LABWORKS

LABWORKS Exchange Portal

Solutions Architecture version 2.2

Last Updated:	June 01, 2022

REVISION HISTORY

DATE	REVISION TYPE	REVISION #	COMMENTS	INITIALS
11/11/2020	*Major*	*1*	*Initial version, Context view*	*DR*
11/11/2020	*Minor*	*1.1*	*Changed context view*	*DR*
12/08/2020	*Major*	*2.0*	*Container view, Component view*	*DR*
12/29/2020	*Minor*	*2.1*	*Security view, feedback*	*DR*
09/07/2020	*Minor*	*2.2*	*Small language tweaks to document*	*LS*
06/01/2022	*Minor*	*2.2*	*Formatting*	*SM*

© Copyright 1998-2022 Labworks LLC. All rights reserved

CONTENTS

Revision History 2
Contents 3
1 Solution Overview *4*
2 Context View *5*
2.1 External entities: 5
2.1.1 Laboratory 5
2.1.2 Mailing Server 6
2.1.3 Auth Provider 6
2.2 Users of the LW Exchange Portal 6
2.3 Fault Tolerance 6
3 Security View *7*
4 Container View *8*
4.1 LW Exchange Portal 8
4.1.1 SPA Frontend 9
4.1.2 Backend 9
4.1.3 DB 9
4.1.4 Reporting Software 9
4.1.5 License App 9
4.1.6 File system 9
4.2 Laboratory 10
4.2.1 Mediator Service 10
4.2.2 eLIMS Software 10
5 Component View *11*
5.1 LW Exchange Portal 11
5.1.1 Backend 11
5.1.2 Frontend 13
5.1.3 Installer 17
6 Code view *18*

© Copyright 1998-2022 Labworks LLC. All rights reserved

1 SOLUTION OVERVIEW

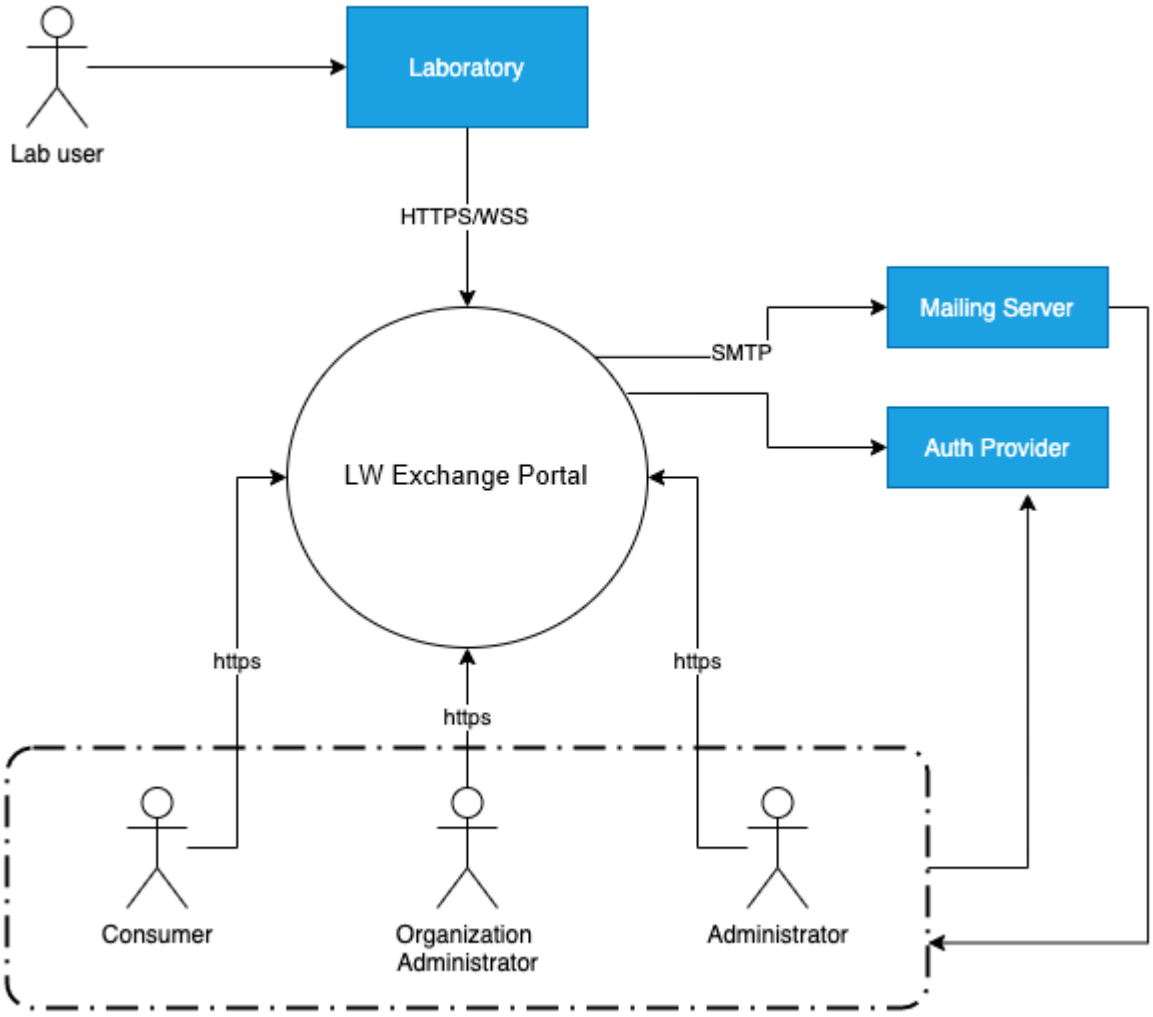
This solution introduces the new (external to eLIMS laboratory) software that will allow external users to order some services from the laboratory, receive results, check statuses, etc.

