table of contents

This version of the template contains some help and explanations. It is used for familiarization with arc42 and the understanding of the concepts. For documentation of your own system you use better the *plain* version.

1. Introduction and Goals

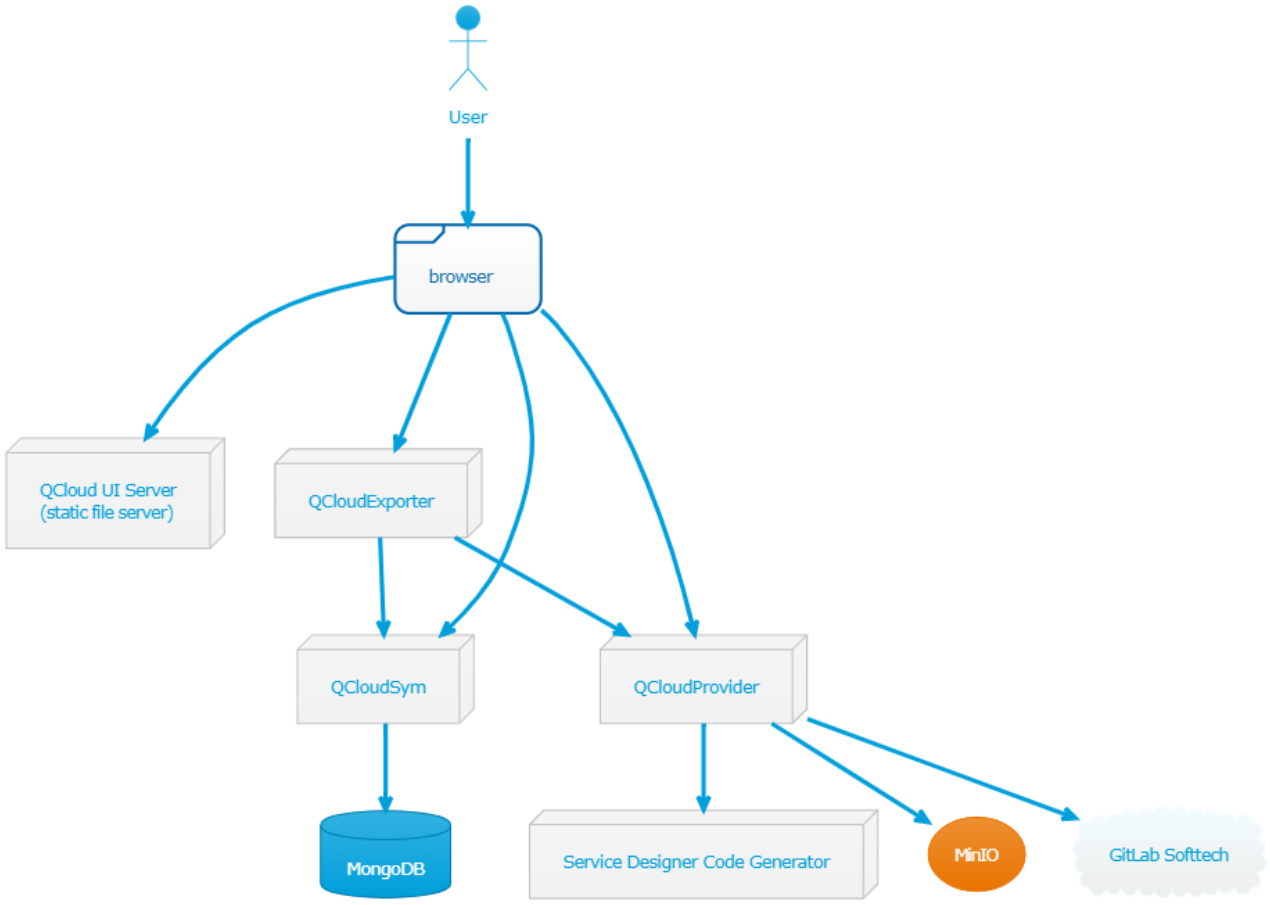
1.1. Definition of Problem or Requirement

Günümüz dünyasında uygulamalar çok karmaşıklaşmış, ekipler büyümüş, dolayısıyla, kaliteli uygulamaları istenen sürede geliştirmek zorlaşmıştır. Diğer taraftan turn-over süreleri kısalmış, uygulama geliştiren ekiplerin on-boarding süreleri ise uzamıştır. Artan rekabet koşulları şirketleri daha büyük ve karmaşık projeleri daha hızlı, daha kaliteli geliştirmeye zorlamaktadır. Daha düşük profildeki kaynaklardan da belli uygulamaları geliştirebilmek şirketlere daha da büyük avantaj sağlayacaktır. Bu koşullar Low Code/No Code geliştirme platformlarını ortaya çıkarmaktadır.

1.2. Definition of Solution

Plateau Low Code, kullanıcıların web, iOS ve android geliştirme bilgisi olmadan ve minimum seviyede programlama bilgisi gerektirerek, ilgili platformda çalışabilecek uygulama geliştirmesini hedefleyen bir tool'dur.

Plateau Low Code web tabanlı bir önyüz geliştirme editörü (Quick editor), süreç editörü (Process editor), bpmn editörü, entity editorü, Symphony BFF, plateau provider, static file server, exporter, MongoDB ve MinIO' dan oluşur.



Typescript, Javascript, NodeJs ve Vue.js kullanılarak geliştirilmiş, tarayıcı üzerinde çalışan, web tabanlı bir tooldur. Çıktılar ekran tariflerini içeren json tipinde modeller, deploy edilebilir zip

dosyası ve gitlab ortamına kaydedilen Plateau uygulama kod ve konfigürasyon dosyalarıdır.

Plateau Low Code hem Plateau framework' ü üzerinde çalışabilecek uygulama, hem de nodeJs üzerinde Plateau framework' ü olmaksızın çalışabilecek uygulamalar geliştirecek şekilde tasarlanmıştır.

1.2.1. User Profile

Quick üzerinde önyüz geliştirmek için temel TypeScript ve/veya JavaScript bilgisi yeterli olacaktır. Geleneksel önyüz web, ios, android bilgisi gerekmemektedir. Karmaşık kullanım durumları, deneyim gerektirebilir. Kullanılan komponent kütüphanelerinin dökümantasyonlarını okuyabilecek seviyede ingilizce bilgisi gereklidir.

Süreç geliştirebilmek için temel süreç ve bpmn bilgisi yeterlidir. İleri düzey süreç geliştirme ihtiyacı olduğunda plateau süreç ekibi tarafından hazırlanmış dökümantasyon ve videolara bakılması faydalı olacaktır.

Entity' leri oluşturabilmek için temel entity relationship kavramlarını bilmek yeterlidir. Entity' ler üzerinde karmaşık işlemler gerektiğinde backend developer desteğine ihtiyaç duyulabilir.

1.2.2. Developer Profile

Plateau Low Code uygulamasını geliştirmek için JavaScript, TypeScript, NodeJs, Vue.js, temel DevOps ve Plateau bilgisi yeterli olacaktır. Plateau kod generation tarafında geliştirme yapacak kişilerin temel Java bilmesi gerekecektir.

1.3. Development Requirements | Geliştirme Gereksinimleri

Plateau Low Code, geliştirme zamanında tarayıcı dışında bir bağımlılık içermemeyi hedeflemiştir.

