare sarcină nu ar trebui să dureze mai mult de 8 ore (dacă o sarcină ar putea dura mai mult, aceasta trebuie împărțită în sarcini mai mici). Sarcinile nu sunt atribuite angajaților – aceștia sunt liberi să aleagă sarcinile pe care le doresc. În timpul întâlnirii de planificare a sprintului, Scrum Master, Product Owner și Echipa decid împreună care probleme nerezolvate vor face parte din următorul sprint pe baza priorităților și obiectivelor stabilite pentru acel sprint. Restul de sprint este fix, ceea ce asigură stabilitatea proiectului. De îndată ce toate sarcinile sprintului curent sunt finalizate, o nouă iterație a sistemului este finalizată și poate fi rezumată într-o notă de lansare.

Sprint Burndown Chart – este un grafic care afișează munca rămasă de făcut până când produsul este terminat în funcție de timp. Este actualizat zilnic și ajută la estimarea datei de lansare când produsul va fi gata.

Estimarea efortului – este un proces iterativ care implică concentrarea atenției pe estimarea, cât mai exact posibil, a efortului depus pentru a gestiona o sarcină nerezolvată.

Sprint planning meeting – este o întâlnire împărțită în două părți organizată de Scrum Master. Prima parte este dedicată viziunii de sprint și backlog-ului de sprint pe care PO le prezintă echipei. Pe parcursul acestei părți sunt definite obiectivele și funcționalitățile următorului sprint. În a doua parte a întâlnirii, Scrum Master și Echipa discută despre modul în care va fi implementată unitatea de produs reprezentată de acest sprint.

Întâlnire zilnică Scrum – are loc zilnic pentru a urmări continuu progresul echipei. Între timp, servește și ca o întâlnire de planificare, deoarece echipa discută o serie de subiecte: acțiunile întreprinse de la ultima întâlnire până în prezent, acțiunile care trebuie întreprinse până la următoarea întâlnire și problemele sau impedimentele care i-au împiedicat pe membrii echipei să atingerea obiectivelor stabilite. Scrum Master/Team Leader conduce întâlnirea, care durează aproximativ 15 minute. Fiecare întâlnire urmează setul de reguli de mai jos:

- Este necesar ca ședința să înceapă la timp și să aibă prezență deplină;
 - Oricine poate fi prezent la întâlnire, dar participă doar cei implicați în procesul de dezvoltare;
 - Întâlnirea durează 15 minute, indiferent de numărul de persoane care participă.
 - O singură persoană vorbește odată, nimeni nu are voie să se întrerupă.
 - Aceste întâlniri ar trebui să aibă loc în fiecare zi, în același loc și la aceeași oră.
- Sprint review and retrospective meeting** – o întâlnire care marchează sfârșitul sprintului și oferă echipei ocazia de a reflecta asupra a ceea ce s-a făcut. Ca parte a întâlnirii, Scrum Master prezintă rezultatul sprintului și încurajează echipa să îmbunătățească procesul de lucru. Cei care participă la întâlnire au ocazia să reflecteze asupra a ceea ce s-a realizat și să ia o decizie cu privire la următorii pași. Revizuirea Sprintului și întâlnirea retrospectivă ar trebui să acopere următoarele:

- Obiecte care au mers bine în timpul sprintului;
- Reflecții asupra dificultăților, problemelor și nemulțumirilor întâmpinate;
- Lucruri pe care echipa le-a învățat în timpul sprintului;
- Ce ar putea fi îmbunătățit în următorul sprint?

Introducere în Managementul Agil de Proiect – SCRUM

Furnizorul are o vastă experiență în implementarea unor proiecte similare. Această experiență se reflectă în fluxurile și metodologiile de implementare, dar și în posibilitatea de a oferi o echipă dedicată care să poată îndeplini toate obiectivele proiectului.

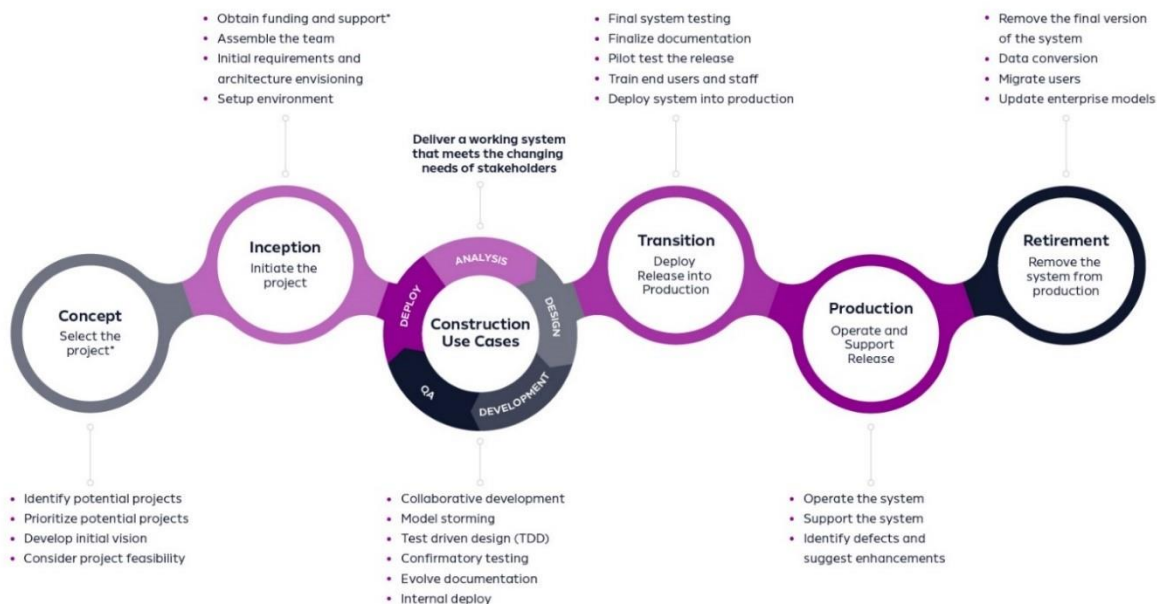
Acest capitol prezintă metodologia și procedurile care vor fi utilizate în timpul implementării proiectului, structura de implementare a acestuia, precum și livrabilele asociate.

Abordarea de dezvoltare/implementare Agile este o metodologie flexibilă de implementare a proiectelor IT care permite satisfacerea nevoilor de afaceri ale Beneficiarului folosind o abordare iterativă. Cadrul, pe care se bazează, susține mai multe tipuri de dezvoltare potrivite ambelor scenarii: atunci când Beneficiarul dorește o nouă soluție IT și când necesită îmbunătățirea uneia existente.

Serviciile noastre răspund nevoilor individuale ale Beneficiarului și ale utilizatorului final. Prioritatea noastră este să satisfacem nevoile Beneficiarului și să livrăm soluții la timp. Software-ul care funcționează este livrat în iterații scurte și în conformitate cu restul cu prioritate ridicată.

Metodologia pe care o aplicăm este adaptată proiectelor agile cu preț fix, precum și contractelor de timp și

Agile Methodology



* Phases and steps not required if the projects are bids and RFPs or the customer already provides a clear project brief.

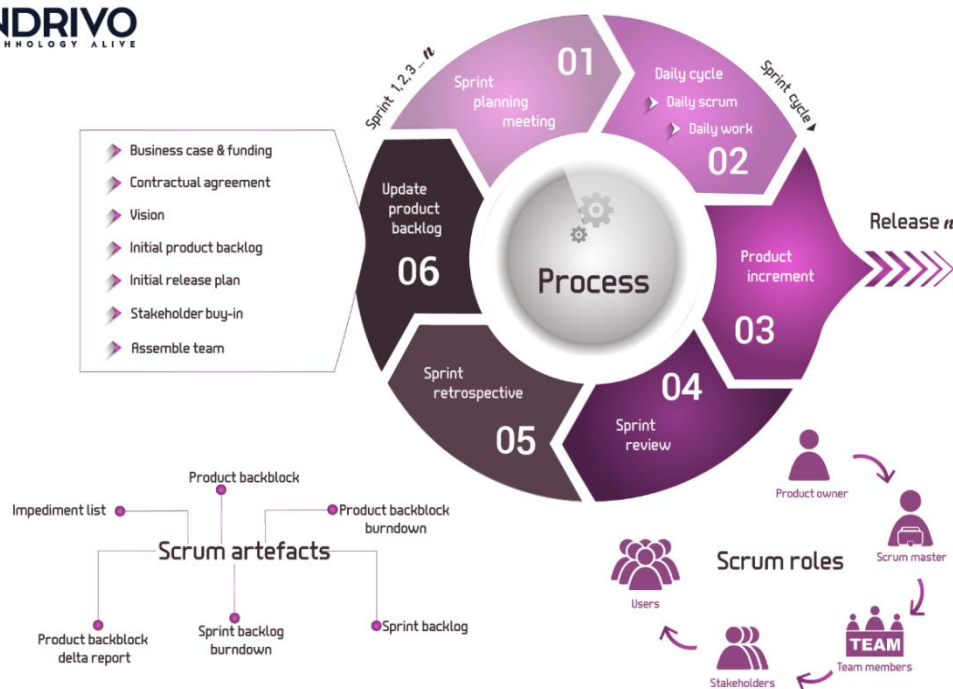
materiale (T&M).

Scrum Overview

Metodologia de lucru pe care o aplicăm (Scrum) se bazează pe abordarea de dezvoltare incrementală a dezvoltării software, păstrând totodată o listă cu tot ceea ce trebuie făcut în produs (Backlog). Prin utilizarea metodologiei de dezvoltare software agilă, riscurile de dezvoltare sunt reduse, împreună cu timpul de execuție. Drept urmare, proiectele sunt implementate într-o manieră extrem de flexibilă, iar calitatea este îmbunătățită după fiecare sprint.

Scrum este mai degrabă „un cadru în care poți folosi diverse procese și tehnici”, decât un proces sau o tehnică de construire a produselor. Cadrul Scrum se bazează pe echipele Scrum, inclusiv pe roluri asociate, evenimente, artefacte și reguli.

La sfârșitul fiecărui sprint, există un livrabil tangibil pentru afacere. Acest proces este descris în diagrama



următoare:

Cerințele care stau la baza proiectului formează un Project Backlog care este actualizat în mod regulat. Caracteristicile care sunt asociate cu aceste cerințe sunt numite Povești de utilizator/Cazuri de utilizare (conform Termenilor de referință pentru fiecare sarcină specifică).

Lucrarea este împărțită într-o serie de 1 până la 4 săptămâni, în care echipa de afaceri și de proiect estimează care cazuri de utilizare, în ordinea descrescătoare a priorității, sunt realizabile în fiecare ciclu sau iterație. Acest subset de cazuri de utilizare din Backlog-ul de proiect formează baza pentru Backlog-ul de iterații planificat pentru livrare în acea perioadă de 1 până la 4 săptămâni. Conform cadrului Scrum, în timpul unei iterații au loc 3 întâlniri în intervale de timp (sau cu durată fixă), pe lângă o întâlnire zilnică stand-up cu echipa de dezvoltare, Scrum Master și Product Owner. La începutul unui sprint, caracteristicile care trebuie dezvoltate în timpul sprintului sunt decise în timpul întâlnirii de planificare a sprintului. De îndată ce iterația s-a încheiat, echipa de revizuire și întâlnire retrospectivă are loc pentru a revizui produsul și a demonstra utilizarea software-ului, precum și pentru a îmbunătăți și reflecta asupra procesului de iterație în sine. După finalizarea sprintului, următorul set de cazuri de utilizare este selectat din Backlog-ul proiectului și procesul începe din nou.

Construirea echipei

Pentru fiecare sarcină specifică, echipa de proiect este un grup de experți auto-organizat care oferă valoare incrementală. Ca rezultat, echipa necesită o reprezentare interfuncțională a abilităților și cunoștințelor în domenii precum date, instrumente și domenii de infrastructură. Pentru proiectele guvernamentale/ONG-uri, echipa este de obicei reunită în timpul procesului de pre-vânzare, precum și experții cheie care vor lua parte la proiect. Scrum definește 3 roluri principale:

Product Owner – reprezintă vocea și interesele Beneficiarului în echipa Scrum. Acest profesionist este responsabil pentru proiectarea, managementul, controlul și prezentarea Backlog-ului de proiect/produs. Totodată, Product Owner are apelul final privind sarcinile legate de produsul neterminat și atribuie priorități.

Scrum Master – este servitorul-lider pentru Echipa de Dezvoltare și reprezintă Furnizorul. Scrum Masters acționează ca manageri de proiect care se asigură că procesul de dezvoltare progresează conform abordării Scrum, valorilor și regulilor acesteia. Acest rol presupune interacțiune atât cu Echipa de Dezvoltare, cât și cu Product Owner și Stakeholders. Scrum Master este, de asemenea, responsabil de înlăturarea oricăror impedimente pentru ca echipa de dezvoltare să-și crească productivitatea.

Echipa de dezvoltare – este un grup de oameni care se ocupă de toate lucrările legate de dezvoltarea produsului. Această echipă are autoritatea de a decide ce măsuri trebuie luate pentru a rezolva sarcinile atribuite

fiecărui sprint și are dreptul de a se autoorganiza în acest scop. Echipa de dezvoltare va fi formată din următorul expert:

- Dezvoltator baze de date;
- Specialist în integrare/Dezvoltator software;
- Specialist DevOps/Dezvoltator de software;
- Tester de software;
- Antrenor.

Unul dintre cele mai importante aspecte ale abordării agile este faptul că utilizatorii, clienții și tot felul de părți interesate sunt implicați în procesul de dezvoltare. Ei oferă feedback cu privire la rezultatele fiecărui sprint pentru a ajusta și îmbunătăți procesul de flux de lucru și eficiența în viitor. Nu doar echipa de dezvoltare joacă un rol important în procesul de dezvoltare a produsului, și următoarele tipuri de părți interesate joacă un rol important:

- **Utilizatori** – un grup de persoane care vor folosi produsul final în interiorul sau în afara organizației.
- **Clienți** – oameni care au achiziționat produsul și au angajat echipa de dezvoltare. Ele definesc obiectivul proiectului.
- **Sponsori** – persoane sau organizații care oferă sprijin financiar proiectului. De asemenea, participă la stabilirea obiectivelor și condițiilor de lucru. Este partea interesată față de care toată lumea este responsabilă atunci când proiectul este finalizat.

Echipa de **proiect** are următoarele caracteristici cheie:

- Cross-funcțional prin natura sa, unde membrii au seturi diferite de abilități;
- Auto-împunător;
- Responsabil cu livrarea produsului;
- Foarte angajat să ofere o creștere de top.

Echipele de proiect au următoarele responsabilități:

- Identificarea sarcinilor necesare pentru livrarea fiecărei caracteristici;
- Estimarea efortului pentru fiecare sarcină;
- Dezvoltarea caracteristicilor;
- Rezolvarea problemelor;
- Asigurarea calitatii;
- Pregătirea notelor de lansare.

Iterație zero

În timpul **Iteration Zero**, echipa explorează ideile de produse, nevoile clienților, practicile de dezvoltare, arhitectura hardware și software. Echipa își prezintă abordarea cu privire la caracteristicile necesare și munca care trebuie făcută. Aceștia ajung la o înțelegere comună a obiectivelor de dezvoltare, inclusiv nevoile pieței, nevoile de afaceri, conținutul produsului și cantitatea de efort necesară pentru a dezvolta produsul. Echipa cu asistența Product Ownerului creează un plan inițial.

Scopul **Iteration Zero** este de a obține o înțelegere clară a:

- Product Backlog și prototipuri;
- Arhitectura Tehnică;
- Relația dintre timp și bani.

Ca rezultat al **Iterației Zero**, echipa ar trebui să aibă următoarele:

- Structura proiectului;
- Design minimal;
- Un număr minim de povești finalizate.

Instrumente și instruire

Iterația Zero este o modalitate de a se asigura că echipa are toate instrumentele și informațiile necesare, astfel încât în Iterația unu să se poată concentra pe construirea unei felii de produs. Unele dintre activitățile care vor fi desfășurate de către Furnizor sunt următoarele:

- Configurarea mediilor de testare și producție;
- Crearea unui depozit de coduri, sistem de urmărire a problemelor, mediu CI/CD, sistem de management al sarcinilor;
- Configurarea instrumentelor pentru testarea acceptării utilizatorilor și testarea automatizării;
- Configurarea convențiilor pentru rapoarte și instrumente.

Inceput. Iterație de încălzire

Această iterație implică definirea sistemului care urmează să fie construit. Proprietarul produsului va crea un **stoc de proiect/produs** sau **specificații pentru cerințele software (SRS)**. Acesta va conține cerințele care sunt deja cunoscute sau sunt descrise în Termenii de referință pentru fiecare misiune specifică. Se acordă prioritate fiecărei cerințe, iar echipa estimează efortul necesar implementării fiecăreia dintre ele. În acest moment, echipa de proiect este în funcțiune și are acces la toate instrumentele și resursele necesare.

Prevederea cerințelor inițiale

Faza de evaluare presupune stabilirea cerințelor funcționale ale sistemului, începând cu evaluarea nevoilor care este legată de activitatea Beneficiarului și coroborarea acestor informații cu diferite tipuri de date disponibile: infrastructură tehnică și de securitate, utilizatori ai sistemului, cazuri de utilizare, fluxuri de lucru și procese, Reglementările legale și interne ale Beneficiarului, procedurile specifice de lucru și standardele internaționale aplicabile în domeniu. De asemenea, prevederea cerințelor va lua în considerare Termenii de referință pentru fiecare sarcină specifică.

Evaluarea se realizează la sediul Beneficiarului pentru a identifica nevoi specifice, procese de lucru și reglementări interne. La sediul Furnizorului se integrează toate datele colectate, se întocmește documentul de evaluare, iar componentele tehnice și detaliile de implementare sunt analizate.

Rezultatele fazei de evaluare reprezintă un set de specificații funcționale care sunt aprobate de către Product Owner și alte părți interesate.

Evaluarea cuprinde:

- Evaluarea situației actuale a Beneficiarului;
- Stabilirea obiectivelor generale ale proiectului;
- Stabilirea obiectivelor specifice prin rafinarea obiectivelor generale;
- Identificarea părților implicate (entități organizaționale din cadrul organizației Beneficiarului implicate în utilizarea și administrarea produsului ce urmează a fi livrat);
- Stabilirea cerințelor funcționale pentru toate părțile implicate;
- Crearea arhitecturii generale, inclusiv a modulelor/componentelor necesare.

În acest moment, se întocmește raportul de evaluare, care va include Product/Project Backlog furnizat de Product Owner.

Product/Project Backlog va conține toate cazurile de utilizare (funcții) ale produsului. Proprietarul produsului este responsabil pentru definirea cazurilor de utilizare și acordarea unei priorități fiecăreia dintre ele pentru livrare. Ordinea cazurilor de utilizare poate fi, de asemenea, influențată de dependențele cazurilor de utilizare sau de valoarea afacerii.

Cazurile de utilizare vor descrie în detaliu setul de caracteristici care se potrivesc fiecărei definiții de cerințe. Cazurile de utilizare cu prioritate ridicată care sunt programate pentru următorul Sprint necesită o descriere detaliată, iar echipa ar trebui să poată oferi o soluție în termen de 4 săptămâni.

Proprietarul produsului se va asigura că fiecare caz de utilizare va îndeplini următoarele caracteristici:

- Obiectivul cazului de utilizare
- Actor primar – oameni care au acces la el
- Nivel - este necesară selectarea rezumatului pentru un caz de utilizare care rezumă mai multe activități sau se află în afara domeniului de aplicare al sistemului; la nivel de utilizator pentru un caz de utilizare care descrie o activitate completă în sistem; sau subfuncție pentru un caz de utilizare care depinde de un caz de utilizare la nivel de utilizator, dar este prea lung pentru a fi inclus în cazul de utilizare la nivel de utilizator.
- Scurtă descriere/Obiectiv descrie funcția pe care actorul principal dorește să o îndeplinească.
- Precondițiile specifică ceea ce sistemul se va asigura că este adevărat înainte de a lăsa cazul de utilizare să înceapă și indică faptul că un alt caz de utilizare a fost rulat pentru a-l configura.
- Declanșatoare - evenimentele care determină inițierea cazului de utilizare.
- Flux principal - cazul de utilizare în care nimic nu merge prost (norma intenționată). Scenariul constă dintr-o succesiune de pași. Acestea sunt interacțiunile necesare între actori și soluția pentru atingerea scopului dorit.
- Flux alternativ (numit și cazuri de utilizare Excepții, Zi ploioasă sau Edge) - variații față de fluxul principal.
- Condiții post - condiții care trebuie îndeplinite înainte de a excita cazul de utilizare.
- Cerințele non-funcționale includ uzabilitate, fiabilitate, securitate, flexibilitate.

- Criterii de acceptare - set de declarații, fiecare cu un rezultat clar de trecere/eșec, care specifică atât cerințe funcționale (de exemplu, funcționalitate minimă comercializabilă) cât și nefuncționale (de exemplu, calitate minimă) aplicabile în etapa curentă a integrării proiectului.

Este important de menționat că pentru a crește acuratețea oricărei estimări, efortul implicat crește exponențial. Ținând Sprinturile sub control, putem revizui mai bine și putem răspunde rapid la abaterile estimate care ar putea apărea în proces.

Arhitectură inițială/Proiectare la nivel înalt

Arhitectură/Design la nivel înalt – aici are loc proiectarea la nivel înalt al sistemului; dacă sistemul există deja, se discută modificările necesare implementării cerințelor, precum și problemele asociate acestor acțiuni.

Echipa de dezvoltare va discuta și apoi va schița o arhitectură potențială pentru sistem. Această arhitectură va evolua, în această etapă, nu va fi foarte detaliată (trebuie doar să fie suficient de bună deocamdată). Având în vedere că scopul este identificarea unei strategii arhitecturale, scrierea mormanelor de documentație este redundantă.

Planul de testare funcțională și planul de acceptare sunt anexe la raportul de evaluare. Pe baza acestora, li-vrabilele proiectului vor fi verificate, validate și acceptate.

Iterații de construcție (sprinturi)

Misiunile sunt organizate în sprinturi, iar ședințele de planificare a sprintului definesc sprintul. Pe parcursul acestei întâlniri, Product Owner-ul informează Echipa despre sarcinile nerezolvate pe care dorește să le abordeze. În consecință, Echipa stabilește câte dintre aceste sarcini pot fi îndeplinite până la următorul Sprint. Misiunile selectate nu pot fi modificate în timpul sprintului. La sfârșitul sprintului, Echipa prezintă caracteristicile dezvoltate și modul în care poate fi utilizat produsul intermediar.

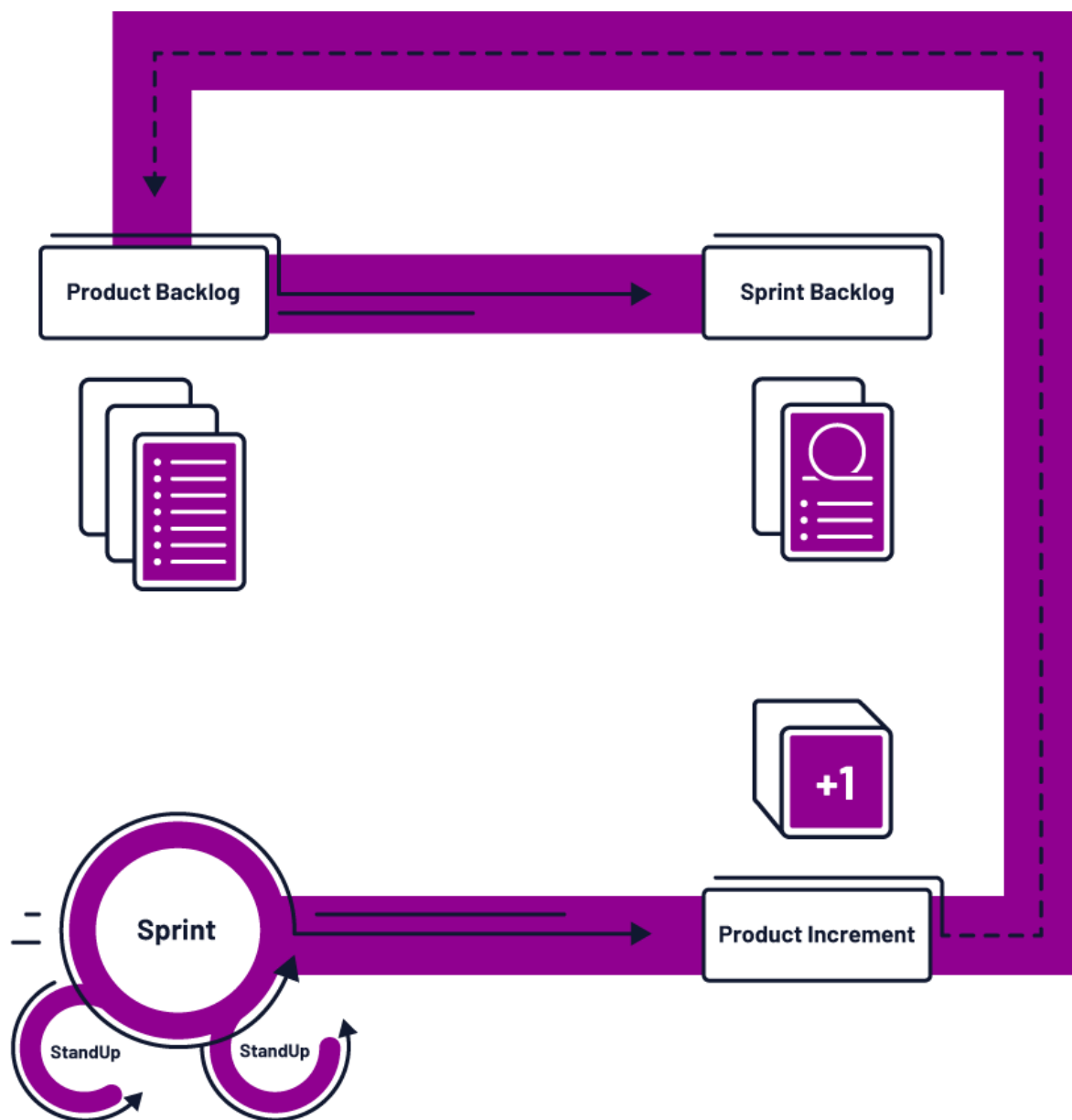
Prin utilizarea iterațiilor de construcție (sprinturi), echipele beneficiază de independență și autoorganizare, în timp ce comunicarea verbală/scriasă este transparentă între toți membrii echipei și diverse departamente legate de proiect.

Este general acceptat că pe parcursul dezvoltării proiectului Product Owner-ul se poate răzgândi cu privire la așteptările de la produsul software. Astfel de schimbări sunt imprevizibile și nu sunt ușor de adaptat la proiect prin metode tradiționale de dezvoltare a software-ului.

Întâlniri de planificare Sprint

Întâlnirea **de planificare a Sprintului** se desfășoară înainte de începerea fiecărui Sprint, permite Clientului și echipei de dezvoltare să discute cerințele și munca necesară pentru următoarea versiune. Acest pas din procesul Scrum se concentrează pe determinarea domeniului țintă de livrare pentru un Sprint și definește Backlogul Sprintului.

Întâlnirea de planificare a sprintului nu trebuie să dureze mai mult de 6 ore (pentru sprinturi de 4 săptămâni). Product Owner și Scrum Master/Team Leader sunt responsabili pentru actualizarea Backlog-ului de proiect ca pregătire pentru întâlnirea de planificare Sprint. Aceasta include clarificarea, prioritizarea și, în unele cazuri, investigarea fezabilității cazurilor de utilizare interdependente. Această activitate trebuie, de asemenea, să ia în considerare orice datorie tehnică moștenită de la Sprinturile anterioare.



1. Specificații de afaceri

Prima parte a întâlnirii de planificare a sprintului își propune să transforme caracteristicile din backlog într-un obiectiv realist pentru acest sprint. Proprietarul produsului face parte din acest proces și stabilește prioritatea sarcinilor. Acest lucru oferă, de asemenea, Product Owner-ului oportunitatea de a comunica domeniul necesar de livrare, de a furniza contextul și prioritatea de afaceri și de a răspunde oricăror întrebări pe care le poate avea echipa de proiect; prin urmare, Product Owner ajută la efectuarea pașilor de descompunere funcțională și estimare.

Această parte a întâlnirii nu trebuie să dureze mai mult de $\frac{1}{4}$ din timp. Asigurați-vă că a fost distribuită o copie a Backlog-ului de produs/proiect înainte de întâlnirea de planificare Sprint pentru a oferi echipei de proiect timp să

ia în considerare posibilele soluții care vor fi discutate în timpul atelierului și pentru a pregăti întrebări de clarificare pentru proprietarul produsului. Această sesiune va avea următorii participanți:

- Proprietar de produs;
- Scrum Master;
- Echipa de dezvoltare;
- Tester de software.

2. Specificații tehnice și estimarea sarcinilor

A doua parte a întâlnirii Sprint Planning este tehnică și are loc de obicei fără Product Owner. Acesta este pasul de descompunere și estimare a soluției în procesul de planificare care urmărește estimarea efortului pentru toate caracteristicile din Descrierea cazurilor de lansare și de testare.

Fluxul general de activitate în această etapă este descris după cum urmează:

1. Selectați cazul de utilizare (determinat de proprietarul produsului);
2. Creați o diagramă UML detaliată sau diagrame de secvență;
3. Creați un model de date logic;
4. Determinați sarcini și subsarcini pentru a livra cazul de utilizare;
5. Aproba Dependențe tehnice și Comandă;
6. Estimați efortul necesar pentru îndeplinirea fiecărei sarcini – o sarcină estimată nu va prevala mai mult de 8 ore;
7. Evaluarea riscurilor cu estimări de sarcini;
8. Atribuiți fiecare sarcină unui membru specific al echipei de dezvoltare.

Sarcinile mari ar trebui împărțite în subsarcini mai mici, de preferință nu mai lungi de 8 ore, în timp ce sarcinile care implică așteptare ar trebui împărțite în subsarcini separate. Sarcinile de cercetare ar trebui să aibă o estimare ridicată a incertitudinii.

Pentru cazurile de utilizare complexe sau cele cu un număr mare de interdependențe, poate fi necesar să dureze 2 zile pentru descompunerea și estimarea activităților, permițând în același timp membrilor echipei să se consulte cu părțile externe cu privire la fezabilitate și să primească informații în procesul de estimare.

3. Cazuri de utilizare/Structură de sarcini

Cazurile de utilizare/sarcinile pentru instrumentul de urmărire a problemelor vor avea următoarea structură predefinită:

- Titlu cu o scurtă descriere – Ex: [DevBE];[DevFE];[DevOps];[QA/Tester];
- Descrierea sarcinii cu următoarele componente: Diagrama fluxului/Diagrama secvenței/Modelul de date logice;
- Artefacte necesare pentru executarea sarcinii;
- Criterii de acceptare;
- Tip de activitate: Caracteristică, Bug, Epic etc.;
- Timp estimat;
- Termen limită (Prioritate în planificare): Data;
- Prioritate: Scăzut, Mediu, Ridicat, Critic;
- Sprint aferent;
- Deputat: Membru al echipei;
- Sarcină/Subsarcină conexe/interdependentă;
- Stare: Neprocesat, Nou, În curs, Dezvoltat, Testat, Lansat, Terminat.

Întâlniri Scrum (Stand-up-uri zilnice)

Scopul întâlnirii zilnice stand-up este de a informa Product Owner cu privire la progresul incremental și livrările ca parte a iterației. Se dorește să fie informativ și interactiv și să alinieze înțelegerea echipei despre ceea ce se lucrează, de către cine și statutul acesteia. Proprietarul de produs, Scrum Master/Team Leader și echipa de proiect participă la întâlnirea Scrum și este stabilită în timp pentru maximum 15 minute. Toți participanții trebuie să răspundă la următoarele trei întrebări:

- Ce ai realizat ieri?
- Ce vei realiza azi?
- Ce impedimente v-ar putea bloca progresul?

Obiectivul Scrum Master/Team Lead este de a elimina orice impediment identificat de echipă.

Sprint Review și întâlniri retrospective

O **revizuire Sprint** are loc la sfârșitul Sprintului pentru a inspecta Creșterea și pentru a adapta Product Backlog, dacă este necesar. În timpul evaluării Sprintului, echipa de proiect și proprietarul produsului au evaluat ce s-a făcut cu acest Sprint. Pe baza acestei activități și a oricăror modificări care au avut loc în Backlog în timpul Sprintului, participanții planifică următorii pași care ar putea fi făcuți în ceea ce privește optimizarea valorii. Aceasta este o întâlnire informală, nu o întâlnire de statut, iar prezentarea Creșterii are scopul de a obține feedback și de a stimula colaborarea. Pentru Sprinturile care durează o lună, această întâlnire este stabilită pentru 4 ore.

Scrum Master/Team Leader se va asigura că evenimentul are loc și că participanții îi înțeleg scopul. Scrum Master îi ghidează pe toată lumea în timpul întâlnirii pentru a o păstra în intervalul de timp.

Sprint Review include următoarele elemente:

- Echipa de proiect și Product Owner participă la eveniment;
- Product Owner explică ce articole din Product Backlog au fost „Terminate” și ce nu au fost „Efectuate”;
- Echipa de dezvoltare discută ce a mers bine în timpul Sprintului, ce probleme au fost întâlnite și cum au fost rezolvate acele probleme;
- Echipa de dezvoltare demonstrează munca care a fost „realizată” și răspunde la întrebări despre creștere;
- Proprietarul de produs discută despre Product Backlog actualizat. El sau ea estimează datele așteptate pe baza progresului actual (dacă este necesar);
- Întreaga echipă plănuiește ce să facă în continuare, astfel încât Sprint Review să ofere informații valoroase pentru următoarea planificare Sprint.

Rezultatul revizuirii Sprint este un Backlog de produse revizuit și actualizat pentru următorul Sprint. Backlogul de produse poate fi personalizat pentru a satisface noi oportunități.

Retrospectiva Sprint este o oportunitate pentru echipa de proiect de a se autoinspecta și de a crea un plan de îmbunătățiri care urmează să fie puse în aplicare în timpul următorului Sprint. Retrospectiva Sprint are loc după Revizuirea Sprintului și înainte de următoarea Planificare Sprint. Această întâlnire are o durată de 3 ore pentru Sprinturi de 1 lună.

Scrum Master/Team Leader participă ca membru al echipei de egali la întâlnire pentru a asigura responsabilitatea în timpul procesului Scrum. Scopul retrospectivei Sprint este de a:

- Inspectați cum a decurs ultimul Sprint în ceea ce privește oamenii, relațiile, procesele și instrumentele;
- Identificați și puneți în ordine articolele majore care au mers bine și potențialele îmbunătățiri;
- Creați un plan pentru implementarea îmbunătățirilor în modul în care echipa de proiect își desfășoară activitatea.

Prevederea detaliată a cerințelor

Evaluarea detaliată cuprinde:

- Identificarea și analiza infrastructurilor tehnice și de comunicații ale Beneficiarului;
- Specificarea fluxurilor de procese/activități relevante;
- Urmărirea activităților identificate cu privire la cerințele părților interesate, care au fost identificate în faza de evaluare generală, pentru a asigura acoperirea întregului domeniu de activitate;
- Identificarea cerințelor funcționale pe baza activităților;
- Urmărirea corelației dintre cerințele funcționale și funcționalitățile aplicației standard;
- Identificarea nevoilor de schimbare a funcționalităților standard și de dezvoltare a altora suplimentare;
- Identificarea potențialelor probleme și riscuri care pot afecta implementarea/utilizarea sistemului (inclusiv potențiale probleme de incompatibilitate între diverse module);
- Analizarea măsurilor de atenuare a riscurilor și de remediere a potențialelor probleme care pot fi prevăzute în această fază;
- Stabilirea și definirea grupurilor de lucru;
- Stabilirea cerințelor de instruire pentru utilizatorii finali.

Design detaliat (Model Storming)

Arhitectura funcțională a livrabilelor este definită în timpul iterației de proiectare. Această activitate presupune elaborarea unor scenarii detaliate de cazuri de utilizare, la nivelul unor cazuri particulare de utilizare, inclusiv rolurile, constrângerile și regulile asociate fiecărui caz, precum și potențialele excepții.

În același timp, este definită arhitectura tehnică, în special definiția tehnică a sistemului, inclusiv o descriere a infrastructurii suport (servele, stații de lucru, infrastructuri și protocoale de comunicații, surse de date, stocare de date etc.).

Dezvoltarea modelului informațional presupune definirea arhitecturii de date a sistemului, la nivel logic și fizic.

Modulele de sistem sunt proiectate și descrise în această fază.

Proiectarea sistemului poate identifica mai multe soluții, urmărind ușurința și eficiența realizării și implementării cerințelor beneficiarului, cu respectarea restricțiilor tehnice, organizatorice, financiare sau legale.

Procesul de proiectare pleacă de la nevoile și prioritățile Beneficiarului, având în vedere importanța implicării utilizatorilor sistemului, pentru o înțelegere corectă a proceselor de lucru și aprobarea oferită de utilizator cu privire la noul sistem.

Testare de confirmare QA

Echipa realizează o evaluare obiectivă pentru a asigura calitatea. Aceasta include găsirea defectelor, validarea faptului că sistemul funcționează așa cum a fost proiectat și verificarea îndeplinirii cerințelor. Toată lumea este în mod egal responsabilă pentru calitatea produsului sau succesul proiectului.

Aceasta înseamnă că testarea noastră de confirmare este efectuată de întreaga echipă, inclusiv de membrii echipei a căror experiență principală poate fi în programare, analiză de afaceri, baze de date sau administrare de sistem, nu doar testerii desemnați sau profesioniști în asigurarea calității.

Membrii echipei de proiect a căror experiență este în testarea software-ului sau în utilizarea unui anumit instrument de testare nu se limitează doar la a face acea activitate, ei vor colabora cu Product Owner la cerințele produsului și vor lucra cu alți membri ai echipei pentru a dezvolta cod de înaltă calitate care îndeplinește aceste cerințe.

În timpul Procesului de Dezvoltare Sprint, soluțiile evoluează prin colaborarea între echipe auto-organizate, interfuncționale, care au descoperit - adesea prin încercare și eroare - cele mai bune procese, practici și instrumente de utilizat în diferite contexte.

Dezvoltarea agilă are o abordare bazată pe primul test, mai degrabă decât abordarea testată la sfârșit specifică dezvoltării tradiționale. Testarea și codificarea agile sunt realizate în mod incremental și interactiv, construind fiecare caracteristică până când oferă suficientă valoare pentru a o lansa în producție.

Deoarece testarea agilă se bazează pe [feedback-ul regulat din partea utilizatorului final](#) sau a unui proprietar de produs, abordează, de asemenea, o problemă comună pe care o au multe echipe de software, care este construirea unei soluții greșite, deoarece echipa interpretează greșit o caracteristică și aliniază ceea ce văd cu ei. expertiză în dezvoltare, mai degrabă decât ceea ce spune cerința sau ceea ce dorește utilizatorul final.

4. Ciclu de viață aplicabil pentru testarea QA

Spre deosebire de metodologia Waterfall, testarea QA nu este secvențială sau efectuată după o fază de codare, ci mai degrabă continuă. Testarea continuă a QA este una dintre numeroasele activități continue care au loc simultan pe majoritatea Sprint-urilor, inclusiv:

- Construcție continuă;
- Integrare continuă (CI);
- Livrare continuă (CD);
- Implementare continuă.

În mod ideal, build-urile și testarea ar avea loc aproape zilnic, ceea ce înseamnă că dezvoltatorii vor împinge codul în depozit în fiecare zi, iar build-urile vor fi programate să ruleze la o anumită oră. Pentru a face acest pas mai departe, dezvoltatorii ar putea implementa cod nou la cerere. Pentru a implementa acest lucru, echipele vor folosi un proces de **integrare continuă și implementare continuă (CI/CD)**. CI/CD limitează posibilitatea unei versiuni eșuate în ziua Notei de lansare.

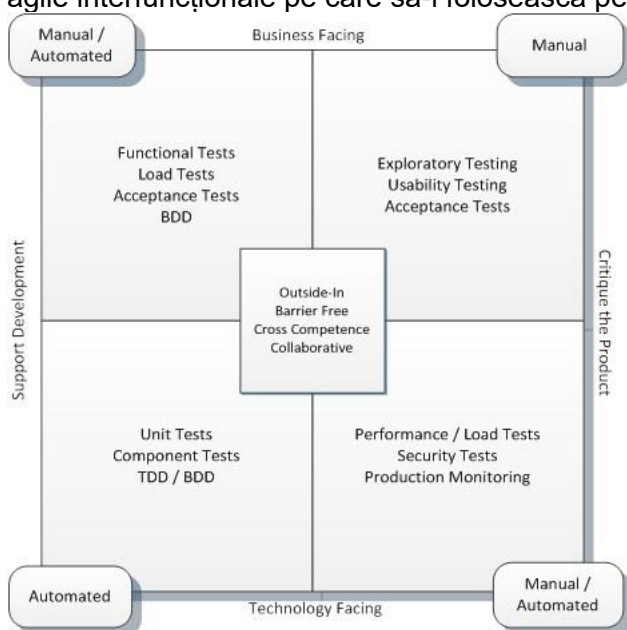
Integrarea continuă este o practică în care membrii unei echipe de dezvoltare folosesc un sistem de control al versiunilor și își integrează munca frecvent în aceeași locație, cum ar fi o ramură principală. Fiecare modificare este construită și verificată folosind teste și alte verificări pentru a detecta eventualele erori de integrare cât mai repede posibil. Cu automatizarea build-ului, construirea software-ului are loc automat, folosind instrumente precum Makefiles sau Ant, mai degrabă decât atunci când un dezvoltator invocă manual compilatorul.



În ultima etapă a unei **conducente CI/CD**, odată ce aplicația trece toate testele necesare, este apoi lansată în producție. Aceasta înseamnă, de fapt, eliberarea fiecărei versiuni bune pentru utilizatori.

5. Testarea cadranelor

Deoarece Agile este o metodologie de dezvoltare iterativă, testarea și codificarea sunt realizate în mod incremental și interactiv, unde caracteristicile pot evolua ca răspuns la cerințele în schimbare ale clienților. Testarea Agile acoperă toate tipurile de testare, inclusiv teste de unitate, de integrare, funcționale, de încărcare și de performanță. Următoarea diagramă Agile Testing Quadrants este un model util pentru echipele de dezvoltare agile interfuncționale pe care să-l folosească pentru a planifica și executa activități de testare.



Conceptul Agile Testing Quadrants are patru părți:

Quadrant Q1 : Acestea sunt teste orientate spre tehnologie care ghidează dezvoltarea, cum ar fi testele unitare, testele API, testarea serviciilor web și testele componentelor care îmbunătățesc designul produsului. Testele din Q1 sunt adesea asociate cu testarea automată și integrarea continuă.

Cadranul Q2 : Acestea sunt teste orientate spre afaceri care ghidează dezvoltarea, cum ar fi cele utilizate pentru testarea funcțională, testele de poveste, prototipurile și simulările care se asigură că produsele dvs. software sunt aliniate corect cu afacerea. Testele din Q2 sunt adesea asociate atât cu testarea automată, cât și cu cea manuală.

Quadrant Q3 : Acestea sunt teste orientate spre afaceri utilizate pentru a evalua sau critica produsul. Q3 acoperă teste precum testarea exploratorie, testarea bazată pe scenarii, testarea gradului de utilizare, [testarea de acceptare a utilizatorilor](#) și testarea alfa/beta și poate implica demonstrații de produse concepute pentru a obține feedback de la utilizatorii reali. Testele din Q3 sunt adesea asociate cu testarea manuală.

Quadrant Q4 : Acestea sunt teste orientate spre tehnologie utilizate pentru a evalua sau critica produsul. Q4 acoperă teste precum teste de performanță, încărcare, stres și scalabilitate, teste de securitate, mentenanță, gestionarea memoriei, compatibilitate și interoperabilitate, migrarea datelor, infrastructura și testarea de recuperare. Aceste teste sunt adesea automatizate.

Practica Scrum recomandată ar fi ca Scrum Master să organizeze un atelier cu clientul/utilizatorul final sau cu proprietarul produsului și cu echipa de dezvoltare, astfel încât să poată prezenta nota de lansare și rapoartele de testare necesare pentru fiecare revizuire Sprint.

6. Dezvoltare bazată pe teste de acceptare

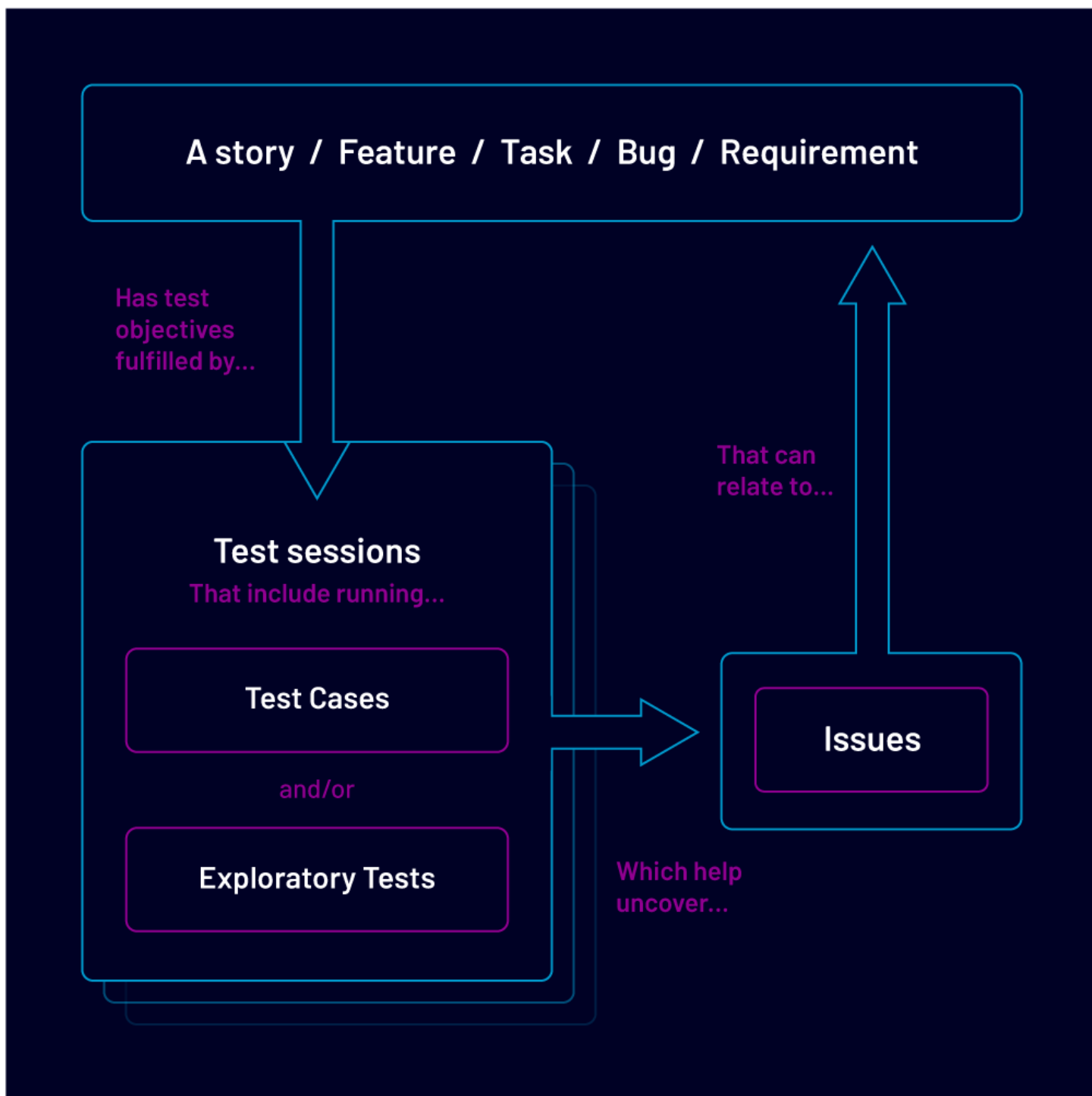
Dezvoltarea bazată pe teste de acceptare (ATDD) este o altă îmbunătățire a dezvoltării bazate pe teste care [promovează colaborarea](#) între proprietarul de produs, testerii și dezvoltatorii pentru a defini criteriile de acceptare automatizate înainte de a începe codarea. ATDD și TDD sunt tehnici complementare: ATDD ajută la descrierea obiectivelor de afaceri de nivel înalt, în timp ce TDD ajută dezvoltatorii să le implementeze ca cerințe. ATDD este o modalitate de a se asigura că toți membrii proiectului înțeleg ce este implementat, deoarece testele ATDD eşuate oferă feedback rapid cu privire la cerințele care nu sunt îndeplinite.

O parte cheie a testelor ATDD este că acestea rulează automat ori de câte ori se face o modificare a codului sursă. Pe lângă testarea aplicației, testele automate de acceptare sunt utile pentru a măsura progresul pe care îl face echipa ta de dezvoltare. În ceea ce privește un proiect agil, software-ul de lucru este considerat a fi singura măsură obiectivă a performanței.

7. Testare bazată pe sesiune

Testarea bazată pe sesiune este un tip de testare exploratorie structurată care necesită testatorilor să identifice obiectivele testului și să își concentreze eforturile de testare pe îndeplinirea acestora. Sesiunile de testare diferă de cazurile de testare în două moduri:

- mai mult de un test poate fi efectuat într-o singură sesiune;
- Ca și cazurile de testare, sesiunile de testare vă pot spune cine a testat ce, dar vă pot spune și cum, când și de ce au fost efectuate testele.



Scripturile de testare pre-scrise pot fi rulate în timpul unei sesiuni. Cu toate acestea, deoarece sesiunile de testare pun accentul pe obiectivele testelor față de cazurile de testare specifice, testerii sunt încurajați să creeze și să execute mai multe teste pe baza a ceea ce au descoperit și învățat. Acest tip de [testare exploratorie](#) este o modalitate extrem de puternică de optimizare a acoperirii testelor fără a suporta cheltuieli asociate cu scrierea și întreținerea cazurilor de testare.

8. Test de automatizare

Testarea de automatizare funcționează pe proiecte agile prin rularea unui număr mare de teste în mod repetat pentru a vă asigura că o aplicație nu se întrerupe ori de câte ori sunt introduse noi modificări la nivel de unitate, la nivel de API și la nivel de interfață grafică. Pentru multe echipe de dezvoltare Scrum, aceste teste automate sunt executate ca parte a unui proces de [integrare continuă \(CI\)](#), în care dezvoltatorii verifică codul din depozitul

partajat de mai multe ori pe zi. Fiecare înregistrare este apoi verificată printr-o versiune automată, permițând echipelor să detecteze erorile și conflictele cât mai curând posibil. Instrumentele CI precum Jenkins, Bamboo și Selenium sunt, de asemenea, folosite pentru a construi, testa și implementa aplicații automat atunci când cerințele se modifică pentru a accelera procesul de lansare.

Automatizarea testelor permite echipelor de dezvoltare Scrum pentru a executa mai multe teste în mai puțin timp, crește acoperirea și permite profesioniștilor QA să facă mai multe teste exploratorii de nivel înalt. Deoarece scripturile de testare de automatizare sunt reutilizabile, ele pot fi folosite pentru a face teste mai cuprinzătoare prin testarea pașilor repetitivi cu diferite seturi de date, cum ar fi cele pentru compatibilitatea între browsere sau dispozitive încrucișate.

Printre acestea **riscurile de automatizare** se numără cele legate de necesitatea controlului versiunilor și mentenabilitatea scripturilor de testare și a rezultatelor testelor. Alegerea instrumentului potrivit de testare a automatizării este extrem de importantă, deoarece doriți să le evitați pe cele care sunt incompatibile cu alte instrumente de testare software din mediul dvs. de testare. Odată ce aveți un instrument de automatizare a testelor care funcționează bine cu celelalte instrumente de testare, testele automate ar trebui, de asemenea, efectuate în mod regulat pentru a oferi feedback continuu despre starea întregului dvs. sistem, de preferință prin abordarea de integrare continuă.

9. Testare de integrare

Testarea integrării evaluează funcționalitatea diferitelor module atunci când sunt integrate pentru a forma o singură unitate. Această testare validează tranzițiile fără probleme între diferitele componente integrate ale software-ului. Scopul testării integrării este de a găsi defecte și defecțiuni între mai multe interfețe software.

Testarea integrării include următorii pași:

1. Înțelegerea arhitecturii aplicației;
2. Găsirea diferitelor module ale sistemului;
3. Înțelegerea funcționalității fiecărui modul;
4. Evaluarea tranzațiilor de date între interfețe;
5. Analiza punctelor de intrare și ieșire din sistem;
6. Pregătirea planului de testare;
7. Selectarea unei abordări de testare;
8. Clasificarea modulelor în funcție de nevoile de testare;
9. Găsirea diferitelor condiții de testare pentru cazurile de testare;
10. Proiectarea **scenariilor de testare, scripturilor și cazurilor de testare** ;
11. Implementarea modulelor alese și efectuarea testelor de integrare;
12. Executarea cazurilor de testare;
13. Urmărirea defectelor și înregistrarea rezultatelor.

Planuri de testare

De obicei, **planul de testare** este scris și actualizat pentru fiecare plan de lansare. Un plan de testare va conține următoarele:

- Domeniul de aplicare al testării;
- Noi funcționalități de testat;
- Tipuri de testare/ **Niveluri de testare** ;
- de performanță la stres și **sarcină** ;
- Luarea în considerare a infrastructurii;
- Planul de riscuri;
- Planificarea misiunii;
- Note de lansare.

Documentația în evoluție

Pentru a avea o soluție potențial funcțională pentru fiecare iterație, trebuie să menținem **documentația livrabilă** sincronizată cu software-ul/soluția - cu alte cuvinte, să continuăm să scriem documentația livrabilă în mod continuu pe tot parcursul proiectului. Documentația livrabilă include de obicei manuale de utilizare, materiale de instruire, manuale de operațiuni, manuale de asistență și prezentări generale ale sistemului. Nu include specificațiile cerințelor sau specificațiile de proiectare, cu excepția situațiilor în care o astfel de documentație este necesară sau în negocierile contractuale în care este cerută ca parte a contractului. Documentația livrabilă pentru iterația N este scrisă în timpul iterației N+1.