It is basically a web application that can be installed on either eLIMS server or any other server. After installation administrators will be able to configure DNS and use their own domain names.

This architecture document is based on the C4 model for software architecture (Context, Container, Component, and Code levels). Each following level uncovers more details about previous abstractions.

© Copyright 1998-2022 Labworks LLC. All rights reserved

2 CONTEXT VIEW



2.1 Context Diagram

2.1 EXTERNAL ENTITIES:

2.1.1 Laboratory

Laboratory is a laboratory with eLIMS software deployed. It sends data to the exchange portal (*LW Exchange Portal*), mainly order updates. LW Exchange Portal

Communication between *Laboratory* and *LW Exchange Portal* is done via secure HTTPS and WSS connections. *Laboratory* is always the initiator of any connection, it will not keep any port open for incoming requests for security reasons.

An administrator configures the communication port on *LW Exchange Portal*, typically 443 (https).

© Copyright 1998-2022 Labworks LLC. All rights reserved

Authentication of clients (*Laboratory*) of *LW Exchange Portal* is based on API keys with an ability to invalidate them in case they are stolen or compromised.

2.1.2 Mailing Server

Mailing Server is a mailing server on the customer side that is used for sending various notifications to users of *LW Exchange Portal*. Administrators specify configuration (host, port, username, password, etc.) in *LW Exchange Portal.*

2.1.3 Auth Provider

Auth Provider is an Authentication server (SSO system) on the customer side which is used for authentication and management of users who will be able to login to *LW Exchange Portal*. Administrators will specify configuration of the *Auth Provider* in *LW Exchange Portal*.

The list of supported protocols will be defined (currently evaluating OAuth).

2.2 USERS OF THE LW EXCHANGE PORTAL

As it is a web application, users access *LW Exchange Portal* by the website (HTTPS protocol). They authenticate either through the standard *LW Exchange Portal* mechanism based on JWT tokens or through customers' *Auth Provider*.

There are different types of users with different levels of access to the portal.

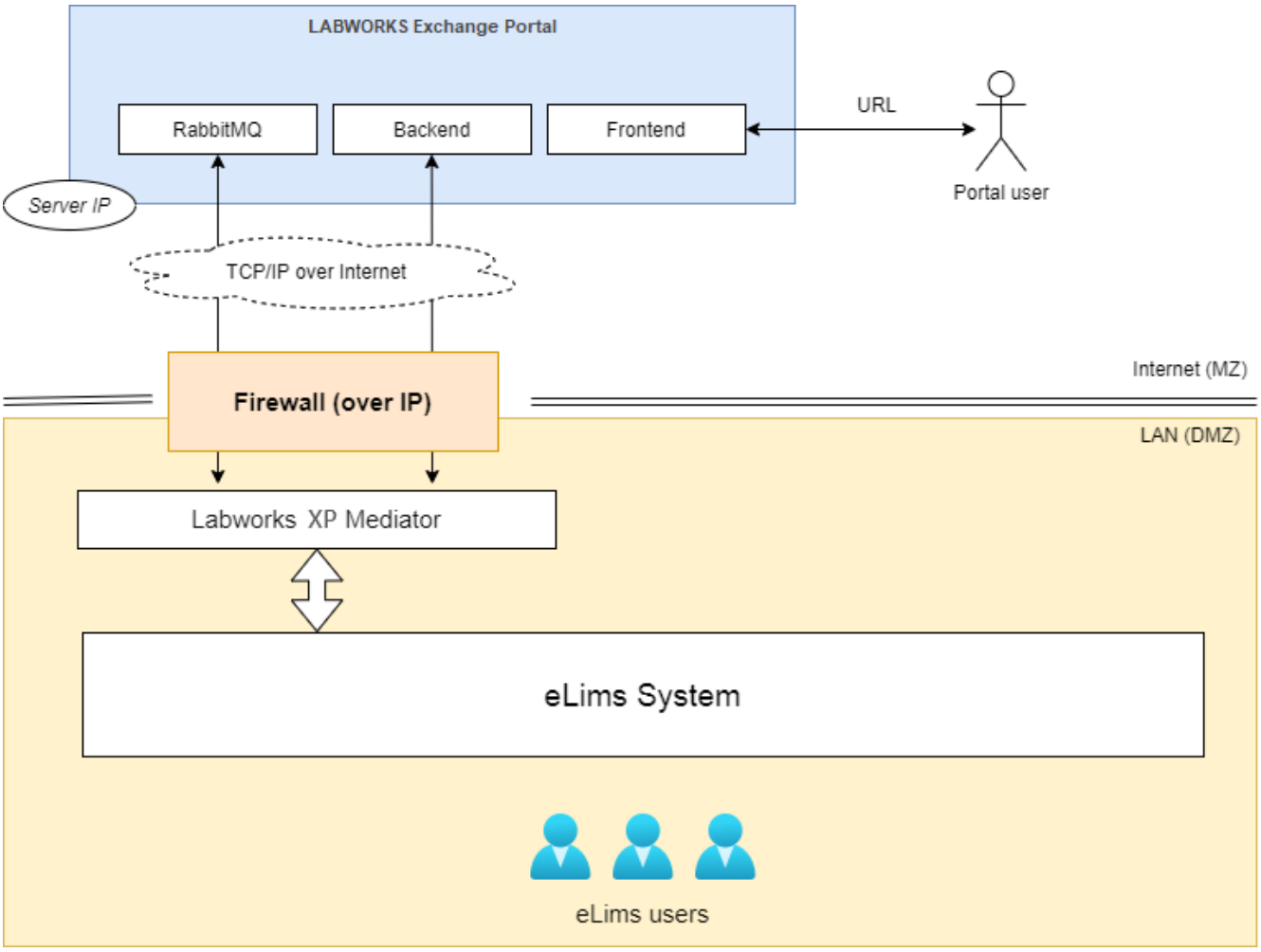
Laboratory users are special users, they can also authenticate to the *LW Exchange Portal* with a limited access level directly from the *Laboratory*. *Laboratory* issues auth tokens and sends it to *LW Exchange Portal* where they are stored as a white list, *LW Exchange Portal* creates sessions for these users, although it also allows them to generate a password via the *LW Exchange Portal* so they can authenticate to it directly.

2.3 FAULT TOLERANCE

LW Exchange Portal is responsible for storing consumer data, eLIMS is responsible for managing all laboratory data. *Laboratory* continues to work even if a connection between it and *LW Exchange Portal* cannot be established. *Laboratory* waits for reconnection and sends data once communication is stabilized (the circuit breaker pattern). In addition, data is synchronized periodically (job in *Laboratory*).

© Copyright 1998-2022 Labworks LLC. All rights reserved

3 SECURITY VIEW



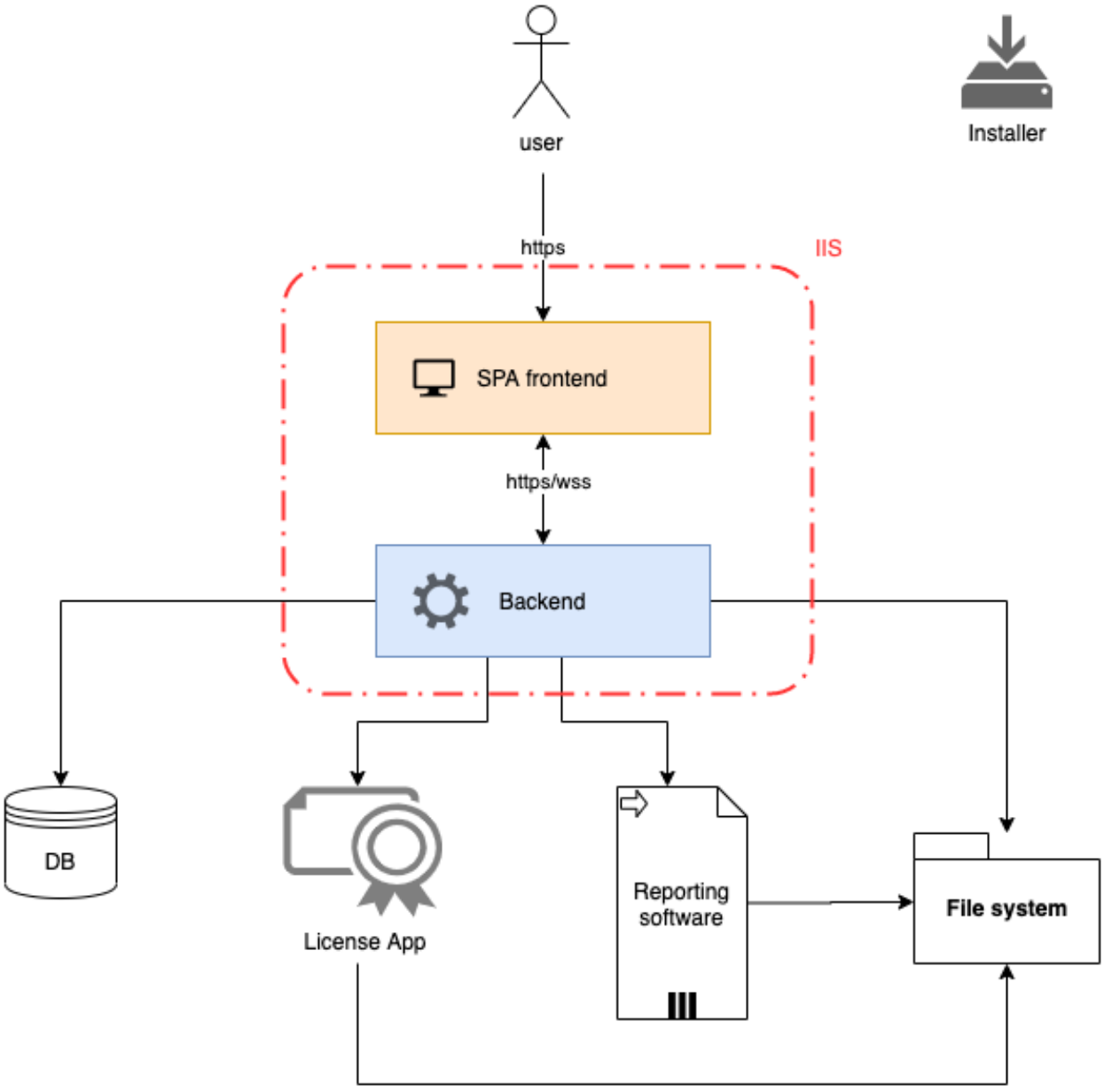
3.1 Security View

LABWORKS Exchange Portal is isolated from the laboratory server (though may be installed on the same machine). *Laboratory* users work in their LAN(DMZ) shielded from MZ with Firewall (Over IP). *LABWORKS Exchange Portal users* work in MZ using the auth mechanisms described above.

© Copyright 1998-2022 Labworks LLC. All rights reserved

4 CONTAINER VIEW

4.1 LW EXCHANGE PORTAL



© Copyright 1998-2022 Labworks LLC. All rights reserved

4.1 LW XP Container View

4.1.1 SPA Frontend

SPA Frontend is a SPA client application (website) hosted on IIS server. Users reach it using a browser via HTTPS protocol. The main framework used for building this application is angular 11.

4.1.2 Backend

Backend is a .Net Web API application hosted on IIS server which provides REST API for *SPA Frontend* and *Laboratory*. Communication with *SPA Frontend* is done using HTTPS and WSS (secure WebSocket) protocol.

SPA Frontend initiates a WebSocket connection, it is used for receiving live updates from *Backend* (status changes, notifications, etc.).

4.1.3 DB

DB is a database management system used for storing data. All the consumer-wise data stored in this DB. Only Microsoft SQL Server is supported at this time. It is installed and configured by an administrator of *LW Exchange portal*. Product also ships with a job that backups the *DB* each day (or any other custom period configured by the administrator).

Backend oversees storing and retrieving data from the *DB*.

4.1.4 Reporting Software

Reporting software is used for generating consumer reports. *LW Exchange Portal* will use Crystal Reports, it is installed on the same server as the Exchange Portal. Reports are saved to *File System* and may be accessed by the *Laboratory*.

4.1.5 License App

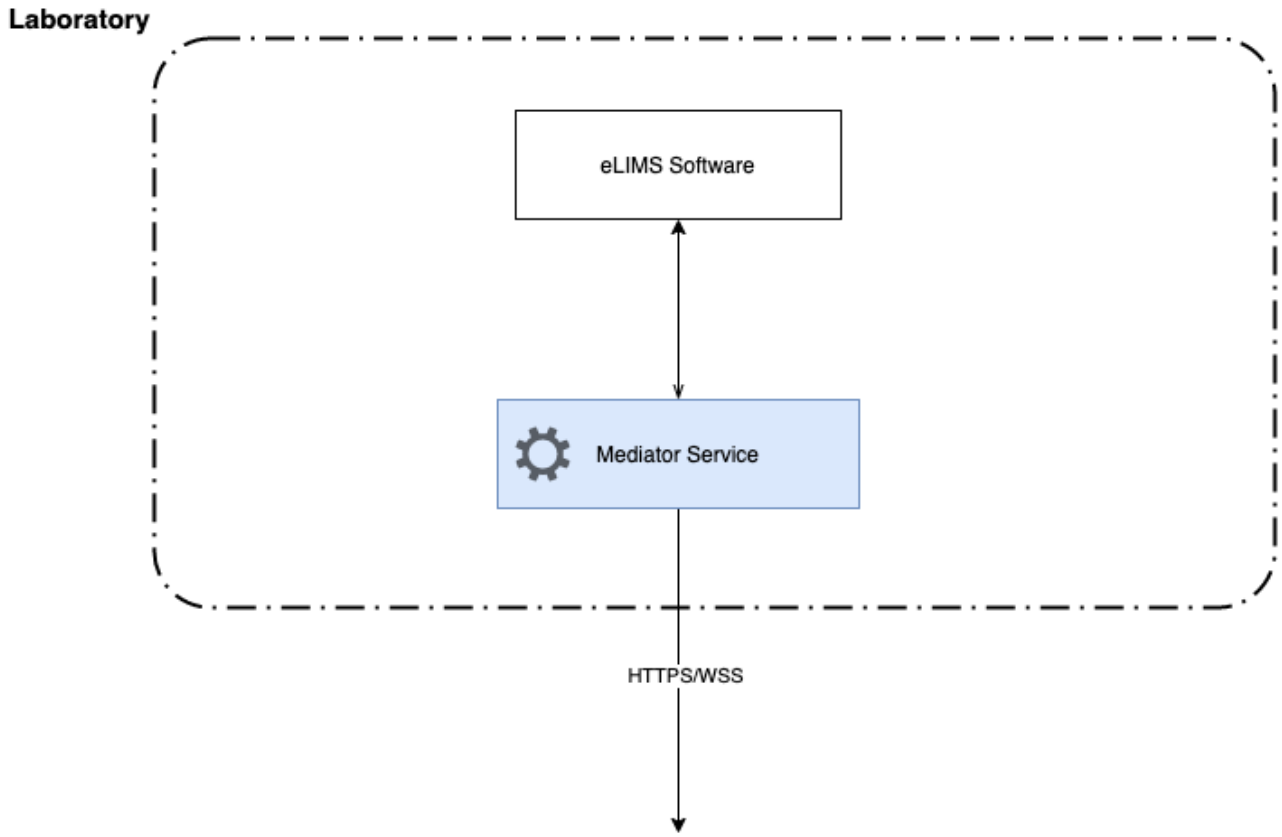
License App is a service used for generating the *LW Exchange portal*`s end-user licenses.
It can generate a file with system information provided from the *LW Exchange portal* server and be used by the LABWORKS Support Team to verify the installation is licensed.

4.1.6 File system

File System is the OS file system level storage which is used for storing reports, the license file, etc. Attachments to orders (including reports) are downloaded from the file system as well. However, the data management layer is being abstracted so it is possible to move data into a CDN in a future version of the product.

© Copyright 1998-2022 Labworks LLC. All rights reserved

4.2 LABORATORY



4.2 Laboratory Container View

4.2.1 Mediator Service

Mediator Service is a .Net application that requests data and updates from the *LW Exchange Portal*, pushes data to *LW Exchange Portal*, communicates with *eLIMS Software*. It buffers updates to *LW Exchange Portal* and ensures fault tolerance.

4.2.2 eLIMS Software

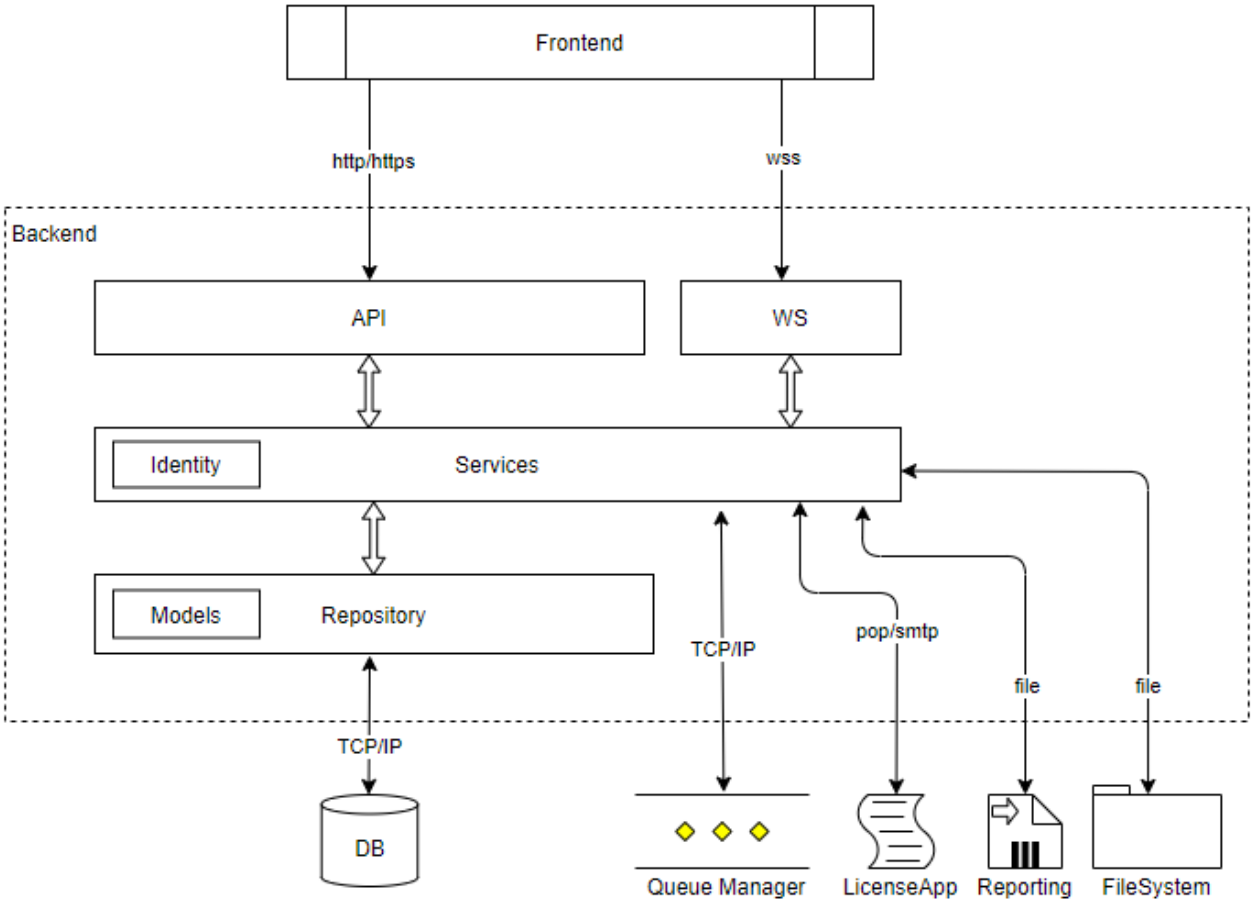
eLIMS Software is software installed in laboratories that requests and consumes updates from/pushes updates to *LW Exchange Portal* through the *Mediator Service.*

© Copyright 1998-2022 Labworks LLC. All rights reserved

5 Component View

5.1 LW Exchange Portal

5.1.1 Backend



5.1.1.1 API

API is *Backend*'s service for *Frontend*. *Frontend* uses API to access and perform actions on *LW Exchange Portal* data. Interactions with the API are performed via a secure HTTPS connection.

5.1.1.2 WS

WS is a special module that provides a direct link between *Backend* and *Frontend*. Communication between *Frontend* and WS is performed over a secure WebSocket connection.

5.1.1.3 Services

© Copyright 1998-2022 Labworks LLC. All rights reserved

Services are the main executable layer of *Backend*. This layer unites all the functions which define the business logic of *LW Exchange Portal*. All communications with external systems or resources are performed exclusively through the *Services* layer.

5.1.1.4 Repository

Repository is a special layer, designed for integration with external DBMS. *Repository* layer allows us to easily replace DBMS in the future.

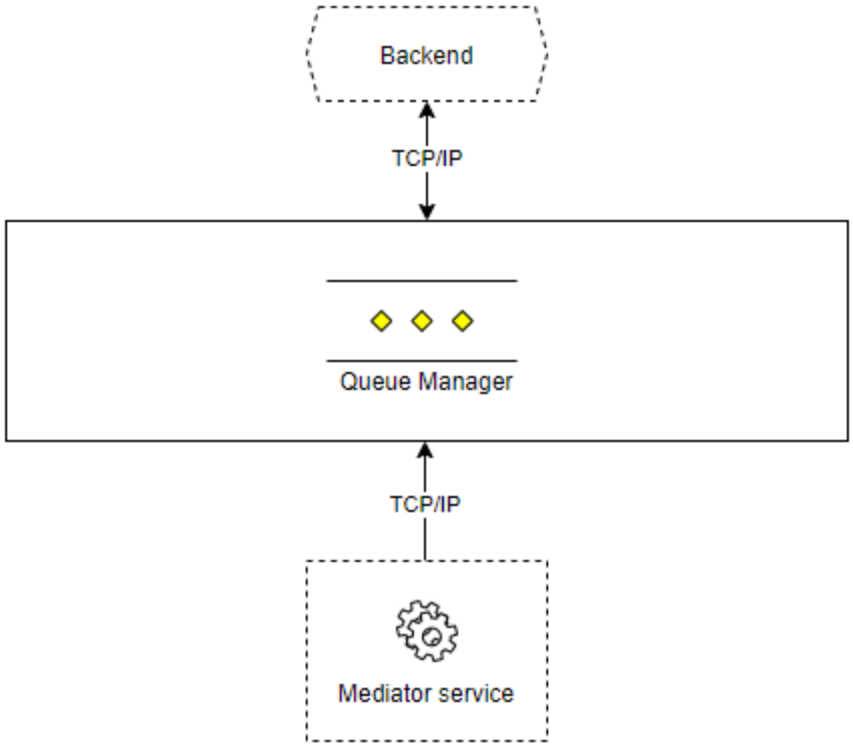
5.1.1.5 Identity

Identity is a special module of *Service* layer, responsible for user management, permissions control, etc. All the data is stored in the *DB*.

5.1.1.6 Models

Models are a collection of data descriptions/interfaces that the *LW Exchange Portal* works with. *Models* are used to define the DB schema (*codefirst* approach) and for internal exchange between backend components.

5.1.1.7 Queue Manager



Queue Manager is used for providing direct communication between *LW Exchange Portal* and the *Laboratory*. Communication between *Queue Manager* and *Backend* or *Mediator service* of the *Laboratory* is done via a TCP/IP-socket. *RabbitMQ* is used as a queue manager. It resides on *LW Exchange Portal's* side.