1.4. Stakeholders

Role	Beklenti	Plateau Low Code geliştirme ekibi
Plateau Low Code' un geliştirilmesinden sorumlu ekiptir.	Citizen developer’lar	Plateau Low Code ile uygulama geliştiren ekiplerdir.
Proje ekipleri	Plateau Low Code ile geliştirilmiş projelerin ayarları/deployment’ı/tasarımı vb. konuları organize eden ekiptir.	Tasarım ekibi

1.5. Quality Goals {new}

Goal	Description
Usability	Eklenen feature'ların hedef (citizen) developer kitlesi tarafından kullanılabilir olması sağlanmalıdır.
Usability	Sade ve anlaşılabilir olmalıdır. Kurulum, öğrenme ve yaygınlaşma kolay olmalıdır.
Portability	Yapılan uygulamayı bir model içerisinde tanımlayıp, bu modelin hedef Plateau veya NodeJs ortamlarında çalışabilir olması sağlanmalıdır.
Compatibility	Oluşturulan modellerin ilgili platformlardaki runtime'lar güncellendikçe çalışmaya devam etmesi sağlanmalıdır.
Functional Suitability	Uygulama verilerinin temel ihtiyaçları (CRUD) karşılanmalıdır.
Functional Suitability	Tekli onay, ikili onay gibi temel onay süreçlerinin tanımlanabilmesi sağlanmalıdır.
Performance Efficiency	Kaynak kullanımı konusunda verimli olmalıdır.
Maintainability	Modüler ve test edilebilir olmalıdır.

Performance efficiency	Score
Time behaviour	4
Resource utilization	4
Capacity	4

Compatibility	Score
Co-existence	4
Interoperability	4

Usability	Score
Appropriateness recognizability	3
Learnability	3
Operability	3
User error protection	3
User interface aesthetics	4
Accessibility	4

Reliability	Score
Maturity	2

Reliability	Score
Availability	3
Fault tolerance	3
Recoverability	4

Security	Score
Confidentiality	4
Integrity	4
Non-repudiation	3
Accountability	3
Authenticity	3

Maintainability	Score
Modularity	4
Reusability	4
Analysability	4
Testability	4
Modifiability	3
Authenticity	3

Portability	Score
Adaptability	3
Installability	3
Replaceability	3

2. Architecture Constraints

2.1. Overview

2.2. Contstaints & Solutions {new}

Constraints	Description
ARCH-01-C	Sunulan çözümler generic olmalı ve proje bağımlılığı içermemelidir.
ARCH-01-S	Teknoloji stack seçiminde third-party ürün ve kütüphanelerden mümkün olduğunca kaçınılmıştır.
ARCH-02-C	Geliştirmeler, ürünün vizyonu (Low Code) ile paralellik göstermelidir. Kullanımı kolay olmalıdır.
ARCH-02-S	Flow editör geliştirilecektir.
ARCH-03-C	Plateau Low Code geliştirme zamanında, (ARCH-02 ile paralel olacak şekilde) tarayıcı dışında bir bağımlılık içermemeyi hedeflemiştir. Kullanıcı bilgisayarına herhangi bir kurulum gerektirmemelidir.
ARCH-03-S	Web uygulaması olarak geliştirilmektedir.
ARCH-04-C	Low code ekibi dışındaki geliştiricilerin extend edebilmelerine olanak sağlanmalıdır.
ARCH-04-S	Roadmap kapsamında marketplace desteği sağlanacaktır.

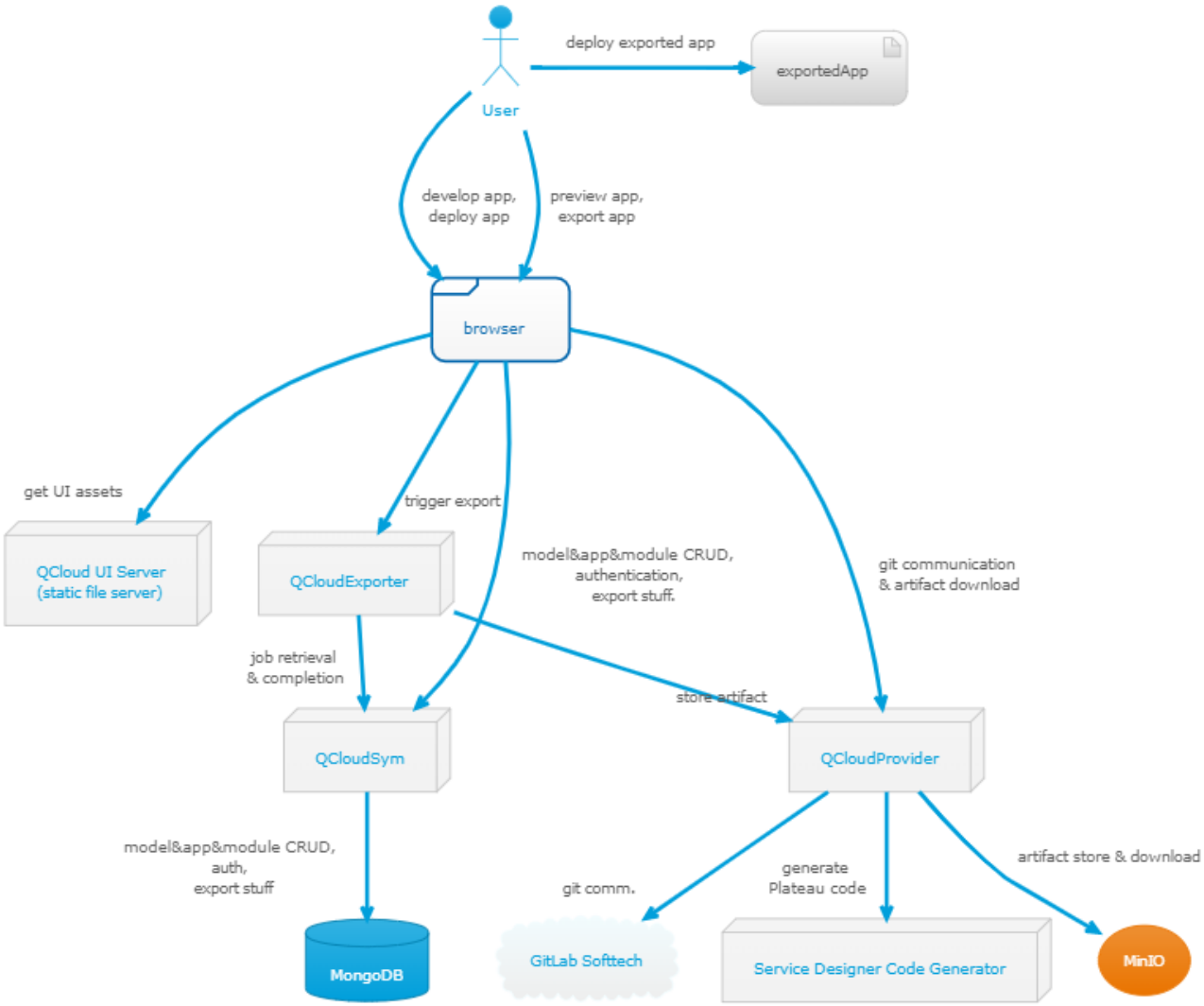
2.3. Performance Goals

Metrics	Description
Name(s) of the service(s)	UI: https://studio.onplateau.com, Exporter: https://studio.onplateau.com/exporter, Sym: https://studio.onplateau.com/sym, Provider: https://studio.onplateau.com/provider.
Number of threads(users)	UI: Static file server, others 1 thread per app
Ramp-up Period (seconds)	Exporter: istekle jobları çekmeye başlar, bir thread çalışır (CPU bound işi yoktur, disk I/O ve network I/O işi yapar, async çalışır) diğerleri NA

Metrics	Description
Duration (seconds)	Exporter: tetiklenme sonrası joblar bitene kadar çalışır, işler bitince çalıştığı platform tabanlı uyumaya geçer, diğerleri NA
Loop Count	Exporter: job olduğu sürece job çekme loop' u devam eder, diğerleri NA
TPS	<10 (ileride artabilir)
Minimum CPU & RAM	Ölçülmedi
Max Response Time (miliseconds)	500 ms
Max Service Up Time (miliseconds)	5000 ms
UI	
	static file server
sym	
createModel	POST /createModel
createModule	POST /createModule
createApplication	POST /createApplication
createLog	POST /createLog
getModelInfo	POST /getModelInfo
getModel	POST /getModel
getModelBody	POST /getModelBody
updateModel	POST /updateModel
updateModule	POST /updateModule
updateModule	POST /updateModule
updateFolder	POST /updateFolder
updateApplication	POST /updateApplication
deleteModel	POST /deleteModel
deleteModule	POST /deleteModule
deleteFolder	POST /deleteFolder
deleteApplication	POST /deleteApplication
duplicateModel	POST /duplicateModel
getUserDomains	POST /getUserDomains
login	POST /login
listApplications	POST /listApplications
listModules	POST /listModules
listModuleChildItems	POST /listModuleChildItems

Metrics	Description
listModelBodies	POST /listModelBodies
getApplicationDetails	POST /getApplicationDetails
getApplication	POST /getApplication
logout	POST /logout
getUserInfobyToken	POST /getUserInfobyToken
getExportTypes	POST /getExportTypes
getExportSelectionData	POST /getExportSelectionData
listModelHistories	POST /listModelHistories
getModelHistory	POST /getModelHistory
createExportJob	POST /createExportJob
createDeployJobs	POST /createDeployJobs
listExportJobs	POST /listExportJobs
listExportJobSteps	POST /listExportJobSteps
listExportJobStepLogs	POST /listExportJobStepLogs
listExportJobArtifacts	POST /listExportJobArtifacts
listTreeviewItems	POST /listTreeviewItems
updateApplicationSettings	POST /updateApplicationSettings
attachModuletoApplication	POST /attachModuletoApplication
detachModuleFromApplication	POST /detachModuleFromApplication
getApplicationUISettings	POST /getApplicationUISettings
getTenantDefinition	POST /getTenantDefinition
setRootScreen	POST /setRootScreen
putObjectInStore	POST /putObjectInStore
PROVIDER	
getProjectsByGroupId	GET /getProjectsByGroupId
getProjectsOfProduct	GET /getProjectsOfProduct
getGroupNamesWithId	GET /getGroupNamesWithId
checkRepoHasService	POST /checkRepoHasService
EXPORTER	
go	POST /go

3. System Scope and Context



4. Solution Strategy

4.1. Overview

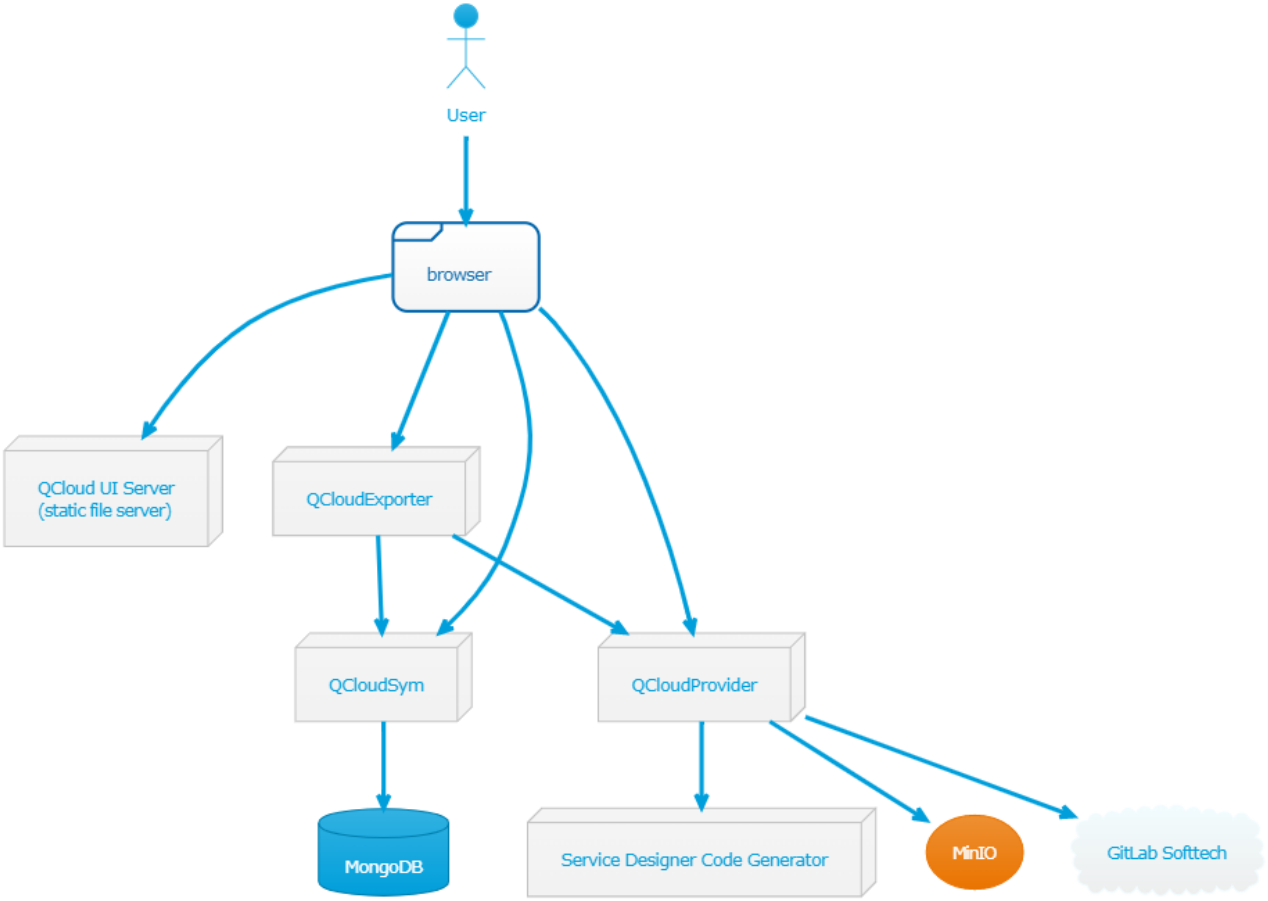
Typescript, Javascript, NodeJs ve Vue.js kullanılarak geliştirilmiş, tarayıcı üzerinde çalışan, web tabanlı bir tooldur. Çıktılar ekran tariflerini içeren json tipinde modeller, deploy edilebilir zip dosyası ve gitlab ortamına kaydedilen Plateau uygulama kod ve konfigürasyon dosyalarıdır.

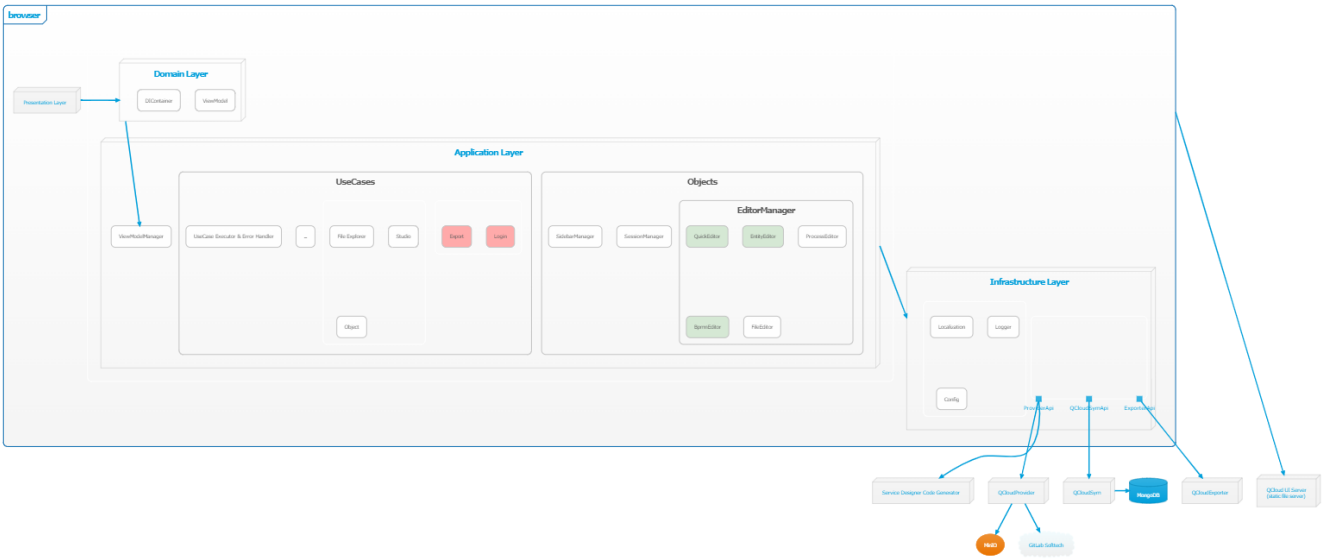
Plateau Low Code hem Plateau framework' ü üzerinde çalışabilecek uygulama, hem de nodeJs üzerinde Plateau framework' ü olmaksızın çalışabilecek uygulamalar geliştirecek şekilde tasarlanmıştır.

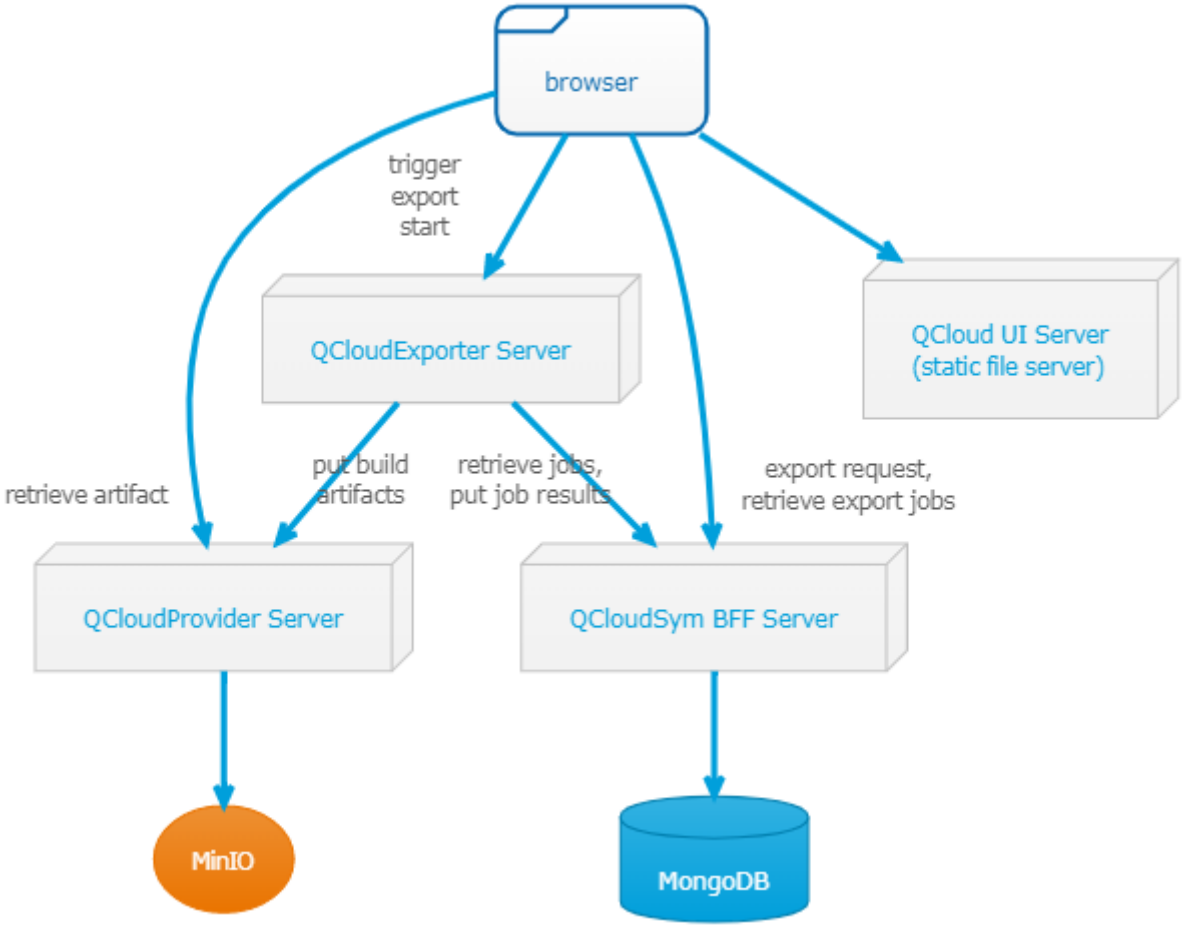
4.2. Technology at a Glance

Technology	**Description**
UI	TypeScript, JavaScript, Vue.js, html, css ile yazıldı.
Mobil	NA
Gateways or BFF	QCloud-Sym, Qloud-Provider, QCloud-Exporter, üçü de TypeScript, NodeJs ile yazıldı.
Security	Şifreler DB' de tutulmaktadır. JWT token ile authentication yapılmaktadır. Authorization roadmap kapsamında geliştirilecek.
Communication	Rest API
Cache	no server cache
Databases	MongoDB
DB scripts	no db scripts, created during runtime
DB ORM	no orm
Languages	TypeScript, JavaScript, Vue.js, html, css, Java
Messaging	NA
Others	Camunda BPMN

5. Building Block View (Context and Scope)







5.1. Service & Caching Architecture {new}

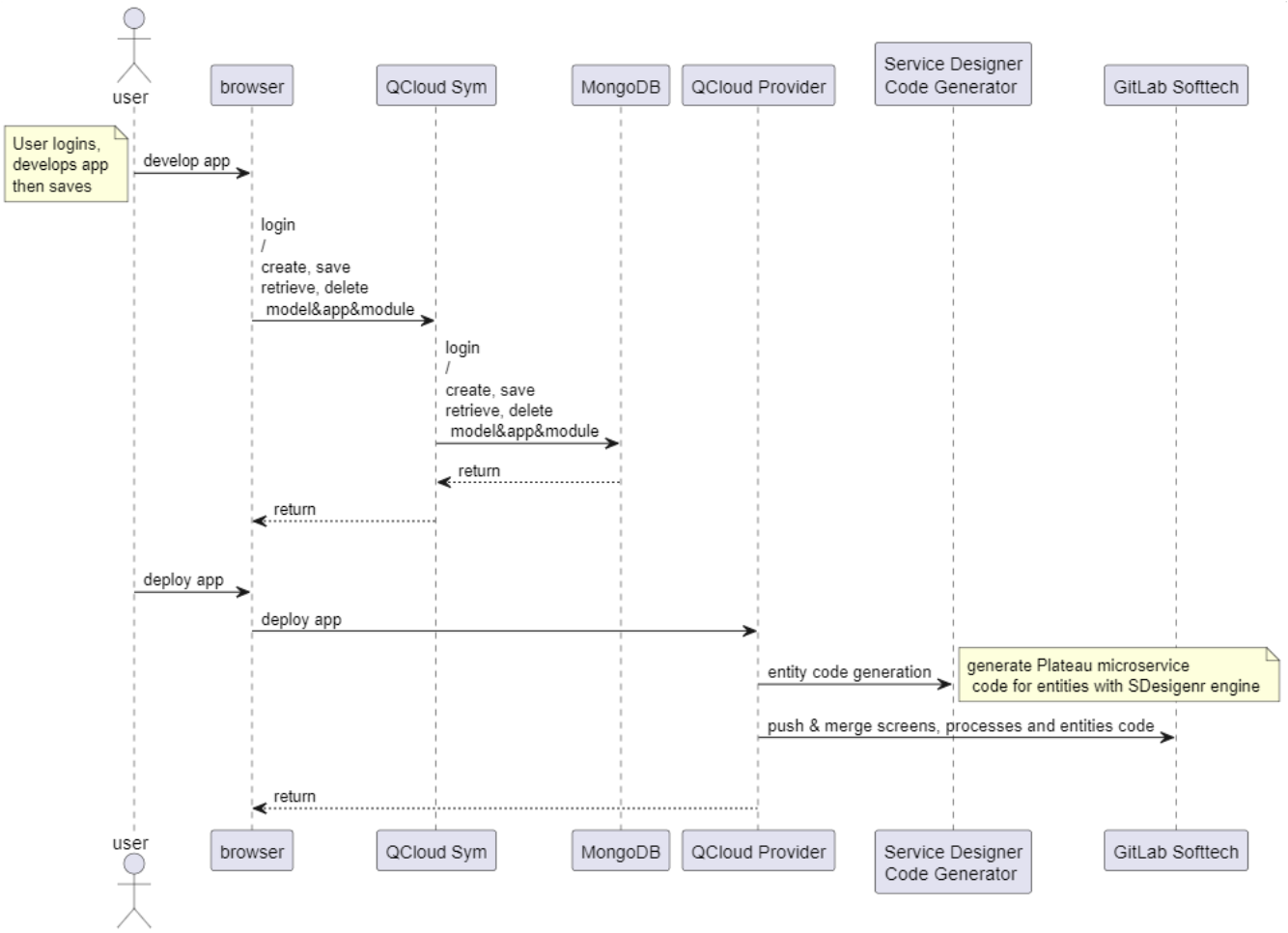
NA

6. Runtime View

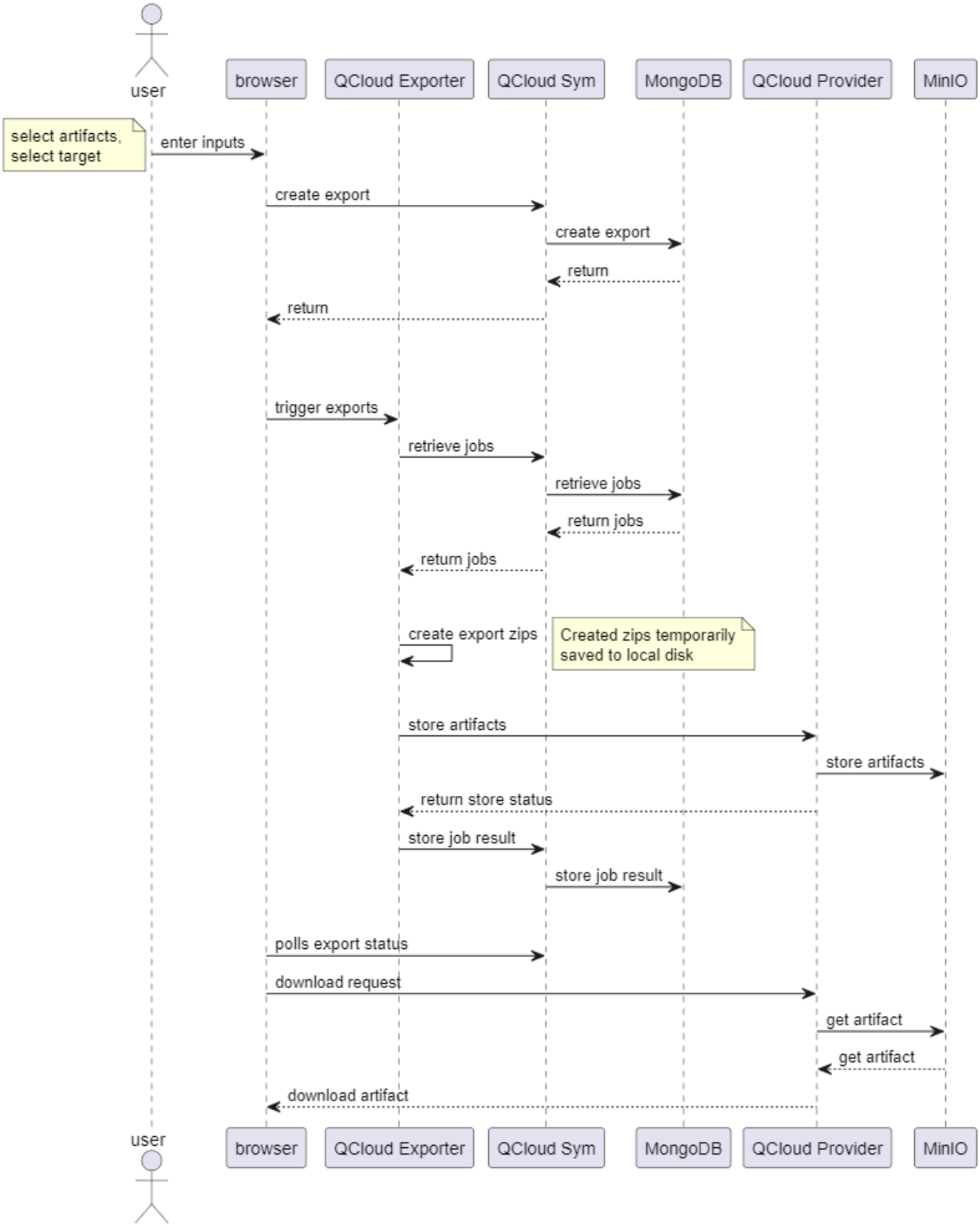
6.1. Model Change

Kullanıcı (Citizen Developer) browser' dan Plateau Low Code IDE' ye login olur. Application ve altındaki modulleri yaratıp, ekran, süreç, entity geliştirmelerini yapar. Bu geliştirmeler model halinde QCloudSym' e iletilir, oradan da MongoDB' ye yazılır.

