**Часть 1: Проектирование архитектуры**

1. Схема системыВысокоуровневая диаграмма: lims.drawio  
  
2. Выбор технологийОбоснование выбора фреймворков и инструментов

ELK   
- отказ от лишнего микросервиса (БД, бизнес-логика, UI)  
- подходит под задачу аудита действия пользователей  
  
RabbitMQ  
- гибкая маршрутизация сообщений  
- надежное хранение/доставка сообщений  
- простота  
  
API Gateway/Load balancer: HAProxy  
- сочетает Load balancer и API Gateway, минус одна точка  
- популярный

.NET  
- язык C# - удачный для построения сложных архитектур  
- nuget  
- кроссплатформенность  
- популярность  
  
Dapper  
- высокопроизводительный  
- легковесный, простой

PostgreSQL  
- open source  
- адаптирован под highload  
- может хранить большие объёмы данных  
- наличие типов данных, кроме стандартных

Redis  
- наличие типов данных, кроме стандартных

Решение для контейнеризации  
Docker  
Обоснование  
- обширная экосистема  
- контейнеры потребляют относительно мало ресурсов  
- относительно высокая скорость запуска контейнеров  
- популярный  
  
Образ: alpine   
Обоснование:   
- малый размер  
- минимальное потребление ресурсов

3. Безопасность  
Аутентификация/Авторизация  
Происходит в микросервисе Users: с помощью middleware ASP.NET  
Аутентификационные куки доступны во всех микросервисах через RabbitMQ  
Использовать двухфакторную аутентификацию

Защита данных  
Чувствительные поля: логины, пароли пользователей, хранятся в БД в зашифрованном виде  
Все узлы системы взаимодействуют по HTTPS

**Часть 3: Документация**1. Описание архитектуры  
  
Балансировщик нагрузки/API Gateway: HAProxy

Микросервисы:  
SampleTracking (образцы) БД: PostgreSQL  
Reports (отчёты) БД: PostgreSQL  
Users (пользователи) БД: PostgreSQL  
  
Для микросервисов SampleTracking, Reports данные из БД кэшируются (Redis)  
  
Каждый микросервис, ElasticSearch, Logstash, Kibana оборачиваются в Docker контейнеры  
Микросервисы, RabbitMQ недоступны извне. Публичный доступ – для клиентов - только у HAProxy

Масштабируемость для работы с 10 000+ операций в день  
Максимальный RPS у HAProxy - 2 000 000. Требование выполняется

UML диаграмма доменной модели: domain.drawio

Роли системы: Администратор (admin), Руководители (head), Лаборанты (assistant)  
  
REST API  
Микросервис SampleTrackingGET /samples – получить все образцы. Роли: admin, head  
GET /samples/{id} – получить образец по Id. Роли: admin, head  
POST /samples – добавить образец. Роли: admin  
PUT /samples – изменить данные образца. Роли: admin  
DELETE /samples/{id} – удалить образец. Роли: admin

Микросервис ReportsGET /reports – получить все отчёты. Роли: admin, head  
GET /reports/{id} – получить отчёт по Id. Роли: admin, head  
POST /reports – добавить отчёт. Роли: admin, head  
PUT /reports – изменить данные отчёта. Роли: admin, head  
DELETE /reports/{id} – удалить отчёт. Роли: admin, head

Микросервис Users GET /users – получить всех пользователей. Роли: admin  
GET /users/{id} – получить пользователя по Id. Роли: admin  
POST /users – добавить пользователя. Роли: admin  
PUT /users – изменить данные пользователя. Роли: admin  
DELETE /users/{id} – удалить пользователя. Роли: admin

Аудит действий пользователей  
Во всех микросервисах при любых действиях пользователя формируются логи.   
Информация в логе: пользователь, дата и время, действие. Лог фиксируется Filebeat-м и отправляется в Elasticsearch через Logstash  
  
Масштабирование системы при росте нагрузки  
Куб масштабирования: вертикальное, горизонтальное. Поиск компромиссов в зависимости от обстоятельств: характер нагрузки, финансовая ситуация и др.  
Горизонтальное: запускаются копии микросервиса  
Вертикальное: замена процессора, планок ОЗУ на сервере

Возможные узкие места и способы их устранения  
узкое место: сетевое подключение: может быть медленным/ненадёжным. Сетевых подключений может быть много  
способ устранения: пересмотреть аппаратную конфигурацию сети, оптимизировать сетевое взаимодействие

узкое место: Аппаратная конфигурация микросервиса. Может быть «впритык» – например, из-за ограниченного финансирования на старте проекта  
способ устранения: выбор более дорогой материнской платы сервера «на перспективу», вертикальное масштабирование. На начальном этапе можно установить скромные процессор, память и др.

узкое место: громоздкие протоколы / форматы данных.   
способ устранения: использовать компактные протоколы/ форматы данных. Добиться, чтобы объём сообщения был минимальным  
  
При признаках отказа микросервиса поднимается его копия – балансировщик распределяет нагрузку