© Copyright 1998-2022 Labworks LLC. All rights reserved

5.1.1.8 QA

5.1.1.8.1 Unit testing

Unit testing is used to ensure the quality of Backend. Microsoft MSTest framework is used. Integration testing for *Backend* is assumed only for critical parts.

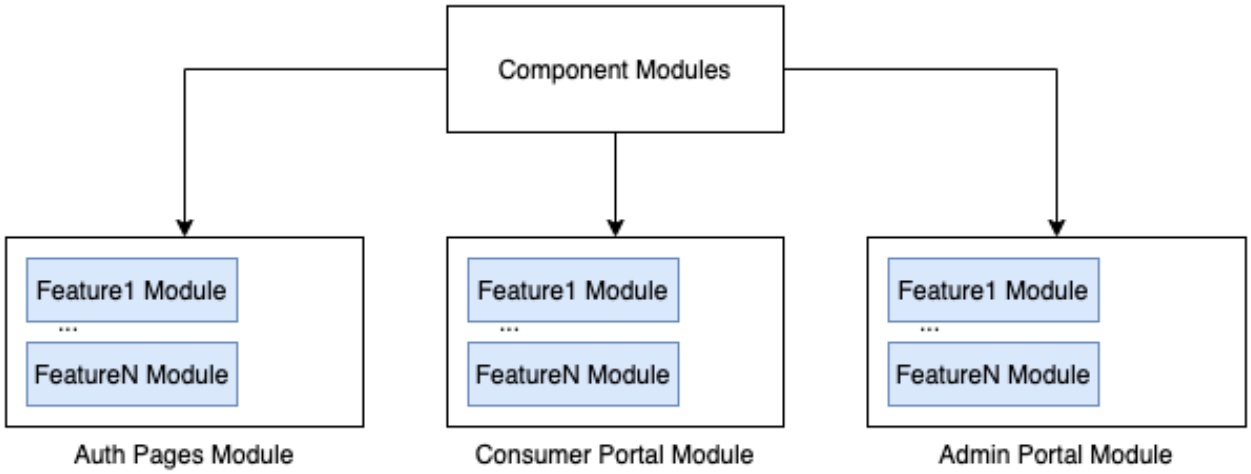
5.1.1.8.2 Load testing

Load testing is not assumed as number of consumer users is expected to be low.

5.1.2 Frontend

Frontend is a single page application web application that consumes data from *Backend* and provides user interfaces. It is implemented using the angular 11 framework.

5.1.2.1 Modules



5.1 Modules

The application is divided into modules to organize related things together. These modules are lazy-loaded which allows us to decrease the bundle size and significantly improve user experience.

5.1.2.1.1 Component Modules

Component Modules is a component library, a bunch of modules that provides reusable components like buttons, inputs, etc. They are loaded by feature modules once they are needed.

© Copyright 1998-2022 Labworks LLC. All rights reserved

5.1.2.1.2 Auth Pages Module

Auth Pages Module is a module that provides user/permission management functionality and feature modules. Feature modules are basically pages like Sign In, Restore Password, etc.