Kullanıcı browser' da deploy butonuna basar, istek QCloud Provider' a iletilir, eğer model' de entity CRUD geliştirmesi varsa Service Designer Code Generator ile ilgili kodlar üretilir. Süreç geliştimesi varsa bpmn ve yaml kodları üretilir. UI quick model dosyaları ile birlikte tüm bu kodlar gitlab' a push' lanır, merge edilir. Merge ile Jenkins' te otomatik olarak ilgili ortama deploy başlatılır.



6.2. Export

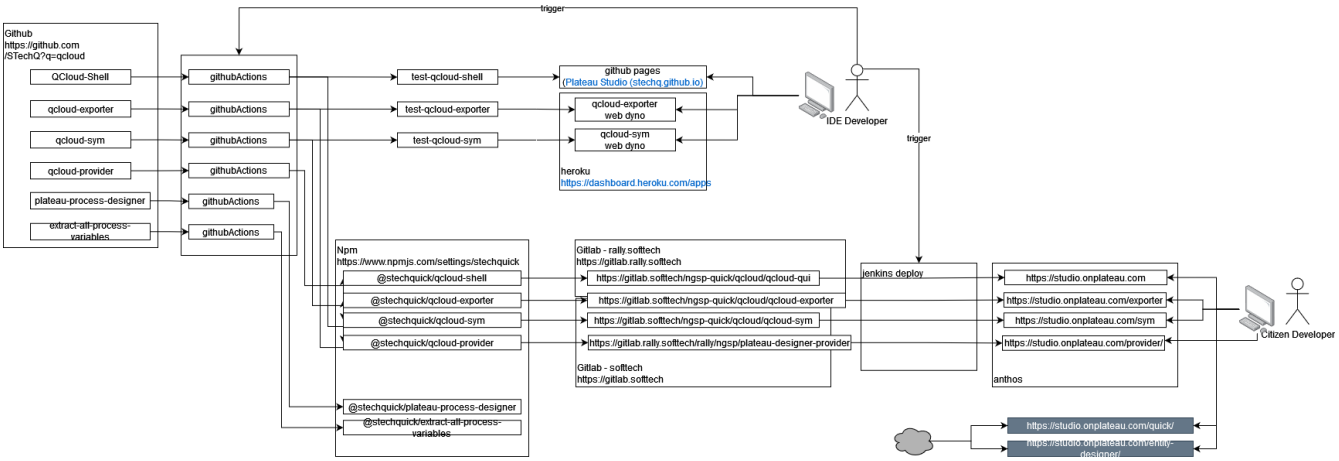


Kullanıcı browser' dan export etmek istediği ekran, süreç vb, seçip export butonuna basar. Export başlatma talebi QCloud Sym' a iletilir ve MongoDB' ye kaydedilir.

Kullanıcı browser' da bekleyen export taleplerinin gerçekleştirilmesi için Trigger Exports butonuna basar. Talep QCloud Exporter' a iletilir. QCloud Exporter, QCloud Sym aracılığıyla MongoDB' den export job' larının listesini alır. Ardından her bir job için local diskte export zip oluşturur, QCloud Provider aracılığıyla MinIO' ya kaydeder ve QCloud Sym aracılığıyla MongoDB' deki export statüsünü günceller.

Browser QCloud Sym' den export statülerini poll eder. Başarılı tamamlanmış export' lardan hangisini download etmek istiyorsa QCloud Provider aracılığıyla ilgili zip dosyasını download eder.

7. Deployment View



8. Cross-cutting Concepts

8.1. Domain concepts

8.1.1. User Experience concepts (UX)

Material UI uyumlu önyüz geliştirilmektedir.

8.1.2. Internationalization/Localization

Her bir dil bazında key-value dosyaları ile dil desteği sağlanmaktadır. Ancak kullanıcı runtime' da dil tercihi yapamamaktadır. Default dil İngilizcedir. Tüm localization client-side yapılmaktadır.

8.2. Safety and security concepts

8.3. Architecture and design patterns

Clean (Onion) architecture mimarisine uygun geliştirilmektedir. Presentation katmanında MVVM pattern' i kullanılmıştır. Bunun yanısıra use-case strategy deseni kullanılmıştır.

8.3.1. Initialization and reset.

Config bilgileri, yardımcı uygulamaların ve editörlerin adreslerinin ortamlara göre tutulduğu bir app settings yapısı kurgulanmıştır. Ortamlara göre tanımlar uygulamanın kodunda tutulmaktadır. Server bileşenleri, hangi ortam bilgisini kullanacağını ortam değişkeni veya run parametresi yardımıyla seçmektedir. Önyüz bileşeni ise çalışmakta olduğu url' den seçmektedir.

8.3.2. Memory management.

MVVM pattern' ine uygun merkezi bir obje (ViewModel nesnesi) üzerinden yönetilmektedir. Use-case' ler viewModel' ı manipüle edip bağlı olan view'ların değişmesine yol açmaktadır.

8.4. "Under-the-hood"

8.4.1. Persistency

MongoDB kullanılmaktadır. Bir merkezi veritabanının yanısıra her bir domain bazında ayrı birer veritabanı tutulmaktadır.

8.4.2. Transaction Handling

NA

8.4.3. Session Handling

Oturum yönetimi için JWT token kullanılmaktadır. Kullanıcının username, tenant bilgileri ile

session yönetilmektedir.

8.4.4. Communication and Integration

Bileşenler TCP/IP üzerinden REST ve socket çağrılarıyla haberleşmektedir.

8.4.5. Exception and Error Handling

Uygulamanın business akışı use-case' ler ile sağlanmaktadır. Bu use-case bileşenlerini çalıştıran merkezi bir executor bulunmaktadır. Default exception handling' i bu executor sağlanmaktadır. Use-case bazında spesifik exception handler implemente edilebilmektedir.

8.4.6. Parallelization and Thread-handling

Uygulama bileşenleri birer single-thread nodeJs uygulaması olup, concurrency Javascript' in promise/async yapısı ile sağlanmaktadır.

8.4.7. Validation

Önyüz bileşeni için merkezi bir validation yapısı bulunmaktadır. Backend tarafında uygulama kodu içerisinde validasyonlar yapılmaktadır.

8.4.8. Batch

NA

8.4.9. Reporting

NA

8.5. Development Concepts

8.5.1. Build-test-deploy

WebPack ile npm paketleri oluşturmaktadır. Paketleme ve deployment 07 Deployment View başlığı altında detaylı anlatılmıştır.

