Лабораторная работа №7. Диаграммы развёртывания

Тема: Система управления работой станции метрополитена

Выполнила студентка группы М3311

Авсюкевич Анастасия

Для эффективной работы системы централизованного управления станцией метро, которая включает различные подсистемы (диспетчеризация, контроль станции, продажа билетов, работа турникетов, эскалаторов, безопасность), нужно учитывать не только функциональные аспекты, но и требования к технологиям, архитектуре и внешним устройствам. Эти требования включают в себя выбор языков программирования, СУБД, операционных систем и прочего программного обеспечения.

Будем придерживаться трехзвенной архитектуры, которая состоит из следующих уровней:

1. Клиентский уровень (Presentation Layer): Это фронт-энд приложения, которое взаимодействует с пользователями, такими как операторы и сотрудники станции.
2. Бизнес-логика (Business Logic Layer): Серверный уровень, который обрабатывает логику приложения, взаимодействует с данными и обеспечивает их обработку.
3. Уровень данных (Data Layer): Уровень для хранения и управления данными, включает базы данных и другие хранилища данных.

Нефункциональные требования:

1. Язык программирования серверной и клиентской частей с соответствующими средами выполнения

* Серверная часть (BL):
  + Язык программирования:
    - Java — хорош для масштабируемых, высоконагруженных систем, таких как система управления поездами и турникетами.
  + Среда выполнения:
    - Для Java — JDK 11+.
* Клиентская часть (PL):
  + Язык программирования: JavaScript (React)
  + Среда выполнения: Кроссплатформенные фреймворки (React native) для веб-браузеров с поддержкой современных стандартов HTML5 и CSS3.

2. СУБД с указанием схем хранимых данных (DL)

Для разных подсистем могут использоваться различные СУБД:

* Основная СУБД для всех подсистем: PostgreSQL
  + Данные о поездах и расписаниях: таблицы для управления поездами (поезд ID, маршрут, время отправления, текущее местоположение).
  + Данные о продажах билетов: таблицы для хранения транзакций, данных о пассажирах, типах билетов, платежных данных.
  + Данные о турникетах: таблицы для учёта входных и выходных данных по турникетам, пользователей и устройств.
  + Данные о системе безопасности: таблицы для мониторинга событий тревоги, записи данных с видеокамер.
  + Данные о работе эскалаторов: фиксация состояний, работоспособности машин.
  + Обработка и анализ событий безопасности: данные с видеонаблюдения и события тревоги от системы безопасности.
* СУБД для системы работы с реальным временем: Apache Cassandra.
  + Данные о текущем местоположении поездов и эскалаторов: для быстрой обработки и хранения данных в реальном времени.
* СУБД для хранения логов и мониторинга: Kibana (ELK stack).
  + Данные о событиях системы: таблицы с логами ошибок, состояния оборудования, тревог и т.д.
  + Хранилище логов может передавать информацию в PostgreSQL через стандартное соединение PostgreSQL по протоколу TCP/IP (с использованием SQL-запросов) или через API с использованием HTTP/HTTPS.

Языки программирования для СУБД:

* SQL - для PostgreSql
* NoSQL – для Cassandra

3. Операционные системы

* Для серверов:
  + Linux (Ubuntu) — для обработки критически важных задач с высокой производительностью и надёжностью.
* Для рабочих станций и клиентских устройств:
  + Windows 10/11 — для сотрудников, работающих с клиентским интерфейсом (например, операторы).
  + Linux (Ubuntu) — для работы с терминалами и встроенными устройствами, такими как датчики и камеры.
* Для виртуализации:
  + Docker — для создания изолированных контейнеров для различных микросервисов.
  + Kubernetes — для оркестрации контейнеров и управления развертыванием сервисов.

4. Внешние устройства

Для работы системы потребуются различные внешние устройства:

* Датчики и камеры для системы безопасности:
  + Камеры видеонаблюдения с возможностью видеозаписи и передачи данных на сервер.
  + Детекторы движения и устройства для мониторинга действий в зоне безопасности.
* Турникеты:
  + Устройства для считывания билетов (RFID, магнитные карты).
  + Считыватели QR-кодов и штрихкодов (MTX200)
  + Турникеты с электромеханическими приводами.
* Эскалаторы:
  + Датчики для контроля движения эскалаторов.
  + Системы мониторинга и автоматического отключения эскалаторов при неисправностях.
* Системы связи и взаимодействия:
  + Устройства для передачи данных (Ethernet).
  + Средства передачи данных между подсистемами (MQTT, WebSocket, RESTful API).

5. Прочее ПО

* Очередь сообщений:
  + RabbitMQ для асинхронной передачи данных между микросервисами и обеспечения взаимодействия различных подсистем (например, для передачи событий тревоги или обработки информации о поездах).
  + RabbitMQ использует протокол AMQP для обмена сообщениями, но для передачи данных в PostgreSQL используется соединение через TCP/IP или HTTP/HTTPS через промежуточные слои, такие как микросервисы или API.
  + **Протокол MQTT** будет оптимален для датчиков, работающих в реальном времени (например, датчики движения, камеры, датчики температуры и т.д.), поскольку этот протокол поддерживает эффективную передачу данных через ограниченные каналы и используется в IoT-системах.
* Система логирования и аналитики:
  + ELK stack (Kibana) для сбора, обработки и визуализации логов.

Структура взаимодействия:

1. Аппаратные узлы:
   * Станции метро: ПК, турникеты, камеры и датчики.
   * Центральный сервер: сервер, на котором развернуты основные компоненты системы — базы данных, API-сервисы, приложения для обработки данных.
   * Хранилища данных (СУБД): базы данных для хранения информации о поездах, пассажирах, билетах, событиях безопасности.
2. Среды выполнения:
   * Контейнеры Docker для изоляции микросервисов и обеспечения масштабируемости.
   * Kubernetes для управления развертыванием и масштабированием.
3. Компоненты/пакеты:
   * Серверы приложений (API, веб-сервера).
   * Базы данных для хранения информации о поездах, билетов и событий системы безопасности.
4. Подключения и протоколы:
   * HTTP/HTTPS для взаимодействия с клиентскими устройствами.
   * MQTT/WebSocket для асинхронной передачи данных между компонентами.
   * Ethernet для передачи данных с удалённых устройств, таких как камеры и датчики.
5. Файлы конфигурации:
   * Конфигурации для подключения баз данных, серверов API и настройки виртуальных машин.

Диаграмма развертывания:

