Faculty of Computer Sciences and Engineering

Arkitektura Softuerike

Sistemi i Menaxhimit të Bibliotekës

Students:

Blend Kqiku,
Rigon Kadriu, Aldin Shabani, Aid Aliu

Professor:

Can.PhD. Xhelal Jashari

Qershor, 2025

Prishtinë

Change History

Date	Version	Description	Author
02/03/2025	1.0	Initial version	Rigon Kadriu, Blend Kqiku, Aldin Shabani, Aid Aliu
29/05/2025	2.0	Final Version	Rigon Kadriu, Blend Kqiku, Aldin Shabani, Aid Aliu
…	…	…	…
…	…	…	…
…	…	…	…

Content

1. Project Context

Sistemi i Menaxhimit të Bibliotekës (Library Management System) është një aplikacion softuerik që ndihmon në menaxhimin e një biblioteke në mënyrë dixhitale. Ai automatizon shumë procese që përndryshe do të bëheshin me dorë, si regjistrimi i librave, huazimi, kthimi i librave, regjistrimi i anëtarëve, etj.

Qëllimi dhe fushëveprimi i projektit

Qëllimi i këtij sistemi është të lehtësojë dhe të përmirësojë efikasitetin e operacioneve të bibliotekës. Sistemi mbulon proceset kryesore të menaxhimit të bibliotekës, duke përfshirë katalogimin e librave, regjistrimin e anëtarëve, huazimin dhe kthimin e librave, kërkimin e librave, gjenerimin e raporteve dhe menaxhimin e njoftimeve.

Përdoruesit e synuar dhe mjedisi

Sistemi është i destinuar për:

- Bibliotekarët: Të cilët menaxhojnë inventarin e librave, regjistrojnë anëtarët dhe procesojnë huazimet/kthimet.
- Anëtarët e bibliotekës: Të cilët kërkojnë libra, rezervojnë libra dhe shikojnë statusin e huazimeve të tyre.
- Administratorët: Të cilët menaxhojnë përdoruesit e sistemit dhe kanë qasje në raporte të detajuara.

- Sistemi është projektuar për të operuar në një mjedis të bazuar në web, duke lejuar qasje nga pajisje të ndryshme dhe lokacione të ndryshme.

2. Architecture Requirements

- ## 2.1 Key Objectives

  Modulariteti: Dizajnimi i një sistemi me komponentë të ndarë qartë dhe me ndërfaqe të mirëpërcaktuara.
- **Shkallëzueshmëria**: Mundësimi i rritjes së numrit të përdoruesve dhe të dhënave pa degradim të performancës.
- **Përdorshmëria**: Krijimi i një ndërfaqeje intuitive dhe të lehtë për përdorim për të gjitha kategoritë e përdoruesve.
- **Siguria**: Sigurimi i të dhënave të bibliotekës dhe informacioneve personale të anëtarëve.
- **Mirëmbajtja**: Lehtësimi i përditësimeve dhe zgjerimeve të sistemit në të ardhmen.

2.2 Use Cases

Rasti i Përdorimit 1: Huazimi i Librit (Blend Kqiku)

Përshkrimi: Bibliotekari proceson kërkesën e një anëtari për të huazuar një libër nga biblioteka.

Aktorët: Bibliotekari, Anëtari

Para-kushtet:

- Anëtari është i regjistruar në sistem.
- Libri është i disponueshëm në inventar.
- Anëtari nuk ka tejkaluar limitin e huazimeve ose nuk ka gjoba të papaguara.

Rrjedha Bazë:

1. Anëtari paraqet kartën e anëtarësimit dhe librin që dëshiron të huazojë.
2. Bibliotekari skanon ose fut manualisht ID e anëtarit.
3. Sistemi tregon statusin e anëtarit dhe limitin e mbetur të huazimeve.
4. Bibliotekari skanon ose fut manualisht ID-në e librit.
5. Sistemi kontrollon disponueshmërinë e librit dhe afatet e lejuara të huazimit.
6. Sistemi regjistron huazimin dhe përditëson statusin e librit në "I huazuar".
7. Sistemi lëshon një njoftim me datën e kthimit.
8. Bibliotekari i dorëzon librin anëtarit.

Rrjedha Alternative:

- Nëse anëtari ka gjoba të papaguara, sistemi njofton bibliotekarin dhe kërkon pagimin e gjobave para se të vazhdojë.
- Nëse libri nuk është i disponueshëm, sistemi tregon statusin aktual dhe datën e mundshme të kthimit.

Post-kushtet:

- Statusi i librit është ndryshuar në "I huazuar".
- Rekordi i huazimit është krijuar në sistem me detajet e anëtarit, librit dhe datës së kthimit.
- Numri i librave të disponueshëm në inventar është përditësuar.

Rasti i Përdorimit 2: Kërkimi i Librave (Rigon Kadriu)

Përshkrimi: Përdoruesi kërkon libra në bazë të kritereve të ndryshme si titulli, autori, kategoria, etj.

Aktorët: Bibliotekari, Anëtari

Para-kushtet:

- Përdoruesi ka qasje në sistemin e bibliotekës.

Rrjedha Bazë:

1. Përdoruesi hap modulin e kërkimit.
2. Përdoruesi fut kriteret e kërkimit (p.sh., titulli, autori, kategoria, ISBN).
3. Sistemi proceson kërkesën dhe shfaq rezultatet që përputhen me kriteret.
4. Përdoruesi mund të shfletojë rezultatet dhe të shohë detajet e çdo libri.
5. Përdoruesi zgjedh një libër për të parë informacione më të detajuara si disponueshmëria, vendndodhja fizike, etj.

Rrjedha Alternative:

- Nëse nuk gjenden rezultate që përputhen me kriteret, sistemi shfaq një mesazh dhe sugjeron kritere të tjera të kërkimit.
- Përdoruesi mund të rafinojë kërkimin duke shtuar filtra shtesë.

Post-kushtet:

- Përdoruesi ka marrë informacionin e kërkuar për librat.
- Sistemi mund të regjistrojë historinë e kërkimit për qëllime analitike (nëse konfigurohet).

Rasti i Përdorimit 3: Regjistrimi i Librit të Ri (Aldin Shabani)

Përshkrimi: Bibliotekari regjistron një libër të ri në inventarin e bibliotekës.

Aktorët: Bibliotekari

Para-kushtet:

- Bibliotekari është i autentifikuar në sistem me të drejtat e duhura.
- Të dhënat e librit janë të disponueshme (titulli, autori, ISBN, etj.).

Rrjedha Bazë:

1. Bibliotekari hap formularin e regjistrimit të librit të ri.
2. Bibliotekari fut të dhënat e librit (titulli, autori, botuesi, viti i publikimit, ISBN, kategoria, etj.).
3. Sistemi verifikon nëse libri me të njëjtin ISBN tashmë ekziston në sistem.
4. Bibliotekari cakton një vendndodhje fizike për librin në bibliotekë.
5. Bibliotekari konfirmon regjistrimin.
6. Sistemi krijon një rekord të ri për librin dhe gjeneron një ID unik.
7. Sistemi konfirmon regjistrimin e suksesshëm.

Rrjedha Alternative:

- Nëse libri me të njëjtin ISBN tashmë ekziston, sistemi tregon detajet ekzistuese dhe ofron opsionin për të shtuar një kopje të re ose për të ndryshuar detajet.
- Nëse të dhënat e futura janë të pasakta ose jo të plota, sistemi njofton bibliotekarin dhe kërkon korrigjim.