DevOps

Adoptarea **DevOps** asigură că software-ul trece rapid de la mediul de testare și punere în scenă la producție. Mediul care găzduiește aceste aplicații este furnizat rapid, lucrând adesea într-un serviciu cloud.

Echipele de dezvoltare trebuie să proiecteze, să dezvolte, să livreze și să ruleze software-ul cât mai rapid și mai fiabil posibil. Practicile DevOps sunt aplicate pentru a identifica și rezolva problemele cât mai curând posibil prin monitorizare, predicție a defecțiunilor, gestionarea mediului și remedierea problemelor. Această abordare comună în DevOps este combinată cu capacitatea de a monitoriza și analiza blocajele și de a optimiza cât mai repede posibil.

În multe cazuri, echipa noastră de proiect ia în considerare viabilă pentru un proiect/produs, implementarea practicii DevOps care poate adăuga valoare organizației dumneavoastră prin diferite beneficii. DevOps va acoperi o serie de procese din ciclul de viață al dezvoltării software:

- **Definiți și planificați** - planificați fluxurile de lucru DevOps pentru iterații, gestionarea versiunilor și urmărirea problemelor.
- **Codați, construiți și configurați** - dezvoltarea și revizuirea codului, gestionarea codului sursă și îmbinarea codului.
- **Testare** - se verifică dacă calitatea versiunii software și a codului sunt menținute pe tot parcursul procesului de dezvoltare și dacă cea mai înaltă calitate este implementată în producție.
- **Ambalare și preproducție**, care se referă la activitățile implicate odată ce lansarea este gata pentru implementare; se mai numește și punere în scenă sau preproducție.
- **Lansare, implementare și orchestrare**, care este procesul de lansare efectivă a software-ului și implică, de obicei, gestionarea modificărilor, aprobări de lansare, automatizarea lansărilor, orchestrarea programului, furnizarea și implementarea în producție.
- **Management continuu și configurare** - automatizarea continuă a configurației, managementul configurației și infrastructura ca cod.
- **performanței aplicației și identificarea problemelor** care afectează experiența utilizatorului.

Echipa noastră Scrum va produce software în iterații scurte pe un program de livrare continuă de noi funcții și remedieri de erori în cicluri rapide de la două până la patru săptămâni. În schimb, DevOps va reuni echipele de dezvoltare și operațiuni pentru a se concentra pe eliminarea silozurilor, pentru a reduce timpul de abordare a feedback-ului clienților și pentru a elimina blocajele pentru a permite livrarea continuă a software-ului. În consecință, ei pot construi, testa și lansa software-ul mai rapid, cu cât mai multă eficiență și viteză posibil.

Operațiunile noastre DevOps vor include următoarele:

- Integrare continuă – codificare, construire, integrare și testare.
- Livrare continuă - integrare continuă, dar se concentrează în principal pe lansările de produse.
- Implementare continuă - automatizarea lansărilor proiectelor cât mai curând posibil.
- Efectuarea operațiunilor de dezvoltare de management al configurației și monitorizare continuă.

Managementul lansărilor

Planificarea, coordonarea și verificarea implementării soluțiilor IT în producție fac parte din procesul **de gestionare a lansărilor**. Managementul lansării va necesita colaborarea între Echipa de proiect care produce soluția și persoanele responsabile pentru infrastructura IT operațională a Beneficiarului.

Scopul proceselor de management al versiunilor este de a coordona dezvoltarea, operațiunile și implementarea software-ului, asigurând în același timp alinierea la prioritățile de afaceri. În cazul majorității misiunilor, procesul este construit pentru mai multe obiective cheie:

- Gestionarea riscului;
- Coordonarea resurselor IT;
- Asigurarea conformității și a proceselor de audit;
- Supravegherea trecerii la versiuni noi;
- Menținerea alinierii afacerii cu dezvoltarea de software.

Componentele procesului de management al lansării

- **Conducta de eliberare** - un proces specific de lansare de la planificarea caracteristicilor până la livrare;
- **Release Value Stream** - procesele de lansare care adaugă sau creează valoare în conducta de lansare;
- **Politica de lansare** - definirea tipurilor de lansări, standardelor, cerințelor de guvernare pentru o organizație;

- **Șablon de lansare** - un singur proces de flux de lucru repetabil pentru pipeline de lansare care include activități umane și automate și urmează politicile de lansare ale unei organizații;
- **Release Plan** - o instanță a unui șablon de lansare dezvoltat pentru o anumită ediție;
- **Plan de implementare** - activități de implementare a unei ediții în mediul de producție;
- **Release unit** - setul de artefacte eliberate împreună pentru a implementa o caracteristică specifică;
- **Pachet de lansare** - o combinație de una sau mai multe unități de lansare implementate împreună ca o singură lansare din cauza interdependențelor, programării sau priorităților de afaceri;
- **Lansări majore** - pachete de lansare rare care includ multe unități de lansare care au un impact ridicat sau critic asupra afacerii;
- **Lansări minore** - pachete de lansare mai frecvente cu mai puține unități de lansare care nu includ componente critice pentru misiune.

Factorii de proces care vor fi luați în considerare pentru gestionarea versiunilor și Notele de lansare sunt:

- **Planificarea lansării** - Product Owner cu Scrum Master va identifica și va crea programul de lansare care va conține Note de lansare pentru fiecare Sprint planificat. Există multe strategii pe care organizațiile pot alege să le adopte atunci când vine vorba de programarea lansărilor, inclusiv trenuri de lansare, fluxuri de lansare, lansări ad-hoc și ferestre de lansare.
- **Configurarea infrastructurii** - Echipa de proiect va lucra îndeaproape cu Product Owner pentru a efectua gestionarea configurației mediului operațional. Pentru a implementa în siguranță în producție, trebuie să știți ce este în prezent în producție și cum depind aceste elemente hardware și software unele de altele. Cu cât infrastructura dumneavoastră operațională este mai complexă și cu cât aveți mai multe echipe de livrare IT, cu atât acest factor de proces devine mai important.
- **Pregătirea producției** - o parte a procesului de lansare este de a verifica dacă soluția este gata pentru a fi implementată și că părțile interesate sunt gata să le fie implementată. Cu cât lansările tale sunt mai mari și mai rare, cu atât aceasta devine o problemă.
- **Echipele de livrare sprijină** - Product Ownerul va lucra îndeaproape cu echipa de proiect pentru a le ajuta să implementeze cu succes.

Lansați fluxul de lucru

Planul de lansare trebuie să aibă următoarele elemente cheie :

- Cronologie;
- Date de livrare;
- Cerințe.

Lansați Build

Cu planul de lansare finalizat, puteți începe să proiectați și să construiți produsul pentru lansare. Aceasta este „dezvoltarea” reală a produsului pe baza cerințelor prezentate în planul de lansare.

Odată ce toate problemele care ar fi putut apărea sunt abordate, este timpul să supuneți construcția la testarea scenariilor din lumea reală.

Acest lucru ar putea dura mai multe iterații. Pe măsură ce echipa elaborează produsul, acesta este trimis (de obicei automat) într-un mediu de testare pentru acceptarea utilizatorului. Acest lucru permite echipei să identifice orice erori sau probleme care pot apărea în mediul real.

Pe măsură ce sunt identificate probleme, versiunea este trimisă înapoi pentru dezvoltare în etapa a doua. Cu alte cuvinte, în cadrul procesului iterativ de management al lansării, munca poate decurge de la etapa a doua la etapa a treia și înapoi până la aprobarea lansării.

Testarea de acceptare a utilizatorului

Testarea de acceptare a utilizatorilor, cunoscută și sub numele de UAT, este atunci când utilizatorii finali încep să o folosească și oferă feedback. Acest lucru se face adesea ca o încercare beta online gratuită sau produsul este partajat unui grup mai mare de angajați din cadrul companiei.

Testarea de acceptare a utilizatorilor este cel mai important pas pentru gestionarea lansării, din cauza cantității de date colectate și a remediilor necesare pentru a duce versiunea acolo unde trebuie să fie pentru lansarea oficială.

După cum am menționat mai devreme, aceasta face parte dintr-un proces iterativ. Pe măsură ce sunt identificate erori, echipa se întoarce la planșa de desen pentru a remedia problemele și a reproiecta construcția pentru o mai mare integritate. Compilarea trebuie să treacă de etapa UAT pentru a fi luată în considerare pentru implementarea și lansarea finală.

Pregătirea eliberării

Acest pas este de a pune ultimele retușuri asupra produsului, ținând cont de tot ce a fost învățat în timpul UAT. Pregătirea lansării include, de asemenea, o evaluare finală a calității de către Testerul de software.

În timpul revizuirii, Testerul de software va efectua verificări finale pentru a se asigura că versiunea îndeplinește standardele minime acceptabile și cerințele de afaceri prezentate în planul de lansare.

Deși UAT și asigurarea calității nu pot reproduce întotdeauna fiecare scenariu care ar putea apărea odată cu lansarea produsului, acești pași pot ajuta echipa să găsească și să remedieze cele mai comune erori, astfel încât să fie mai ușor să anticipăm și să prevenim orice probleme la lansare.

Odată ce revizuirea este finalizată, echipa funcțională va valida constatările și va finaliza lansarea pentru implementare. Înainte ca versiunea să poată fi implementată într-un mediu activ, aceasta trebuie să fie aprobată de proprietarul produsului.

Implementarea unei ediții

Ziua cea mare a sosit în sfârșit și aici este locul în care toată munca grea a echipei tale dă roade. Este timpul să vă lansați produsul în sălbăticia mediului de producție live.

Pe lângă simpla trimitere a construcției în producție, etapa de implementare include, de asemenea, mesaje și educație despre produs atât pentru utilizatorul final, cât și pentru compania dumneavoastră în general.

De exemplu, utilizatorii ar trebui să fie informați cu privire la modificările odată cu lansarea și despre modul de operare în cadrul noilor funcții. În funcție de cât de semnificative au fost schimbările, poate fi necesar să oferiți o pregătire solidă și continuă pentru a pune toată lumea la curent.

Acest lucru este deosebit de important pentru versiunile interne în care angajații care utilizează software-ul trebuie să îl înțeleagă pentru a-și face munca eficient și productiv.

În cele din urmă, în timpul etapei de implementare, echipa de proiect ar trebui să se întâlnească pentru a evalua performanța versiunii și a discuta cum a decurs implementarea. Dacă există probleme persistente, acestea ar trebui identificate și documentate pentru următoarea iterație.

Mentținerea stocului de produse

- Proprietarul de produs va menține actualizat stocul de produse, astfel încât să reflecte lista prioritarizată a funcționalităților dorite;
- Răspunzând întrebărilor venite de la dezvoltatori – Product Owner-ul va fi disponibil în orice moment echipei de dezvoltare pentru a răspunde eventualelor întrebări de clarificare, evitând astfel comunicarea complexă și formală în cadrul proiectului. Acest lucru este esențial pentru a ne asigura că echipa are toate informațiile la timp pentru a livra un produs funcțional la sfârșitul sprintului;
- Acceptarea pachetelor de lucru – pachetele de lucru livrate sunt prezentate Beneficiarului pentru acceptare la finalul fiecărui sprint. Beneficiarul va accepta pachetul de lucru sau va notifica Furnizorul cu privire la orice defecte pe parcursul următorului sprint.