8.5.2. Code Generation

Entity designer' da oluşturulmuş entity' lerin CRUD işlemleri için Service Designer' in code generation engine' i kullanılmaktadır.

8.5.3. Migration

NA

8.5.4. Configuration

Config bilgileri, yardımcı uygulamaların ve editörlerin adreslerinin ortamlara göre tutulduğu bir app settings yapısı kurgulanmıştır. Ortamlara göre tanımlar uygulamanın kodunda tutulmaktadır. Server bileşenleri, hangi ortam bilgisini kullanacağını ortam değişkeni veya run parametresi yardımıyla seçmektedir. Önyüz bileşeni ise çalışmakta olduğu url' den seçmektedir.

8.6. Operational Concepts

8.6.1. Administration

NA

8.6.2. Management

NA

8.6.3. Disaster-recovery

NA

8.6.4. Clustering

NA

8.6.5. Logging

Her bileşenin (UI, qcloud-sym, qcloud-exporter, qcloud-provider, vb) kendi loglama mekanizması vardır. Hiçbir bileşen merkezi bir loglama sistemine bağlı değildir ancak UI teknik hataları veritabanına beslemektedir.

UI tarafında exception' lar teknik ve business hatası olacak şekilde 2 ana gruba ayrılır. Kullanıcının yanlış kullanımına dair olan hatalar business, diğer hatalar ise teknik hata olarak ele alınır.

8.6.6. Tracing

NA

9. Design Decisions

Timestamp	Context	Decision	Status	Consequences
2021	Editor View	IDE üzerinde editlenen her bir model için ayrı bir editör instance' ı yaratmak yerine her editör tipinden bir instance yaratılıp, içeriği model ile yönetilmesi kararı verildi.	Completed	Her bir editör tipinden sadece bir instance yaratıldığı için aynı anda birden fazla model açan kullanıcılar için memory kullanımı düşürülmüştür.
2021	Editor View	Mevcut editörler, kullanımlarını engellememek için frame içerisinde açılacaktır.	Completed	Kütüphane, css ve global değişken çakışmaları engellenmiş oldu. Single look-feel sağlanamadı. Editörlerin iletişimi ayrı bir messaging mekanizması geliştirilmesi gerekti.
2022	Editor View	Yeni geliştirilecek editörlerin frame bağımlılığı olmaksızın kullanılması için bir mekanizma kurulması.	Completed	Process editör, process design editör ve yeni geliştirilecek editörler (flow editörü vb) ayrı frame yerine internal konumlandırıldı.
2022	Plateau Dependency	IDE geliştiricilerine gerçek bir no code/low code deneyimi sunabilmek adına, plateau altyapısının getirdiği karmaşıklık ve bundan mütevellik yavaşlığı barındırmayan, bağımlılık içermeyen ve IDE yaklaşımına daha uygun bir uygulama yapısının, plateau'nun yanısıra yapılmasına karar verildi.	RoadMap	Plateau bağımlı geliştirmeyi etkilemeyecek bir şekilde App Developer'ın tercihi doğrultusunda kullanılabilecek bir Flow geliştirmesi roadmap'e alındı.
2022	Export	Deploy sistemi üzerinden yapılan entegrasyon yapısı gereği scale etmeyen bir çözüm olacaktı. Bundan dolayı Softtech dışındaki müşterilerin kullanımı için bir Export mekanizması kurgulandı.	Continues	UI için export mekanizması kuruldu. Backend export çalışmaları devam etmektedir.

10. Quality Requirements

10.1. Quality Tree

Category	Quality	Description	Scenario
Usability	Ease of Use	Bir low-code geliştirme ortamı olarak, kolay ve hızlı modelleme ortamı sunmalıdır.	SC1, SC2
	Ease of Learning	Temel özelliklerin öğrenimi, uzun süreli bir hazırlık gerektirmemeli, basit ve sezgisel olmalıdır.	SC3

10.2. Quality Scenarios

Id	Scenario
SC1	Kullanıcıya, bir IDE' den beklenen fonksiyonlar (drag-drop, redo/undo, context menu, treeview, multi-tab, multi-model editing, vb) sunulur.
SC2	Kullanıcıya kolay tanımlanabilir, tekrarlanan model parçaları için wizard' lar sunulur.
SC3	Kolay öğrenilebilmesi ve sezgisel olması açısından context menüler, görsel validasyonlar, vb, sunulur.

11. Risks and Technical Debts

11.1. Technical Risks

Risk	Description
Plateau Deploy	Deploy sürelerinin çok uzun olması müşteri memnuniyetsizliği yaratabilir.
Plateau Dependency	Plateau' da yapılacak yapısal bir değişiklik Plateau Low Code ile üretilecek uygulamanın çalışmamasına neden olabilir.

11.2. Technical Debts

Debt	Description
Exporter node' larının autoscale ayarlanması	Çok fazla export job' u talebi geldiği takdirde bekleme süreleri çok uzayacaktır. Bunun için autoscale edilecek duruma getirilmelidir. Container scaling / child process / cluster gibi alternatifler değerlendirilmelidir.
Internationalization	Kullanıcının dil seçebilme/değiştirebilme desteği sunulmalıdır.
Authorization	Yetki bazlı geliştirme desteği sunulmalıdır. App Developer'ların yetkileri ölçüsünde modelleri ve modülleri değiştirebilme kısıtlamaları ve ayrıca tenant' ların bunu yönetebilmeleri için admin ekranları geliştirilmelidir.
Super-admin ekranları	Tenant ve domain tanımlarının yapılabilmesi için Softtech super-admin ekranları geliştirilmelidir.
SSO	İsteyen tenant'lar için kullanıcıları kendi sistemlerinden (LDAP vb) login edebilme desteği sunulmalıdır.
Sensitive Data	Hassas verilerin ortam değişkenlerinden alınması sağlanmalıdır.

12. Glossary

Term	Definition
Tenant	Sattığımız şirket
Domain	Birbirinden izole duran Low Code ürün geliştirme alanları (Örnek: AHE domaininde AHE ve Softtech tenant' ları geliştirme yapabilir. TURIS domaininde sadece Softtech tenant'ı geliştirme yapabilir.)
Bileşen	IDE yi oluşturan yazılım parçaları (QCloud-Shell, QCloud-exporter, QCloud-sym, QCloud-provider, vb)
Application	Bir domain içerisinde geliştirilmekte olan önyüzlerden biri. Her bir application içerisinde N adet modül bulunabilir.
Modül	Ayrı deploy edilebilen birbiriyle alakalı, servis, önyüz ve process grubudur. Her bir application içerisinde N adet modül bulunabilir.