Post-kushtet:

- Libri i ri është shtuar në inventarin e bibliotekës.
- Libri është i gatshëm për huazim nga anëtarët.

Rasti i Përdorimit 4: Gjenerimi i Raporteve (Aid Aliu)

Përshkrimi: Administratori gjeneron raporte të ndryshme për të analizuar aktivitetin e bibliotekës.

Aktorët: Administratori

Para-kushtet:

- Administratori është i autentifikuar në sistem me të drejtat e duhura.

Rrjedha Bazë:

1. Administratori hap modulin e raporteve.
2. Administratori zgjedh llojin e raportit (p.sh., huazimet sipas kategorisë, librat më të huazuar, anëtarët aktivë, etj.).
3. Administratori përcakton periudhën kohore për raportin.
4. Administratori zgjedh formatin e raportit (PDF, Excel, etj.).
5. Sistemi proceson kërkesën dhe gjeneron raportin.
6. Sistemi shfaq raportin dhe ofron opsionin për ta shkarkuar ose printuar.

Rrjedha Alternative:

- Nëse nuk ka të dhëna për periudhën e zgjedhur, sistemi njofton administratorin dhe sugjeron një periudhë tjetër.
- Administratori mund të modifikojë parametrat e raportit dhe të gjenerojë një raport të ri.

Post-kushtet:

- Raporti është gjeneruar dhe është i disponueshëm për shkarkim ose printim.
- Sistemi mund të ruajë raportin për referencë të mëvonshme.

2.3 Stakeholders & Concerns

Bibliotekarët

- **Shqetësimet:** Efikasiteti i proceseve të përditshme, lehtësia e përdorimit, automatizimi i detyrave të përsëritura, saktësia e të dhënave.

Anëtarët e Bibliotekës

- **Shqetësimet:** Qasja e lehtë në katalog, rezervimi i librave, njoftime për afatet e kthimit, privatësia e të dhënave personale.

Administratorët e Sistemit

- **Shqetësimet:** Siguria e sistemit, menaxhimi i përdoruesve, monitorimi i performancës, gjenerimi i raporteve analitike.

Ekipi i Zhvillimit

- **Shqetësimet:** Modulariteti i kodit, testueshmëria, shkallëzueshmëria, dokumentimi, mirëmbajtja e kodit.

Drejtuesit e Bibliotekës

- **Shqetësimet:** Kosto e implementimit dhe mirëmbajtjes, raportet për vendimmarrje, përmirësimi i shërbimeve ndaj anëtarëve.

2.4 Constraints

Kufizimi 1: Kompatibiliteti me Sistemet Ekzistuese (Blend Kqiku)

Sistemi duhet të jetë i aftë të integrohet me sistemin ekzistues të katalogimit të librave që përdor standardin MARC21 (Machine-Readable Cataloging). Ky integrim është i nevojshëm për të shmangur humbjen e të dhënave historike dhe për të lehtësuar tranzicionin nga sistemi i vjetër në sistemin e ri. Kjo nënkupton përshtatjen e strukturës së të dhënave dhe zhvillimin e API-ve të posaçme për komunikim mes sistemeve.

Kufizimi 2: Përputhshmëria me Rregulloret e Privatësisë (Rigon Kadriu)

Sistemi duhet të përputhet me ligjet dhe rregulloret e mbrojtjes së të dhënave personale, duke përfshirë GDPR. Kjo kufizon mënyrën se si të dhënat e anëtarëve mund të ruhen, procesohen dhe ndahen. Sistemi duhet të implementojë mekanizma për pseudonimizimin e të dhënave, të ofrojë mundësinë për eksportimin dhe fshirjen e të dhënave personale, si dhe të mbajë gjurmë të aktiviteteve që përfshijnë përpunimin e të dhënave personale.

Kufizimi 3: Mundësia për Punë Offline (Aldin Shabani)

Sistemi duhet të jetë i aftë të funksionojë edhe në mungesë të lidhjes me internet, me sinkronizim të mëvonshëm të të dhënave kur lidhja të jetë e disponueshme. Kjo është e nevojshme për të siguruar funksionalitetin e vazhdueshëm të bibliotekës në rast të ndërprerjeve të internetit. Të dhënat e krijuara gjatë periudhës offline duhet të ruhen lokalisht dhe të sinkronizohen me serverin kryesor kur të rivendoset lidhja.

Kufizimi 4: Kapaciteti i Kufizuar i Serverit (Aid Aliu)

Sistemi duhet të operojë brenda kufizimeve të infrastrukturës ekzistuese të IT-së së bibliotekës, e cila ka një server me 16GB RAM dhe 1TB hapësirë diskut. Kjo kufizon numrin e transaksioneve të njëkohshme dhe sasinë e të dhënave që mund të ruhen. Sistemi duhet të implementojë strategji efikase të cache-imit dhe të optimizojë shfrytëzimin e burimeve për të ofruar performancë të kënaqshme brenda këtyre kufizimeve.

2.5 Non-functional Requirements

Kërkesa 1: Performanca (Blend Kqiku)

Sistemi duhet të jetë i aftë të procesojë të paktën 100 transaksione huazimi/kthimi në orë gjatë periudhave të pikut. Koha e përgjigjes për kërkesat e katalogut duhet të jetë më pak se 2 sekonda, edhe kur sistemi përdoret nga deri në 50 përdorues të njëkohshëm. Kërkimet komplekse me filtra të shumta duhet të kompletohen brenda 5 sekondave.

Kërkesa 2: Disponueshmëria (Blend Kqiku)

Sistemi duhet të ketë një disponueshmëri prej të paktën 99.5% gjatë orëve të punës së bibliotekës (9:00-20:00, nga e hëna deri të premten; 10:00-16:00 të shtunën). Mirëmbajtja e planifikuar duhet të kryhet jashtë këtyre orareve. Sistemi duhet të implementojë mekanizma të rimëkëmbjes nga gabimet për të minimizuar kohën e jofunksionalitetit.

Kërkesa 3: Siguria (Rigon Kadriu)

Të gjithë transaksionet e sistemit duhet të jenë të sigurta dhe të enkriptuara duke përdorur protokollet standarde të industrisë (TLS 1.3 ose më i ri). Të dhënat e ndjeshme të përdoruesve, si fjalëkalimet, duhet të ruhen të hashuar duke përdorur algoritme të forta kriptografike (p.sh., bcrypt). Sistemi duhet të implementojë autentifikim me dy faktorë për përdoruesit administrativë.

Kërkesa 4: Përdorshmëria (Rigon Kadriu)

Ndërfaqja e përdoruesit duhet të jetë intuitive dhe e lehtë për t'u përdorur, me një kurbë të shkurtër të të mësuarit për bibliotekarët e rinj (jo më shumë se 2 orë trajnim). Sistemi duhet të ofrojë ndërfaqe responsive që punon mirë në pajisje të ndryshme (desktop, tablet, celular). Të gjitha funksionet kryesore duhet të jenë të qasshme me jo më shumë se 3 klikime nga faqja kryesore.

Kërkesa 5: Shkallëzueshmëria (Aldin Shabani)

Sistemi duhet të jetë i aftë të akomodojë një rritje vjetore prej 20% në numrin e anëtarëve dhe librave pa degradim të performancës. Arkitektura duhet të lejojë shtimin e lehtë të burimeve shtesë (CPU, RAM, hapësirë diskut) kur nevojitet. Komponenti i bazës së të dhënave duhet të mbështesë shpërndarjen horizontale për të akomoduar rritjen e të dhënave.