Deși nu este strict necesar, Product Owner-ul poate participa la întâlnirile de stand-up ale echipei pentru a fi informat cu privire la progres și eventualele blocante pentru o reacție imediată.

Product Owner decide, de asemenea, asupra lansărilor de produs, conform planului de lansare. De asemenea, conform principiilor metodologiei Agile de management de proiect, Beneficiarul va defini Declarația de viziune a produsului și Foaia de parcurs pentru produs pentru a urmări progresul și pentru a asigura dezvoltarea adecvată a produsului.

Iterație de tranziție

Aceasta este faza de finalizare a proiectului; toate cerințele stabilite au fost îndeplinite. În această fază nu mai există elemente sau probleme (în produsul nerezolvat) de abordat și nu mai pot fi identificate altele noi. Produsul este gata de livrare și această acțiune este pregătită (prin integrare, testare, documentare).

Faza de implementare și testare funcțională presupune configurarea și integrarea componentelor sistemului pe infrastructura situată în mediul Beneficiarului. Se vor configura bazele de date, serverele de aplicații, modulele funcționale și alte componente, realizându-se și interconectarea modulară. Ulterior, sistemul este testat din perspectivă funcțională, iar rezultatele testelor sunt consemnate în rapoarte de testare. Această activitate are ca scop identificarea erorilor potențiale ale programului, care ar putea avea un impact negativ asupra activității viitoare.

Faza de implementare se realizează cu perturbarea minimă a activității utilizatorilor. Aceasta poate include călătoria utilizatorului, instalarea configurației hardware și software a aplicațiilor, precum și integrarea cu sistemele existente. Pentru a minimiza timpul de nefuncționare a activității organizației, testele de sistem vor fi efectuate pe configurații de testare, situate în interiorul sau în exteriorul instituției.

Testarea de acceptare finală

Testarea de acceptare are scopul de a confirma că sistemul funcționează conform așteptărilor și cerințelor. Testarea de acceptare se concentrează pe cerințele Beneficiarului.

Testarea de acceptare are scopul de a confirma faptul că produsul livrat îndeplinește cerințele inițiale identificate în faza de evaluare.

Pe parcursul acestui proces, Beneficiarul verifică dacă setul de specificații solicitate, prevăzut în raportul de evaluare, se regăsește printre funcționalitățile furnizate de produsul livrat.

Testarea de acceptare are ca rezultat o matrice de conformitate a produsului cu specificațiile cerute.

Implementarea și integrarea aplicațiilor sunt urmate de testare funcțională, care presupune:

- Identificarea funcțiilor pe care trebuie să le îndeplinească sistemul (în corelație cu raportul de evaluare);
- Crearea datelor de intrare pe baza specificațiilor;
- Determinarea rezultatelor pe baza specificațiilor;
- Executarea cazului de testare;
- Compararea ieșirii sistemului cu ieșirea așteptată.

Testarea finală a sistemului

Testarea nefuncțională, adică testarea caracteristicilor nefuncționale ale sistemului, are scopul de a măsura caracteristicile produsului care pot fi cuantificate la scară largă. Testarea nefuncțională include următoarele:

- **Testare de utilizare** – determină măsura în care produsul este înțeles, ușor de învățat, ușor de operat, atrăgător pentru utilizatori;
- **Stres testing** – evaluează sistemul dincolo de limitele specificate;
- **Testare de securitate** – investigarea funcțiilor care detectează amenințările;
- **Testarea capacității de stocare** – analizează memoria ocupată și validarea capacității de stocare necesare;
- **Testarea performanței** – determină performanța sistemului;
- **Testare de recuperare** – testează opțiunile de backup și de recuperare în caz de incidente;
- **Testarea volumului de date** – numai pentru sistemele care gestionează volume extreme de date;
- **Testarea instalării** – validează instalarea corectă a produsului;
- **Testarea documentației** ;
- **Testare de sarcină** – comportamentul sistemului în condiții de încărcare crescută.

Antrenează utilizatorii finali

Sesiunile de instruire trebuie să includă aspecte teoretice și/sau aspecte practice privind demonstrația pentru livrarea eficientă a transferului de competențe, metodologia la locul de muncă este selectată deoarece permite participanților să învețe prin îndeplinirea unor sarcini specifice la locul de muncă sau simularea loc de muncă. Fiecare metodologie de livrare a instruirii este prezentată în Curriculumul de formare și va fi descrisă în timpul fazei de inițiere a instruirii.

Evaluarea **participanților** va avea o abordare informală, trainerul va evalua imediat participanții cu privire la performanță în timp ce îndeplinește sarcinile atribuite în timpul trainingurilor la locul de muncă.

Limba sesiunilor **de instruire** - formare se desfășoară în limbile română sau engleză, după caz. Limba sesiunii de instruire va fi definită în timpul procedurilor de inițiere și organizare a instruirii, așa cum este descris mai jos.

Documentație de instruire – curricule, cursuri de formare (manuale, tutoriale video, chestionare etc.) pentru administratori, furnizori de servicii și utilizatori finali (persoane fizice și companii) dezvoltate în platforma de e-learning bazată pe Moodle LMS.

Instruirea utilizatorilor sistemului se realizează diferit, în funcție de competențele fiecărei categorii de utilizatori, de preferință în funcție de grupuri de utilizatori.

Procesul de instruire se desfășoară diferit , în funcție de nivelul de cunoștințe al fiecărui grup de utilizatori, conform metodologiilor standard de instruire în materie.

Rapoartele de instruire , transmise după fiecare sesiune de instruire, includ:

- Lista participanților ;
- Agenda sesiunii de antrenament;
- Materiale de instruire (prezentări, laboratoare etc.);
- Rezultatele testelor pentru cursanți.

De asemenea, echipa de proiect va informa Proprietarul de proiect despre orice cerințe, cum ar fi:

- **Cerințe de echipamente** - resurse materiale necesare pentru a susține sesiunea de instruire, cum ar fi, dar fără a se limita la hardware, software, rețele etc.
- **Cerințe de mediu** - condiții, cerințe de facilități, locație etc.
- **Cerințe de personal** - resursele umane necesare pentru a susține programul sesiunii de formare, inclusiv formatorii.
- **Dependențe și limitări** , dacă există.

Productie

În această fază, produsul livrat a fost validat și acceptat de către Beneficiar, iar utilizatorii au fost instruiți cu privire la utilizarea sistemului.

Eventualele bug-uri sau defecțiuni identificate după acest timp constituie obiectul contractului de garanție, iar dezvoltarea de funcționalități suplimentare necesită încheierea de acorduri cu Furnizorul.

Data de implementare a sistemului este data la care începe să se deruleze termenul de garanție.

Rapoarte de progres

Conform metodologiilor de lucru descrise mai sus, au fost stabilite mecanisme de comunicare care să asigure că informațiile necesare sunt generate și utilizate eficient. În acest context:

- **Analiza progresului** este utilă pentru revizuirea progresului în comparație cu planul. Aceasta poate fi, de asemenea, o oportunitate pentru prezentarea și discutarea rapoartelor scrise sau pentru o evaluare orală a problemelor curente;
- **Rapoartele de progres ale proiectului** oferă un rezumat al progresului proiectului, inclusiv informații cheie din indicatorii fizici și financiari.

Rapoartele de progres vor fi întocmite într-un format standard care să permită compararea rapoartelor în timp. Scopul rapoartelor de progres va fi acela de a oferi actualizări ale performanței în comparație cu reperele și reperele.

Echipa de proiect va pregăti rapoarte pe toată perioada contractului. Rapoartele vor acoperi toate activitățile proiectului și vor nota toate rezultatele obținute de Product Owner.

Raport de progres Sprint

Aceste rapoarte vor prezenta principalele progrese ale perioadei raportate, evoluția activităților și întârzierile, dificultățile întâmpinate, abaterile de la planul de activitate, precum și resursele implicate. Fiecare raport de activitate va fi însoțit de un raport financiar.

Product Owner-ului i se va furniza, la cerere, rapoarte intermediare de progres cu privire la probleme specifice identificate de reprezentanții lor.

Rapoartele vor oferi, de asemenea, informații despre rezultatele fiecărui pas, soluțiile date și deciziile majore care trebuie luate în considerare atunci când se iau în considerare anumite aspecte ale activității.

Raport final

Versiunea preliminară a Raportului va fi transmisă Proprietarului Produsului cu cel puțin o lună înainte de încheierea perioadei de execuție declarată de Furnizor. Acesta va descrie întregul proces de implementare și va facilita evaluarea rezultatelor obținute atât în termeni calitativi, cât și cantitativi.

Raportul final va include:

- o evaluare a succesului și a constrângerilor majore pentru fiecare activitate și sarcină;
- realizările generale ale contractului;
- o evaluare a realizării rezultatelor propuse în contract;
- recomandări pentru acțiuni viitoare în vederea asigurării durabilității activităților, a rezultatelor așteptate după finalizarea contractului și a măsurilor care trebuie luate de Product Owner în acest sens.

Produse livrate

Activitățile de implementare a Contestațiilor electorale privind analiza, proiectarea, dezvoltarea, testarea, instruirea și implementarea sunt de așteptat să urmeze următorul calendar preliminar. Începutul și durata (în săptămâni) a cărei etapă sunt reprezentate mai jos începând cu începerea proiectului:

nr	Faza/activitate	1	2	3	4	5	6	7	8	9	...	80	82	82	83
1.	Etapa de inițiere a proiectului														

2.	Etapa de analiză și documentare	x	x	x											
3.	Etapa de dezvoltare			x	x	x	x	x	x	x	x	x			
4.	Etapa de documentare și instruire											x	x		
5.	Lansarea în exploatare industrială												x	x	
6.	Întreținere														x

1. Raport de început și Plan preliminar de proiect elaborat.
2. Documentul de proiectare al modulului elaborat (furnizează două variante inițiale de design ale actualizărilor modulului și versiunea finală a designului modulului, care include o descriere narativă a actualizărilor și funcționalităților).
3. Cod sursă compilat și documentat al modulului de reclamații electorale (inclusiv instrumentele și bibliotecile, acolo unde este cazul, și codul sursă intermediar pentru funcționalitățile actualizate în termenul convenit) complet compatibil și funcțional pe platforma Beneficiarului.
4. Pachetul de instalare a software-ului, care va include instrucțiuni de instalare și configurare (cum se instalează aplicația, cerințele hardware și software, descrierea și configurarea platformei, configurarea aplicației, procedurile de recuperare în caz de dezastru etc.) și licențele software (dacă este cazul) trimise.
5. Plan de testare software și raport furnizat.
6. Rapoartele de testare funcționale, de performanță și de securitate depuse.
7. Acceptarea în producție.
8. Instruire și documentație pentru utilizator furnizată.
9. Lansarea completă a Reclamațiilor electorale - Lansare live (în producție).