5.1.2.1.3 Admin Portal Module

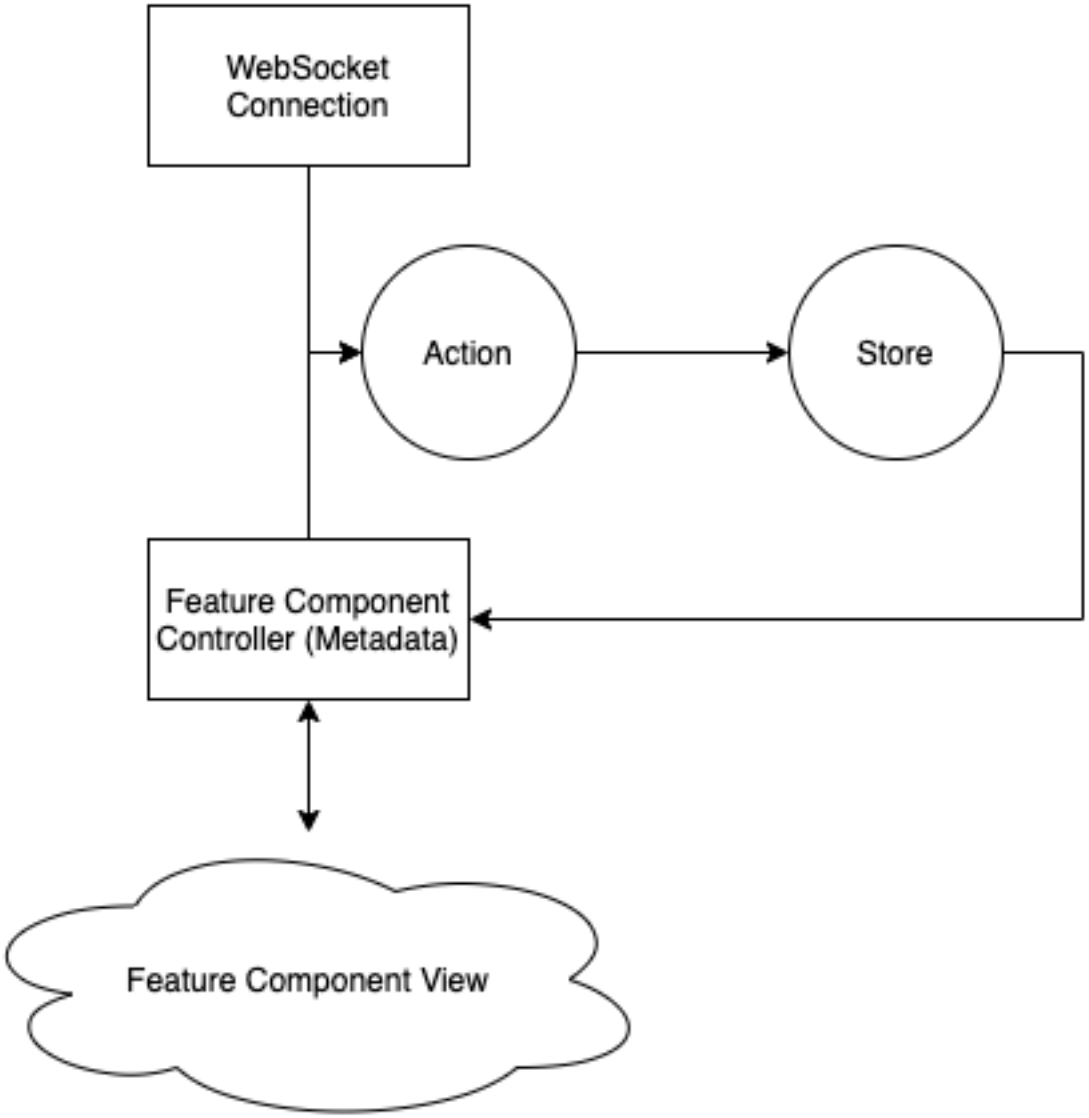
Admin Portal Module is a module that provides the admin part of the portal functionality. It is loaded once user navigates to admin pages.

5.1.2.1.4 Consumer Portal Module

Consumer Portal Module is a module that provides the consumer part of the portal functionality. It is loaded once the user navigates to any of the consumer pages.

© Copyright 1998-2022 Labworks LLC. All rights reserved

5.1.2.2 State Management



5.2 State Management

Frontend uses a centralized store, state management is performed using NGXS library. It is divided into feature modules.

5.1.2.2.1 Actions

Actions are commands that trigger something to happen. It mutates the *Store* and takes care of side effects. Main application business logic is implemented through actions.

© Copyright 1998-2022 Labworks LLC. All rights reserved

5.1.2.2.2 Store

Store is a global storage that provides data (reactive) to feature components.

5.1.2.2.3 WebSocket connections

WebSocket connection is a WSS connection to *Backend* which notifies about updates and dispatches actions. So we can keep the UI up-to-date as data changes occur. It is implemented using @ngxs/websocket-plugin.

5.1.2.2.4 Feature Components

Feature Components are angular components that provide a specific feature (basically pages). They dispatch actions. Views are displayed based on the *Store* state.

5.1.2.3 Design System

Design System defines rules on how the design is organized. It is a single source of truth, it guarantees the design does not unpredictably change and helps keep the design consistent and makes it easier to apply design changes to the software. *Design System* is planned and maintained using FIGMA. Design tokens (colours, fonts, grid, etc.) are defined in a global *.less* constants file, the implementation is based on these tokens. Typography is defined in a separate *.less* file, components are implemented in Component Modules.

5.1.2.4 CSS Methodology

BEM methodology is used to organize CSS in the project. It is a set of guidelines for writing modular, reusable, and scalable CSS. In addition to that, we use CSS preprocessor *less* that provides extensions to basic CSS.

5.1.2.5 L10n and I18n

The application is prepared to be usable in different locales around the world. @angular/localize app is used for that, language tokens are defined in xlf files and may be translated to other languages by translators. Locale-based formats like dates are supported by built-in angular pipes. Custom pipes may be created but should be avoided.

5.1.2.6 Mobile Support

The responsive design approach is used to support mobile applications and improve the overall UX. This is achieved by handling mobile events and CSS3 media queries. CSS grids are used to make handling these events easier.

5.1.2.7 QA

Unit, Integration, and e2e testing are used to ensure the quality of *Frontend*. Unit and Integration tests are implemented when it is required, e2e tests are implemented for the most critical parts of the application. Cypress framework is used for e2e testing.

© Copyright 1998-2022 Labworks LLC. All rights reserved

5.1.3 Installer

WiX is used to implement an installer for windows OS. It installs and configures an IIS site.

© Copyright 1998-2022 Labworks LLC. All rights reserved

6 CODE VIEW

TBD

© Copyright 1998-2022 Labworks LLC. All rights reserved