Kërkesa 6: Mirëmbajtja (Aldin Shabani)

Sistemi duhet të jetë i lehtë për t'u mirëmbajtur, me një kohë mesatare për riparim (MTTR) jo më shumë se 4 orë për problemet kritike. Kodi duhet të jetë i mirëdokumentuar dhe i organizuar në module të qarta. Përditësimet e sistemit duhet të mund të implementohen pa ndërprerje të shërbimit (zero-downtime deployments).

Kërkesa 7: Ndërvepueshmëria (Aid Aliu)

Sistemi duhet të ofrojë API të dokumentuara mirë për integrim me sistemet e tjera të bibliotekës (p.sh., sistemi i financave, sistemi i prokurimit). API-të duhet të ndjekin parimet RESTful dhe të mbështesin formatet standarde të të dhënave (JSON, XML). Sistemi duhet të jetë i aftë të importojë dhe eksportojë të dhëna në formate standarde bibliotekare.

Kërkesa 8: Pajtueshmëria (Aid Aliu)

Sistemi duhet të përputhet me standardet ndërkombëtare të bibliotekave, duke përfshirë MARC21 për katalogimin, Z39.50 për kërkimin ndër-bibliotekë, dhe ISO 2709 për shkëmbimin e të dhënave bibliografike. Gjithashtu, sistemi duhet të përputhet me udhëzimet e qasjes WCAG 2.1 AA për të siguruar përdorshmëri për personat me aftësi të kufizuara.

2.6 Risks

Rreziku 1: Integriteti i të Dhënave

Përshkrimi: Procesi i migrimit të të dhënave nga sistemi ekzistues mund të rezultojë në humbje ose korruptim të të dhënave. **Impakti:** I lartë - Humbja e të dhënave historike ose korruptimi i inventarit mund të ndikojë negativisht në funksionalitetin e bibliotekës. **Strategjia e Zbutjes:** Implementimi i një plani të detajuar të migrimit me kontrolle të validimit, backup të plotë para migrimit, dhe një periudhë testimi paralel ku të dy sistemet funksionojnë njëkohësisht.

Rreziku 2: Rezistenca e Përdoruesve

Përshkrimi: Bibliotekarët mund të rezistojnë ndaj adoptimit të sistemit të ri për shkak të familjaritetit me sistemin ekzistues. **Impakti:** I mesëm - Adoptimi i ulët mund të zvogëlojë efektivitetin e investimit dhe të vonojë realizimin e përfitimeve. **Strategjia e Zbutjes:** Përfshirja e bibliotekarëve në procesin e dizajnimit, ofrimi i trajnimeve të gjera, dhe implementimi gradual i sistemit me periudha mbështetjeje intensive.

Rreziku 3: Problemet e Performancës

Përshkrimi: Sistemi mund të mos performojë sipas pritjeve kur ngarkohet me të dhëna reale dhe përdorues të shumtë. **Impakti:** I lartë - Performanca e dobët mund të ndërpresë operacionet e bibliotekës dhe të frustrojë përdoruesit. **Strategjia e Zbutjes:** Kryerja e testeve të ngarkesës dhe stres-testeve me të dhëna reale para lansimit, optimizimi i bazës së të dhënave, dhe planifikimi për burime shtesë harduerike nëse nevojitet.

Rreziku 4: Kompleksiteti i Sistemit

Përshkrimi: Sistemi mund të bëhet shumë kompleks për t'u përdorur nga stafi jo-teknik i bibliotekës. **Impakti:** I mesëm - Kompleksiteti i lartë mund të rezultojë në gabime të përdoruesve dhe në rënie të produktivitetit. **Strategjia e Zbutjes:** Fokusimi në dizajn të thjeshtë dhe intuitiv të UI/UX, krijimi i udhëzuesve të qartë të përdorimit, dhe implementimi i një sistemi ndihmë brenda aplikacionit.

2.7 Key Architectural Decisions (ADRs)

2.7 Key Architectural Decisions (ADRs)

ADR 1: Përdorimi i Arkitekturës Mikroshërbimeve (Blend Kqiku)

Konteksti: Sistemi i Menaxhimit të Bibliotekës duhet të mbështesë një sërë funksionalitetesh të ndryshme (menaxhimi i librave, anëtarëve, huazimeve, etj.) dhe duhet të jetë lehtësisht i zgjerueshëm për të akomoduar kërkesa të reja në të ardhmen. Gjithashtu, duhet të jetë i shkallëzueshëm për të përballuar rritjen e numrit të përdoruesve dhe të dhënave.

Vendimi: Vendosëm të përdorim një arkitekturë të bazuar në mikroshërbime, ku funksionalitetet e ndryshme të sistemit implementohen si shërbime të pavarura dhe të ndërlidhura përmes API-ve. Shërbimet kryesore do të përfshijnë: Shërbimin e Katalogut, Shërbimin e Anëtarësisë, Shërbimin e Huazimeve, Shërbimin e Njoftimeve, dhe Shërbimin e Raporteve.

Pasojat:

- **Pozitive:**
    - Zhvillim i pavarur dhe lansim i shpejtë për module të ndryshme.
    - Shkallëzueshmëri më e mirë, pasi shërbimet individuale mund të shkallëzohen sipas nevojës.
    - Izolimi i gabimeve, ku dështimet në një shërbim nuk ndikojnë direkt në shërbimet e tjera.
    - Fleksibilitet në zgjedhjen e teknologjive për shërbime të ndryshme.
- **Negative:**
    - Kompleksitet i shtuar në menaxhimin e komunikimit ndërmjet shërbimeve.
    - Sfida në menaxhimin e transaksioneve që përfshijnë shumë shërbime.
    - Nevojë për infrastrukturë më të avancuar për operimin dhe monitorimin e shërbimeve.
    - Kosto potencialisht më e lartë për zhvillim dhe operim fillestar.

ADR 2: Implementimi i Bazës së të Dhënave NoSQL për Katalogun (Rigon Kadriu)

Konteksti: Sistemi duhet të menaxhojë një katalog të madh librash me struktura të dhënash potencialisht të ndryshme (p.sh., libra, revista, media digjitale). Kërkimet në katalog duhet të jenë të shpejta dhe fleksibile, dhe struktura e të dhënave mund të evolojë me kalimin e kohës.

Vendimi: Vendosëm të përdorim një bazë të dhënash NoSQL (MongoDB) për modulin e katalogut, ndërsa do të ruajmë një bazë të dhënash relacionale (PostgreSQL) për modulet që kërkojnë konsistencë të lartë të të dhënave (p.sh., huazimet, anëtarësia).

Pasojat:

- **Pozitive:**
    - Fleksibilitet më i madh në strukturën e të dhënave të katalogut.
    - Performancë më e mirë për kërkime komplekse në katalog.
    - Shkallëzueshmëri më e lehtë horizontale për rritjen e katalogut.
    - Përshtatje më e mirë me formatet e ndryshme të meta-të dhënave bibliotekare.
- **Negative:**
    - Kompleksitet i shtuar në menaxhimin e dy sistemeve të ndryshme të bazave të të dhënave.
    - Sfida në ruajtjen e konsistencës ndërmjet bazave të të dhënave.
    - Nevoja për ekspertizë shtesë në ekipin e zhvillimit dhe operimit.
    - Mungesë e mbështetjes native për ACID transaksione në bazën e të dhënave NoSQL.