Metodologie Mentenanță și suport

1. Introducere

Este cunoscut faptul că în situația unor sisteme informatice complexe ce recurg la organizarea datelor în baze de date problema cea mai dificilă o constituie întreținerea și actualizarea bazei de date. Baza de date formată dintr-o multitudine de colecții sau subcolecții de date impune cu necesitate actualizarea datelor în scopul asigurării coerenței datelor pentru informarea corectă și exactă a utilizatorilor. În caz contrar se va ajunge la o dezinformare a factorilor de decizie sau a altor categorii de utilizatori cu toate implicațiile negative ce pot rezulta din aceasta.

La fel de necesară este și activitatea de corectare, adaptare și perfecționare a sistemului informatic în funcție de schimbările sau problemele ce intervin la nivelul unității beneficiare.

Mentenanța constă în inspectarea frecventă a software-ului livrat în scopul prevenirii situațiilor de nefuncționare a acestora, întreținerii, remedierii defectelor constatate atunci când este necesar s.a.

Activitatea de mentenanță include un proces de revizuire postimplementară pentru a se asigura că sistemele informatice nou implementate corespund obiectivelor, cerințelor și performanțelor prestabilite. Eventualele erori din crearea sau exploatarea sistemului trebuie corectate în cadrul procesului de mentenanță.

Pentru orice asistență necesară Beneficiarul va deschide un caz de asistență printr-un sistem existent de urmărire a problemelor. Echipa de proiect va oferi un răspuns inițial în sistem, ca un comentariu, care va fi, de asemenea, trimis Beneficiarului prin e-mail, în termenul maxim de răspuns inițial menționat în acest document. Asistența va fi acordată între orele 9:00 - 18:00, de luni până vineri, ora Chișinăului.

Pentru situațiile critice în care întreg sistemul informatic este indisponibil iar acest lucru are un impact major asupra serviciilor de business ale Beneficiarului, Furnizorul va oferi un serviciu de urgență 24/7, astfel încât să poată fi contactat pentru demarare.

Scrum Master al echipei propuse pentru implementarea proiectului va fi responsabil pentru managementul contractului și sprijinul oferit în condițiile contractului, pentru escaladări și orice alte subiecte legate de contract.

Furnizorul va asigura asistență tehnică, suport și garanție pentru toate componentele sistemului oferat pentru o perioadă de **12 luni** de la punerea acestuia în funcțiune.

2. Metodologie și standarde

În funcție de scopul urmărit, natura schimbărilor intervenite și urgența efectuării modificărilor în sistem există mai multe tipuri de mentenanță, astfel: mentenanță corectivă, adaptivă, perfectivă și preventivă.

Mentenanța corectivă se referă la efectuarea de modificări în cadrul sistemului cu scopul de a repara și înlătura defectele sau erorile din proiectare, programare sau implementare. Erorile de programare sunt de obicei mai ușor de efectuat și mai puțin costisitoare. Erorile de proiectare sunt dificil de realizat și mai costisitoare deoarece pot implica rescrierea mai multor componente de program. Erorile de cerințe informaționale ale utilizatorilor sunt cele mai dificile și mai scump de corectat din cauză că impun reproiectări mai extinse de sistem.

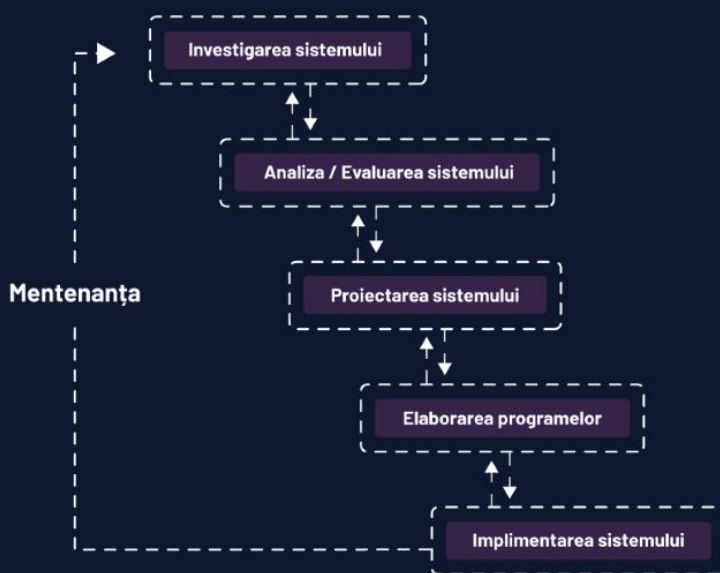
Mentenanța adaptivă presupune efectuarea de schimbări în cadrul sistemului informatic pentru a spori funcționalitatea acestuia sau pentru al adapta la un mediu nou, cum ar fi o platformă hardware diferită sau pentru a utiliza un alt sistem de operare, un alt sistem de gestiune a bazelor de date etc. De exemplu, pentru anumite situații sintetice de informare/raportare necesare conducerii societății comerciale, redactate sub formă tabelară, se solicită redarea acestora și sub formă grafică. Mentenanța adaptivă e mai puțin urgentă decât cea corectivă și reprezintă o mică parte din mentenanța sistemului, adăugându-i totodată valoare. Serviciile respective intervin drept rezultat al unei solicitări de dezvoltare.

Mentenanța perfectivă implică implementarea de noi cerințe de sistem funcționale sau nonfuncționale, cum ar fi îmbunătățirea performanțelor de procese, interfețe facile cu utilizatorii reducerea timpului de așteptare a clienților pentru efectuarea operațiilor de rezervare și cazare într-un hotel etc. Deci prin mentenanța perfectivă se urmărește să se adauge sistemului trăsături dorite, dar nu neapărat necesare.

Mentenanța preventivă implică schimbări făcute sistemului pentru a reduce sau înlătura riscul căderii sistemului datorită unor cauze, cum ar fi sporirea numărului de înregistrări mult peste cele prelucrate în mod curent. Un alt caz ar putea fi prevederea unor copii de siguranță și jurnale de urmărire a actualizărilor în scopul refacerii bazei de date în caz de incident. Pe parcursul ciclului de viață al sistemului mentenanța corectivă foarte probabil apare după instalarea inițială sau după schimbări majore ale sistemului. Într-o astfel de ultimă situație înseamnă că mentenanța adaptivă, perfectivă și preventivă dacă nu sunt atent proiectate și implementate pot duce la mentenanță corectivă.

Realizarea unui sistem informatic poate avea loc conform cu o serie de modele cum, ar fi: modelul în cascadă, modelul prototip, cu elemente de generația a patra, spirală etc. Indiferent de modelul adoptat realizarea unui sistem informatic presupune parcurgerea anumitor etape care definesc ciclul de viață a sistemului. Spre exemplificare, conform modelului în cascadă, ciclul de viață al unui sistem informatic este redat în figura următoare.

Procesul de mentenanță



Aspect iterativ de dezvoltare/perfecționare a unui sistem informatic

Din cele prezentate se poate desprinde concluzia că procesul de mentenanță începe imediat după implementarea și darea în exploatare a sistemului informatic.

Procesul de mentenanță implică următoarele activități majore:

- obținerea cerințelor de mentenanță;
- transformarea cerințelor în schimbări;
- proiectarea schimbărilor;
- implementarea schimbărilor.

3. Responsabilitățile de bază

1. Responsabilitățile de bază ale Furnizorului

Furnizorul va ajuta Beneficiarul în gestionarea proiectelor IT, conform listei de domenii care acoperă prezentul contract. În plus, furnizorul va:

- Răspunde la solicitările de asistență descrise în SLA - în timp rezonabil și de fiecare dată când se cere;
- Face tot posibilul să înțeleagă exact ce trebuie de făcut și ajută Beneficiarul în definirea termenilor și viziunilor exacte;
- Estimează timpul necesar pentru executarea corectă a cererii respective;
- Scalează orice problemă care poate apărea într-un mod adecvat, asigurându-se astfel că problema va fi rezolvată în cel mai scurt timp;
- Menține informarea Beneficiarului cu privire la progresul livrării proiectului;
- Facturarea Beneficiarului în funcție de activitățile desfășurate, timpul raportat și condițiile convenite;
- Cel puțin o dată pe lună, Managerul de proiect raportează Beneficiarului starea și progresul proiectelor în derulare;
- Menținerea în permanență a unei bune comunicări cu Beneficiarul.

2. Responsabilitățile de bază ale Beneficiarului

Beneficiarul va folosi asistența furnizorului acoperit de prezentul contract, conform intenției. În plus, Beneficiarul va:

- Solicită asistență prin intermediul instrumentelor și utilizând procedura descrisă în acest document;
- Furnizează informațiile și accesul solicitat în timp util;
- Oferă furnizorului accesul solicitat la echipamente, software și servicii în scopul finalizării sarcinilor;
- Menține furnizorul la curent cu schimbările potențiale a proiectelor în derulare;
- Achită serviciile furnizate în timp util;
- Menține în permanență o bună comunicare cu furnizorul.

4. Termeni și condiții

Furnizorul va aplica condițiile menționate în cazurile de asistență, care au fost raportate în consecință la contractul semnat.

Termenii se bazează pe presupunerea că sarcina poate fi înțeleasă în mod corespunzător. În cazul în care Indrivo nu poate identifica toate aspectele și informațiile necesare pentru activitatea solicitată, termenii serviciului, definiți de SLA atașat, se preconizează un efect de limitare.

Pentru a activa procesarea sarcinilor, Beneficiarul este obligat să trimită o informație detaliată, dar scurtă, care să clarifice sarcina Furnizorului de asistență. Beneficiarul va depune eforturi adecvate pentru a reduce descrierea la esența sa. Furnizorul va oferi Beneficiarului un plan de acțiuni cuprinzător, astfel încât Beneficiarul să poată iniția proiectul sau să refuze procesarea ulterioară.

1. SLA Terms

Termenii de reacție pe care Furnizorul le va asigura în cazul incidentelor raportate de Beneficiar se prezintă în tabelul de mai jos.

Tipul solicitării	Descrierea	Timpul de răspuns	Timpul soluționării
Critic	Sistemul total nefuncțional	15 Minute	4 ore
Înalt	O eroare care afectează majoritatea funcționalității sistemului	Maximum 1oră	2 zile
Mediu	A apărut o eroare într-o funcție, proces sau componentă, parțial nefuncțională.	Maximum 1oră	2 zile

Timp de răspuns - momentul în care Furnizorul va confirma primirea notificării și va înregistra apelul Beneficiarului; timpurile de răspuns sunt măsurate din momentul notificării unei solicitări valide depuse de Beneficiar și înregistrate la Furnizor.

Timpul de remediere, soluția finală - timpul necesar până când Furnizorul transmite etapele de implementare a soluției finale sau implementează soluția finală sau, dacă aplicația este modificată, până când Furnizorul transmite și acceptă cu Beneficiarul planul de a face modificarea într-o versiune ulterioară.

Timpul de implementare a soluției sau remedierii temporare se măsoară din momentul primirii notificării transmise de Furnizor și înregistrate către Furnizor, cu excepția perioadei de așteptare în care Beneficiarul oferă informații suplimentare pentru soluționarea incidentului.

Formatul de raportare:

#	ID-ul sarcinii	Denumirea activității/ descrierea	Statutul	Timpul alocat	Termen limită