13. Security

13.1. Deployment and Infrastructure Considerations

1. Do you have a properly documented architecture diagram with a high-level explanation of the parts and a network connectivity diagram showing how different component are placed and secured? which part of the software component is in DMZ cluster etc, specify them.
    - **Yes.**
2. Components required for the application: What is the OS supporting the application, hardware requirement, etc.?
    - **There is no requirement.**
3. Port and Service requirement: An application may communicate with other application as well. Identify which ports and services are required to be open for the application.
    - **There is no requirement apart from HTTPS access to studio.plateau.com .**
4. Component Segregation: Components of the application should be segregated from each other. For example, the application server and database server should not reside in the same machine. Do you consider this in your current architecture?
    - **Each component of the system has its own containers**
        - **UI is a static web site running on its own servers.**
        - **Backend is seperated into 3 parts.**
            - **Sym is the backend of IDE**
            - **Exporter is the backend responsible for doing the export/deploy jobs**
            - **Provider is the backend responsible for file storage(minio) and code generation of entity and gitlab.**
5. Disable clear text protocol: Ports running clear text services like (HTTP, FTP, etc.) should be closed and not used for any part of the application. Do you have a control point for this requirement?
    - **IDE does not use clear text protocol. But we don’t have a control point.**

13.2. Input Validation

1. Do you check if the application is validating the user input or processing the input as it is?
    - **User input is validated both at UI and backend.**
2. Do you check whether there is any vulnerability in the services which a user can bypass the validation?
    - **Even though user can bypass UI validation, validation is also done at the backend layer**
3. As a best practice, validation should be applied on all the layers, i.e. business layer, data layer, etc. Have you considered it from this point of view? Do you check whether validation is applied on the client and server as well?

    - **User input is validated both at UI and backend.**
4. Do you check whether the application is safe against the SQL Injection vulnerability by using parameterized query in the back end?
    - **IDE does not use Query languages therefore SQL Injection is not possible**

13.3. Configuration Management

1. Do you make sure all the components and packages required for the application are updated and the latest patches are applied on them?
    - **We started with the latest version but we are not keen on updating.**
2. Do you have sensitive data like database connection string, encryption key, admin credentials or any other secret should not be stored as clear text in the code?
    - **Yes. Sensitive data is stored in config. (Security > Technical Debt > Sensitive Data)**
3. The GET protocol sends the data in the query string. Sensitive information going in a GET request can be accessed from the browser history or logs. Do you check your rest APIs which use GET Http Method from this point of view?
    - **Yes. IDE does not expose any GET API with sensitive data.**

13.4. Exception Management & Auditing, Logging

1. Do you check whether centralized exception management is in place, with minimum information being displayed?
    - **Yes. (Concepts > Under-the-hood > Exception handling)**
2. Do you check if logging is enabled for different log level application and platform level as well?
    - **Yes. (Concepts > Operational concepts > logging)**
3. Do you check your application has sensitive data like user credentials, password hashes, credit card details, etc.?
    - **Yes.**

14. Architectural RoadMap

14.1. Baseline Architecture

Temel mimari şu başlıklarda açıklanmıştır:

- 03. System Scope and Context
- 04. Solution Strategy
- 05. Building Block View (Context and Scope)
- 06. Runtime View
- 08. Cross-cutting Concepts
- 09. Design Decisions

14.2. Target Architecture

14.2.1. Service designer

IDE içerisinde geliştirilen uygulamanın backend davranışını tasarlamak için kullanılan Entity Designer'ın yetersiz kaldığı durumda kullanılan Service Designer'ın IDE ile entegre edilmesi yapılacaktır. Servis designer modeli IDE içerisinde saklanacak ve kullanıcı bilgisayarında kurulu olması durumunda IDE içerisinden websocket ile haberleşilerek model editlemenin service designer üzerinde yapılması sağlanacaktır.

14.2.2. Stil sistemi

Quick Editör (Önyüz editörü) üzerindeki branding ve tema desteği CSS geliştirmesi gerektirmektedir. Bu geliştirme tasarım ekibine adreslenmiş ve geliştiricilerin Quick Editör ile önyüz tasarlarken bağımlılığa neden olmuştur. Bundan dolayı oluşan bağımlılığı Plateau-IDE üzerinde geliştirilecek olan ve branding'i CSS altyapısı gerektirmeyecek ayrı bir editör geliştirmesi yapılacaktır. Bu editörün ürettiği modeli quick runtime kendi içerisinde işleyecektir.

14.2.3. Flow

IDE ile geliştirilen uygulama içerisinde kod bilgisi gerekmeksizin view geliştirmesi yapılabilmektedir. Fakat gerek önyüz, gerek backend, gerekse midddleware (symphony) üzerindeki davranışı geliştirmek için bir şekilde kod bilgisi gereksinimi bulunmaktadır. Bu gereksinimi ortadan kaldırmak için bir akış tasarlanabilecek bir editör ve bu modeli gerek önyüz, gerek backend gerekse middleware adına işleyebilecek bir runtime geliştirmesine yapılacaktır.

14.2.4. Marketplace

IDE içerisinde geliştirlen uygulama içerisinde kullanılan gerek komponentler, gerek akış kutuları, gerekse stil sistemleri yapıldıkları application'lar içerisinde kullanılabilmektedir. Modül mantığı ise application'lar arası kullanılabilmekte fakat domain arası kullanılmamaktadır. Gerek ekiplerin bu sınırlar dışında ortak kullanımı gerekse IDE'ye gömük bazı parçaları kendileri geliştirebilmeleri

adına bir marketplace geliştirmesi yapılacaktır. Marketplace üzerinde aşağıdaki parçalar bulunabilecektir.

- modül
- model (ui page, ui component, named component, process, process design)
- flow step
- stil

14.2.5. Localization

(08.Concepts > Domain Concepts > Internationalization ve 11.Technical Risk > Technical Debts > Internationalization) kalemine istinaden geliştirme yapılacaktır.

14.2.6. App preview

Quick Editör içerisinde bir önyüz ekranının tek basına preview (mobil simulatör ile birlikte simüle) edilmesine olanak verilmektedir. Fakat bu tek ekran preview'i uygulama açısından çok yetersiz kalmaktadır. Bu yetersizliği gidermek adına, bütün bir uygulamanın, anında preview edilecebileceği bir yapı geliştirilecektir.

14.3. Transition Architectures

14.3.1. Stil sistemi

Mevcutta css kullanan uygulamar olacağı için css motoru bir süre daha devam ettirilecektir. Bu süre zarfında stil sistemine her komponent için yeni yetenekler eklenecektir. Bu yetenek miktarı yeteri kadar arttığında önce design time üzerinde css eklemesi kaldırılacaktır. Css kullanan tek bir uygulama dahi olmayana kadar runtime içerisindeki css motoru hayatına devam edecektir.

14.3.2. Flow

Önyüz açısından behaviour tarifi için event'ler içerisinde mevcutta typescript/qscript kullanılmaktadır. Bundan dolayı TS/QS kullanımı devam ettirilecektir. Bunun yanısıra istendiği takdirde flow ile behaviour tariflenebilecektir. Tariflenen bu flow UI veya backend şeklinde bir ayrım barındırmayacak, o event'te olması istenen davranış total olarak yapılabilecektir. Backend üzerinde çalışması gerekecek olan bölüm flow runtime'ı üzerinde çalıştırılacaktır.

14.3.3. Marketplace

Mevcut uygulamara kullanılmadığı takdirde bir etkisi olmayacağı için bir anda devreye alınacaktır.

14.3.4. Localization

Kullanıcılara yeni bir yetenek sunacağı ve mevcut'a etkisi olmayacağı için bir anda devreye alınacaktır.

14.3.5. App preview

Kullanıcılara yeni bir yetenek sunacağı ve mevcut'a etkisi olmayacağı için bir anda devreye alınacaktır.

15. Questionnaire

- Are you, as a programmer who will implement the system, comfortable with the architecture/applications?
  - **Yes**