ADR 3: Përdorimi i React.js dhe Node.js për Zhvillimin e Frontend dhe Backend (Aldin Sh abani)

Konteksti: Sistemi kërkon një ndërfaqe moderne, responsive dhe të lehtë për t'u përdorur. Ekipi i zhvillimit ka nevojë për teknologji që mundësojnë zhvillim të shpejtë dhe efektiv, dhe që mbështesin mirë arkitekturën e mikroshërbimeve.

Vendimi: Vendosëm të përdorim React.js për zhvillimin e frontend-it dhe Node.js me Express për zhvillimin e backend-it. Kjo qasje mundëson përdorimin e JavaScript në të gjithë stack-un e zhvillimit, duke lehtësuar ndarjen e kodit dhe ekspertizës ndërmjet anëtarëve të ekipit.

Pasojat:

- **Pozitive:**
    - Kurbë e shkurtër e të mësuarit për ekipin, pasi të dy teknologjitë përdorin JavaScript.
    - Ekosistem i pasur i librarie dhe komponentëve gati për përdorim.
    - Performancë e mirë dhe përvoja e përdoruesit për aplikacionin web.
    - Mbështetje e mirë për zhvillimin e API-ve RESTful dhe komunikimin asinkron.
- **Negative:**
    - Varësi nga ekosistemi i JavaScript, i cili evoluon shpejt dhe mund të shkaktojë probleme të pajtueshmërisë.
    - Performanca e Node.js mund të mos jetë optimale për operacione intensive llogaritëse.
    - Nevoja për menaxhim të kujdesshëm të versioneve të librarive dhe varësive.
    - Sfida potenciale në skalimin e aplikacioneve Node.js për ngarkesa shumë të larta.

ADR 4: Implementimi i Sistemit të Njoftimeve me WebSockets dhe Email (Aid Aliu)

Konteksti: Sistemi duhet të njoftojë anëtarët për afatet e kthimit të librave, disponueshmërinë e librave të rezervuar, dhe njoftime të tjera të rëndësishme. Këto njoftime duhet të dërgohen në kohë reale kur anëtarët janë online dhe gjithashtu duhet të jenë të qasshme përmes kanaleve alternative.

Vendimi: Vendosëm të implementojmë një sistem të dyfishtë njoftimesh duke përdorur WebSockets për njoftime në kohë reale brenda aplikacionit dhe një sistem email-i për njoftime jashtë aplikacionit. Sistemi i njoftimeve do të jetë një mikroshërbim i dedikuar që përdor një arkitekturë të bazuar në ngjarje (event-driven) me një sistem mesazhesh (RabbitMQ) për të siguruar dërgimin e besueshëm të njoftimeve.

Pasojat:

- **Pozitive:**
    - Njoftime në kohë reale për përdoruesit që janë aktivë në sistem.
    - Qasje fleksibile për dërgimin e njoftimeve përmes kanaleve të ndryshme.
    - Shkallëzueshmëri e mirë për të përballuar volume të larta njoftimesh.
    - Besueshmëri e shtuar përmes përdorimit të sistemit të mesazheve për rradhitjen dhe riprovimin e njoftimeve.

- **Negative:**
    - Kompleksitet i shtuar në infrastrukturë me shtimin e WebSockets dhe sistemit të mesazheve.
    - Nevoja për menaxhimin e statusit të dorëzimit të njoftimeve.
    - Kosto shtesë për mirëmbajtjen e infrastrukturës së email-it.

Kufizime potenciale në disa klientë web që nuk mbështesin plotësisht WebSockets.

3. Architecture Solution

3.1 Chosen Architecture Style & Justification

Për Sistemin e Menaxhimit të Bibliotekës kemi zgjedhur një arkitekturë hibride që kombinon Mikroshërbimet me një Arkitekturë të Orientuar ndaj Shërbimeve (SOA). Kjo qasje na lejon të ndajmë sistemin në shërbime të pavarura, por gjithashtu të ruajmë një nivel të caktuar të centralizimit për funksionalitetet kryesore.

Justifikimi:

- **Modulariteti dhe Zhvillimi i Pavarur:** Arkitektura e mikroshërbimeve lejon ekipet të zhvillojnë, testojnë dhe lansojnë shërbime të ndryshme në mënyrë të pavarur, duke përshpejtuar ciklin e zhvillimit.
- **Shkallëzueshmëria:** Shërbimet individuale mund të shkallëzohen bazuar në kërkesat specifike të ngarkesës, pa nevojën për të shkallëzuar të gjithë sistemin.
- **Teknologji të Përshtatshme:** Mundësia për të përdorur teknologji të ndryshme për shërbime të ndryshme, bazuar në nevojat specifike të çdo shërbimi.
- **Izolimi i Gabimeve:** Dështimet në një shërbim nuk ndikojnë direkt në funksionalitetin e shërbimeve të tjera.
- **Evolucioni i Lehtë:** Sistemi mund të evoluojë me kalimin e kohës, duke shtuar ose modifikuar shërbime pa ndikuar në të gjithë sistemin.

3.2 Architecture Overview

Sistemi i Menaxhimit të Bibliotekës është i organizuar në këto shërbime kryesore:

Shërbimi i Autentifikimit dhe Autorizimit

- Menaxhon identitetin e përdoruesve, autentifikimin dhe kontrollin e qasjes.
- Implementon OAuth 2.0 dhe JWT për autentifikim të sigurt.

Shërbimi i Katalogut

- Menaxhon të dhënat e librave, revistave dhe materialeve të tjera.
- Ofron funksionalitete të avancuara kërkimi dhe filtrimi.
- Përdor bazë të dhënash NoSQL për fleksibilitet maksimal.

Shërbimi i Anëtarësisë

- Menaxhon të dhënat e anëtarëve, regjistrimin dhe rinovimin e anëtarësisë.
- Ruan historinë e aktivitetit të anëtarëve.
- Përdor bazë të dhënash relacionale për integritet të të dhënave.

Shërbimi i Huazimeve

- Menaxhon huazimin dhe kthimin e materialeve.
- Llogarit gjobat për vonesat.
- Menaxhon rezervimet dhe rradhët e pritjes.

Shërbimi i Njoftimeve

- Dërgon njoftime për afatet e kthimit, disponueshmërinë e materialeve të rezervuara, etj.
- Përdor kanale të ndryshme komunikimi (email, njoftime në aplikacion).

Shërbimi i Raporteve dhe Analizës

- Gjeneron raporte për menaxhimin e bibliotekës.
- Ofron analiza për tendencat e huazimeve, demografinë e anëtarëve, etj.

API Gateway

- Shërben si pikë hyrjeje e vetme për klientët e frontend-it.
- Menaxhon kërkesa API dhe i drejton te shërbimet e duhura.
- Implementon politikat e sigurisë dhe limitimit të kërkesave.

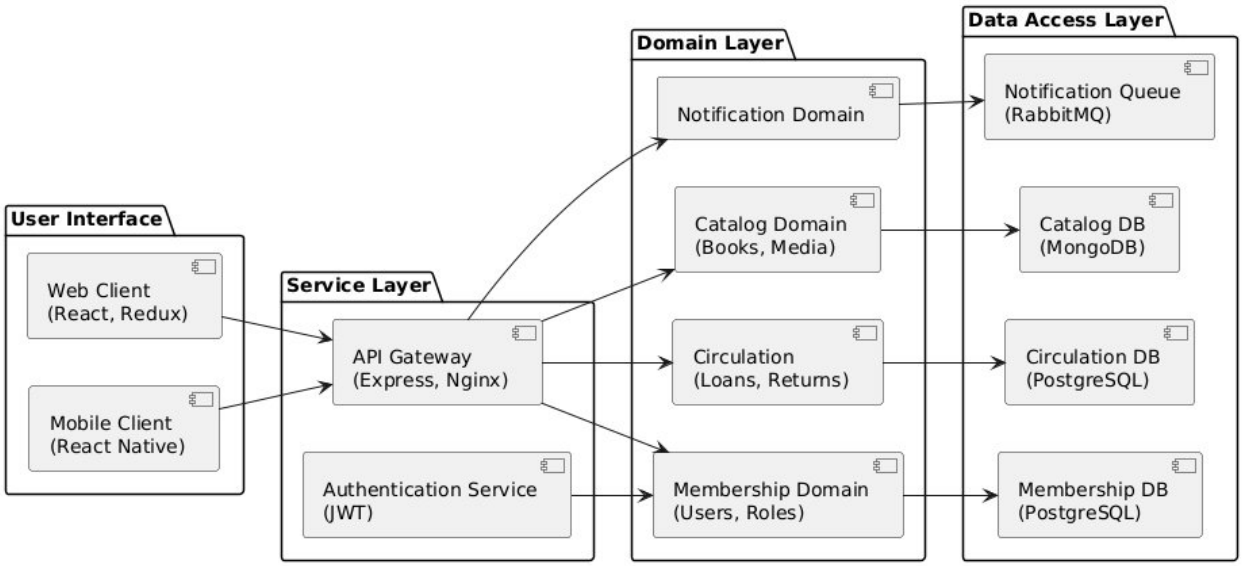
Frontend Application

- Ndërfaqe responsive e bazuar në web, e zhvilluar me React.
- Dizajn i përqendruar te përdoruesi me qasje të lehtë në funksionalitetet kryesore.

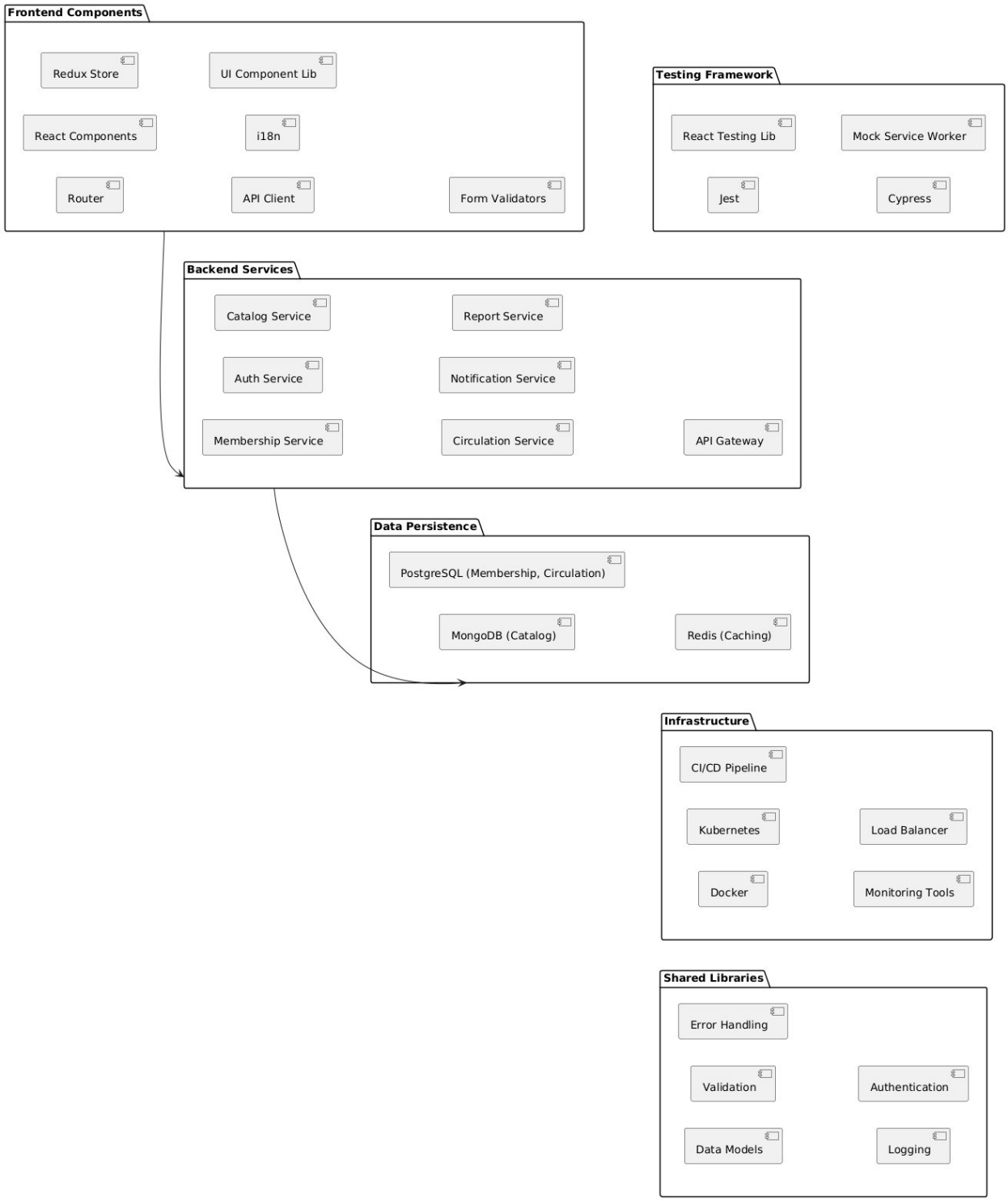
Këto shërbime komunikojnë përmes API-ve RESTful për ndërveprimet sinkrone dhe përmes një sistemi mesazhesh (message broker) për komunikim asinkron.

3.3 Architecture Diagrams

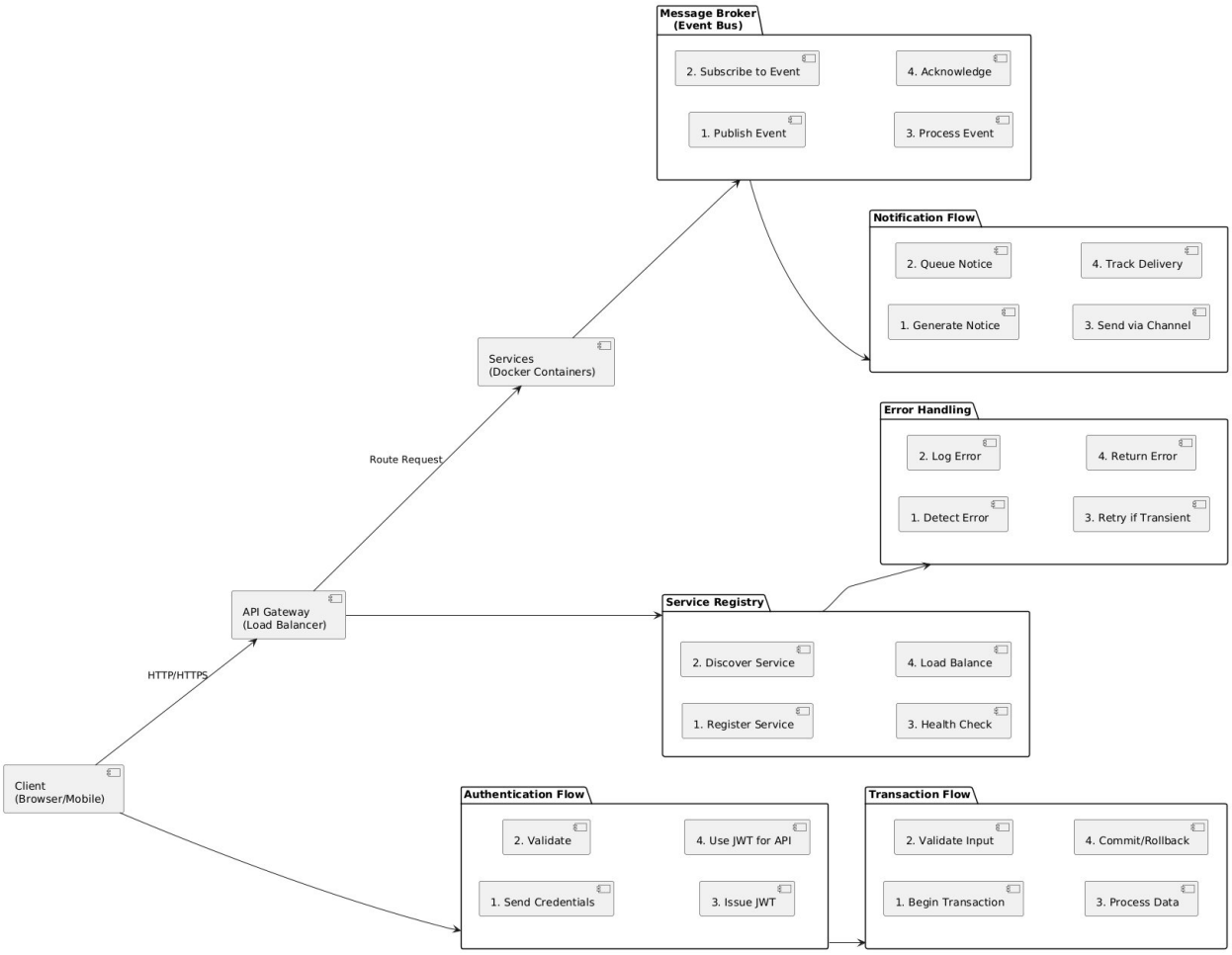
Modeli 4+1: Pamja Logjike (Blend Kqiku)



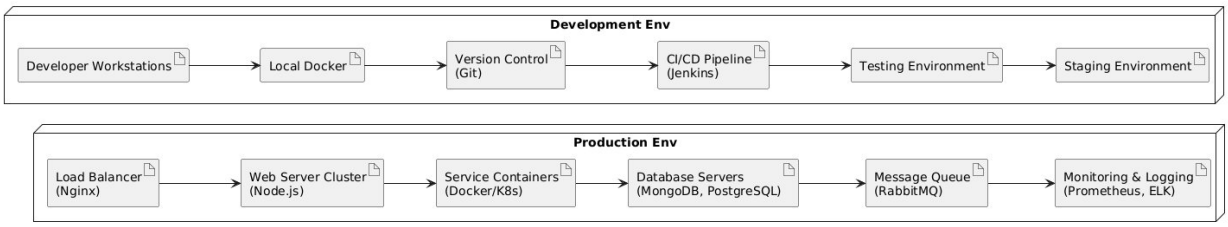
Modeli 4+1: Pamja e Zhvillimit (Rigon Kadriu)



Modeli 4+1: Pamja e Procesit (Aldin Shabani)

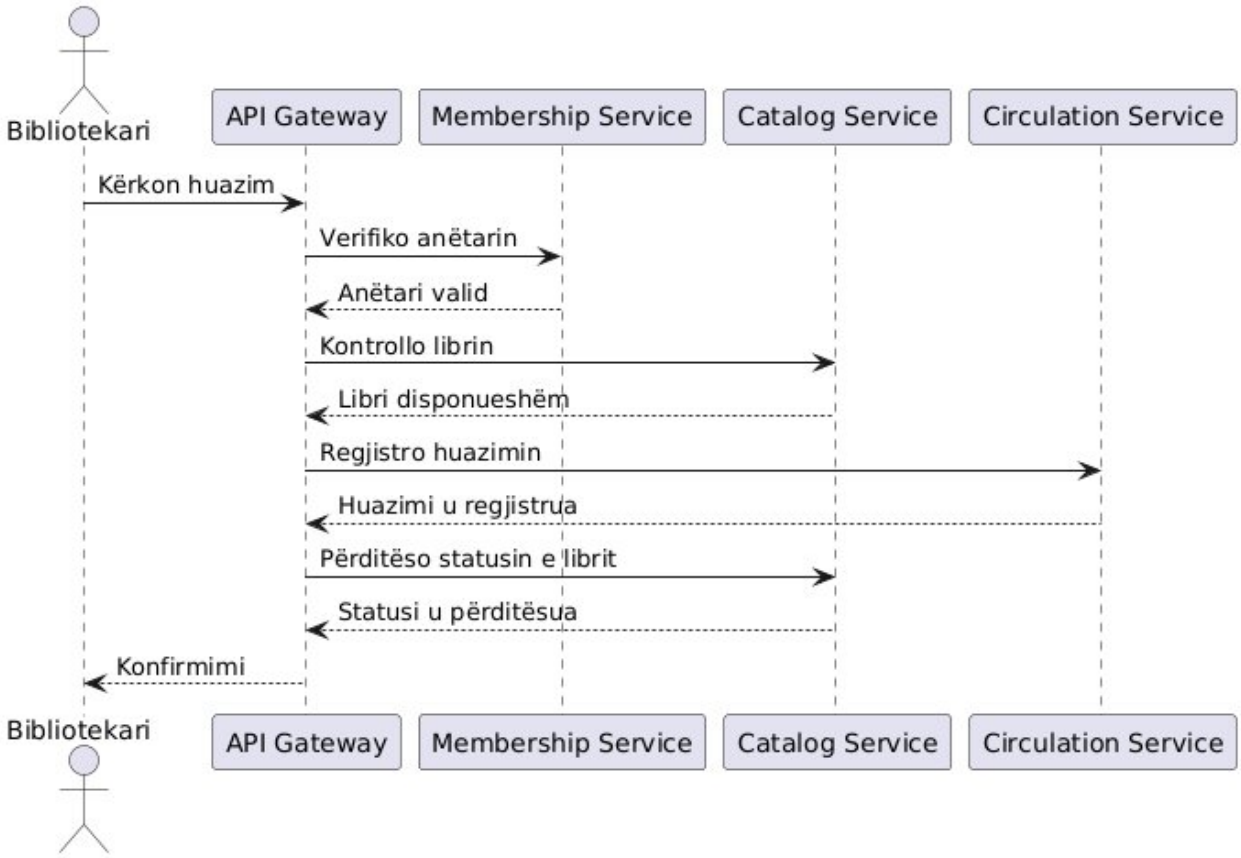


Modeli 4+1: Pamja Fizike (Aid Aliu)



3.4 Behavioral View

Diagrami i Sekuencës: Procesi i Huazimit të Librit



3.5 Trade-offs & Challenges

Trade-offs

Kompleksiteti vs Shkallëzueshmëria Arkitektura e mikroshërbimeve ofron shkallëzueshmëri të lartë, por me koston e rritjes së kompleksitetit të sistemit. Vendosëm të pranojmë këtë kompleksitet shtesë për të përfituar nga avantazhet e shkallëzueshmërisë dhe modularitetit.

Performanca vs Konsistenca Duke përdorur komunikim asinkron për disa operacione, ne përmirësojmë performancën e sistemit, por futim sfidat e konsistencës eventuale. Kemi zbatuar mekanizma për të minimizuar ndikimin e kësaj konsistence të vonuar në përdoruesit.

Teknologjitë e Specializuara vs Mirëmbajtja Përdorimi i teknologjive të ndryshme për shërbime të ndryshme (NoSQL për katalog, SQL për huazime) optimizon performancën për secilin rast, por rrit kompleksitetin e mirëmbajtjes dhe kërkon ekspertizë më të gjerë.

Fleksibiliteti i Zhvillimit vs Standardizimi Lejimi i ekipeve të përdorin teknologji të ndryshme për shërbime të ndryshme rrit fleksibilitetin dhe shpejtësinë e zhvillimit, por redukton standardizimin e kodit dhe praktikat e përbashkëta.

Challenges

Menaxhimi i Transaksioneve të Shpërndara Proceset që përfshijnë shumë shërbime (p.sh., huazimi i librit) kërkojnë menaxhim të kujdesshëm për të siguruar konsistencën e të dhënave. Kemi implementuar një qasje të bazuar në Saga për të menaxhuar transaksionet e shpërndara.

Testimi End-to-End Testimi i plotë i sistemit bëhet më i komplikuar me arkitekturën e mikroshërbimeve. Kemi zhvilluar një strategji të testimit që kombinon testet e njësisë, testet e integrimit dhe testet end-to-end për të siguruar cilësinë e sistemit.

Monitorimi dhe Diagnostikimi Identifikimi dhe diagnostikimi i problemeve në një sistem të shpërndarë është kompleks. Kemi implementuar një sistem të centralizuar të logimit dhe monitorimit për të lehtësuar zbulimin dhe zgjidhjen e problemeve.

Migrimi nga Sistemi Ekzistues Kalimi nga sistemi monolitik ekzistues në arkitekturën e re të mikroshërbimeve paraqet sfida të migrimit të të dhënave dhe operimit paralel. Kemi zhvilluar një strategji të migrimit gradual për të minimizuar rreziqet.

3.6 Implementation Considerations

Teknologjitë Kryesore

Frontend:

- **React.js**: Për ndërfaqen e përdoruesit
- **Redux**: Për menaxhimin e gjendjes së aplikacionit
- **Material UI**: Për komponentët e UI
- **Axios**: Për komunikimet HTTP me backend

Backend:

- **Node.js & Express**: Për zhvillimin e API-ve
- **NestJS**: Për mikroshërbimet që kërkojnë struktura më të avancuara
- **JWT**: Për autentifikim të sigurt
- **Socket.io**: Për komunikimet në kohë reale

Bazat e të Dhënave:

- **MongoDB**: Për katalogimin e librave dhe materialeve
- **PostgreSQL**: Për të dhënat transaksionale (anëtarësia, huazimet)
- **Redis**: Për caching dhe sesionet

DevOps & Infrastrukturë:

- **Docker & Kubernetes**: Për kontejnerizim dhe orkestrimin e shërbimeve
- **Jenkins**: Për CI/CD pipeline
- **Prometheus & Grafana**: Për monitorim
- **ELK Stack**: Për menaxhimin e logeve

Pikat e Integrimit

API Gateway: Shërben si pikë hyrëse e vetme për të gjitha kërkesat nga klientët, duke lehtësuar menaxhimin e sigurisë, limitimin e kërkesave dhe drejtimin e kërkesave te shërbimet e duhura.

Message Broker (RabbitMQ): Mundëson komunikimin asinkron ndërmjet shërbimeve, duke përmirësuar shkëputjen dhe shkallëzueshmërinë.

Service Discovery (Consul): Lejon shërbimet të gjejnë dhe komunikojnë me njëra-tjetrën në mënyrë dinamike, duke lehtësuar shkallëzimin dhe resilieencën.

External APIs:

- Integrim me sisteme pagese për gjobat
- Integrim me sistemin e email-it për njoftime
- API për bibliotekat partnere për huazime ndërbibliothekare

Konsiderata të Implementimit

Strategjia e Lansimit: Planifikojmë një lansim gradual, duke filluar me modulet më pak kritike (p.sh., katalogimin) dhe duke avancuar gradualisht drejt funksionaliteteve më kritike (p.sh., huazimet). Kjo qasje lejon identifikimin dhe zgjidhjen e hershme të problemeve.

Migrimi i të Dhënave: Të dhënat ekzistuese do të migrojnë duke përdorur një kombinim të tools të automatizuara dhe validimit manual. Do të ruajmë backup-e të plota dhe do të kryejmë migrime provë para migrimit final.

Siguria: Implementojmë një qasje të sigurisë në thellësi, duke përfshirë:

- HTTPS për të gjitha komunikimet
- Autentifikim me JWT dhe refresh tokens
- Autorizim të bazuar në role
- Validim i inputeve në të gjitha nivelet
- Auditim i plotë i veprimeve kritike

Performanca: Optimizojmë performancën përmes:

- Caching strategjik në nivele të ndryshme
- Indeksim të duhur të bazave të të dhënave
- Lazy loading për të dhënat jo-esenciale
- Kompresim dhe minifikim të aseteve të frontend-it

4. Architecture Analysis

4.1 Scenario Analysis

Skenari 1: Huazimi i Librit

Arkitektura jonë mbështet efikasitetin e procesit të huazimit përmes integrimit të ngushtë mes shërbimit të anëtarësisë, katalogut dhe huazimeve. API Gateway drejton kërkesën në shërbimet përkatëse, duke verifikuar disponueshmërinë e librit dhe statusin e anëtarit në mënyrë paralele. Transaksioni ruhet në bazën e të dhënave PostgreSQL për të siguruar integritetin e të dhënave, ndërsa një ngjarje publikohet në message broker për përditësimin asinkron të statusit të librit dhe gjenerimin e njoftimeve.

Skenari 2: Kërkimi i Avancuar në Katalog

Arkitektura e bazuar në MongoDB për katalogimin mundëson kërkime të shpejta dhe fleksible në koleksionin e bibliotekës. Indekset e optimizuara dhe struktura e dokumenteve mbështesin kritere të ndryshme kërkimi (titull, autor, kategori, etj.) me performancë të lartë. Sistemi i caching-ut redukton ngarkesën në bazën e të dhënave për kërkesat e përsëritura. API Gateway menaxhon kërkesat me volum të lartë përmes mekanizmave të limitimit dhe load balancing.

Skenari 3: Gjenerimi i Raporteve Analitike

Shërbimi i raporteve mbështet gjenerimin e raporteve komplekse duke përdorur një kombinim të qarqeve të specializuara të të dhënave dhe procesimit asinkron për raporte të rënda. Për raporte në kohë reale, sistemi përdor kueri të optimizuara direkt në bazat e të dhënave përkatëse. Për raporte periodike ose të planifikuara, një sistem i bazuar në ngjarje aktivizon gjenerimin e raporteve në orare të paracaktuara, duke reduktuar ngarkesën gjatë orareve të pikut.

4.2 Risks Revisited

Risku 1: Integriteti i të Dhënave

Statusi: Zbutur **Zgjidhja:** Kemi implementuar një strategji të plotë migrimi me validim të automatizuar të të dhënave dhe verifikim manual për të dhënat kritike. Gjithashtu, sistemet e vjetra dhe të reja do të operojnë paralelisht për një periudhë tranzicioni, duke lejuar verifikimin e kryqëzuar të të dhënave. Backup-et e plota dhe të rregullta sigurojnë aftësinë për të rikuperuar të dhënat në rast problemesh.

Risku 2: Rezistenca e Përdoruesve

Statusi: Në proces zbutjeje **Zgjidhja:** Bibliotekarët janë përfshirë në fazën e dizajnimit të UI/UX, duke siguruar që ndërfaqja të jetë intuitive dhe të përputhet me flukset e punës ekzistuese. Plani i trajnimit përfshin sesione praktike dhe materiale të detajuara. Një ekip mbështetës do të jetë i disponueshëm gjatë periudhës së tranzicionit për të ndihmuar përdoruesit dhe për të adresuar shqetësimet e tyre.

Risku 3: Problemet e Performancës

Statusi: Zbutur **Zgjidhja:** Testet e ngarkesës kanë simuluar situata me 3 herë më shumë përdorues se përdorimi maksimal i pritur, dhe sistemi ka performuar brenda parametrave të përcaktuar. Optimizimi i bazës së të dhënave, strategjitë e caching-ut dhe arkitektura e shkallëzueshme e shërbimeve sigurojnë që sistemi të mund të përballojë rritjen e përdorimit pa degradim të performancës. Monitorimi i vazhdueshëm do të identifikojë çdo problem të mundshëm para se të ndikojë në përdoruesit.

Risku 4: Kompleksiteti i Sistemit

Statusi: Pjesërisht i zbutur **Zgjidhja:** Kompleksiteti i brendshëm i arkitekturës së mikroshërbimeve është izoluar nga përdoruesit fundorë përmes një ndërfaqeje të thjeshtë dhe intuitive. Dokumentimi i plotë për administratorët dhe zhvilluesit lehtëson mirëmbajtjen e sistemit. Megjithatë, pranohet se mirëmbajtja dhe operimi i sistemit do të kërkojë ekspertizë teknike më të lartë krahasuar me sistemin e mëparshëm monolitik.

5. Conclusion

Përmbledhja e mëposhtme është një version i redaktuar i tekstit tuaj, me përmirësime në rrjedhshmëri, koherencë dhe stil profesional të gjuhës shqipe:

Përmbledhje e Pikave Kryesore

Projekti i Sistemit të Menaxhimit të Bibliotekës ka adresuar me sukses nevojat kyçe për automatizimin e proceseve bibliotekare, duke përdorur një arkitekturë moderne dhe të shkallëzueshme. Nëpërmjet një qasjeje të bazuar në mikroshërbime, sistemi ofron:

- **Fleksibilitet** në zhvillimin dhe shkallëzimin e moduleve individuale;
- **Siguri të avancuar**, falë autentifikimit me JWT, enkriptimit të të dhënave dhe kontrollit të hollësishëm të aksesit;
- **Efikasitet operacional**, nëpërmjet automatizimit të proceseve si huazimi, kthimi dhe katalogimi i librave;
- **Përputhshmëri me standardet ndërkombëtare bibliotekare dhe rregulloret e privatësisë (GDPR)**.

Zgjedhjet teknologjike (React, Node.js, MongoDB, PostgreSQL) dhe arkitekturore (mikroshërbime, API Gateway, message broker) rezultuan të justifikuara, duke siguruar një balancë të qëndrueshme midis performancës, shkallëzueshmërisë dhe mirëmbajtjes.

Mësime të Nxëna gjatë Hartimit të Arkitekturës

Kompleksiteti i transaksioneve të shpërndara: Menaxhimi i konsistencës së të dhënave në një arkitekturë të shpërndarë kërkon qasje të sofistikuara, si *Saga pattern*, dhe prioritizim të kujdesshëm të kërkesave të biznesit.

Rëndësia e monitorimit: Implementimi i një sistemi të centralizuar për logim dhe monitorim (ELK Stack, Prometheus) u tregua thelbësor për diagnostikimin dhe mirëmbajtjen efektive të sistemit.

Testimi i hershëm dhe i vazhdueshëm: Integrimi i një pipeline CI/CD me testim automatik (Jest, Cypress) ndihmoi në uljen e rrezikut të regresioneve dhe përshpejtoi ciklin e zhvillimit.

Komunikimi me stakeholderët: Përfshirja aktive e bibliotekarëve gjatë dizajnimit të UI/UX zvogëloi rezistencën ndaj ndryshimeve dhe garantoi përputhshmëri me nevojat reale të përdoruesve fundorë.

Reflektim i Ekipit

Ekipi bashkëpunoi ngushtësisht, duke shpërndarë detyrat sipas fushave të ekspertizës:

Blend Kqiku: drejtoi dizajnin e arkitekturës logjike dhe integrimin ndërmjet shërbimeve;

Rigon Kadriu: u fokusua në optimizimin e bazës së të dhënave dhe zbatimin e strategjive të sigurisë;

Aldin Shabani: zhvilloi mekanizmat e komunikimit asinkron dhe sistemin e njoftimeve;

Aid Aliu: ndërtoi infrastrukturën DevOps dhe zgjidhjet për skalim.

Pavarësisht sfidave teknike dhe kufizimeve kohore, koordinimi efektiv dhe përdorimi i praktikave moderne (Docker, Kubernetes) mundësuan realizimin e një sistemi të qëndrueshëm dhe të gatshëm për përdorim të gjerë.

Perspektivat e Ardhshme

Sistemi është projektuar për të evoluar në përputhje me nevojat në rritje të bibliotekës. Prioritetet e ardhshme përfshijnë:

Shtimin e një moduli për menaxhimin e burimeve digjitale (e-books, audiobooks);

Integrimin e teknologjive të inteligjencës artificiale për rekomandime të personalizuara të librave;

Zgjerimin e sistemit për të mbështetur rrjet bibliotekash shumë-degjëshe.

Ky projekt dëshmon se një qasje e bazuar në mikroshërbime, e kombinuar me një kuptim të thellë të domain-it të biznesit, mund të transformojë proceset tradicionale në zgjidhje moderne, efikase dhe të përputhshme me kërkesat bashkëkohore.

Appendix

A. Team Contribution Table

ID	Name and Surname	Tasks Contributed	Hours Worked
222367309	Rigon Kadriu	U fokusua në optimizimin e bazës së të dhënave dhe zbatimin e strategjive të sigurisë	78
222367418	Blend Kqiku	Drejtoi dizajnin e arkitekturës logjike dhe integrimin ndërmjet shërbimeve	78
222364730	Aldin Shabani	Zhvilloi mekanizmat e komunikimit asinkron dhe sistemin e njoftimeve	78
212257307	Aid Aliu	ndërtoi infrastrukturën DevOps dhe zgjidhjet për skalim.	78

B. Additional Diagrams or References