Итоговое задание

В задании требуется написать приложение, моделирующее работу информационной системы некоторой компании, осуществляющей перевозку грузов. Ниже более подробно описаны предметная область и технические требования.

# Предметная область

Существуют следующие виды сущностей:

* Фура
  + Рег. номер (2 лат.буквы + 5 цифр)
  + Размер смены водителей
  + Вместимость (тонн)
  + Состояние (исправен, неисправен)
  + Текущий город
* Водитель
  + Имя
  + Фамилия
  + Личный номер
  + Отработано часов в этом месяце
  + Статус (отдых, в смене, за рулем)
  + Текущий город
  + Текущая фура
* Заказ
  + Уникальный номер
  + Выполнен (да/нет)
  + Список маршрутных точек
    - Город
    - Груз
    - Тип (погрузка/выгрузка)
  + Фура назначенная выполнять заказ
  + Список водителей, которые выполняют заказ
* Груз
  + Номер груза
  + Наименование
  + Масса (в кг)
  + Статус (подготовлен, отгружен, доставлен)
* Карта страны
  + Города
  + Расстояния

Приложение должно обеспечивать следующую функциональность:

* Для сотрудников компании (через UI-интерфейс):
  + просмотр списка, добавление, редактирование и удаление фур, водителей;
  + просмотр списка и добавление новых заказов с проверкой, что:
    - все загружаемые грузы должны быть где-то выгружены;
    - все выгружаемые грузы должны быть где-то загружены;
  + просмотр состояния заказов и грузов;
  + вывод списка фур, которые подходят для доставки заказа, если:
    - фура находится в исправном состоянии;
    - фура подходит по вместимости (с учетом погрузки/выгрузки грузов в городах по маршруту следования);
    - фура не выполняет в данный момент никаких заказов;
  + подбор и назначение водителей на основании размера смены используемой фуры и примерного времени в пути (рассчитывается по карте городов и путевым точкам):
    - лимит времени за месяц (176 часов) для каждого из водителей в смене не будет превышен в ходе выполнения этого заказа (учесть также смену месяцев в ходе заказа);
    - водитель не выполняет сейчас другие заказы;
    - при назначении водитель находится в том же городе, что и фура.
* Для водителей (через UI-интерфейс):
  + отобразить следующую информацию:
    - личный номер водителя
    - личные номера со-водителя (-ей)
    - рег. номер фуры
    - номер заказа
    - список маршрутных точек
  + изменять фактическое время работы и статус заказа:
    - водитель заступил/окончил смену
    - водитель изменил статус:
      * За рулём
      * Второй водитель
      * Погрузочно-разгрузочные работы
      * Отдых
    - водитель получил/выгрузил груз (изменить статус заказа)
      * Загрузил
      * Выгрузил

# Технические требования

В итоге требуется получить многопользовательское приложение типа клиент-сервер с соединением по сети.

Все данные хранятся на стороне сервера. Каждый клиент может загружать некоторые данные, после каждой операции изменения данные должны быть синхронизованы с сервером.

Клиент должен иметь графический интерфейс.

Приложение должно обрабатывать аппаратные и программные ошибки.

# II часть

Реализовать отдельное клиент-приложение типа электронное табло, которое будет показывать полную информацию о последних заказах (минимум 10), их количество будет зависеть от вашего UI.

На этом же экране должна отображаться сводная информация по водителям и фурам за текущий месяц. Сколько всего водителей, сколько доступных/недоступных. Сколько всего фур, сколько доступных/занятых на заказе/неисправных.

Данные должны подгружаться при старте и храниться на стороне клиента. Перезагрузка данных осуществляется в случае получения уведомления от сервера.

# Критерии успешного выполнения

1. Функциональность работает (обязательно наличие UI)

2. Maven-based проект, разбитый на модули (билд одной командой, деплой одной командой)

3. Описаны интерфейсы предметной области

4. Подключена БД MySQL

5. Созданы сущности предметной области; маппинг на таблицы в БД

6. Работа с сущностями через DAO

7. Приложение развернуто на AS

8. Реализована обработка исключений

9. Подключено логгирование

10. Наличие technical solution description

11. Наличие unit-тестов на бизнес логику

# Используемые технологии

В рамках школы студенты не ограничены в выборе технологий для реализации проекта в рамках стека технологий, используемых в компании в различных проектах (лекции покрывают небольшую часть из них). Список технологий для выбора приведен ниже. Однако проект должен быть написан преимущественно на java (допускается написание незначительных частей, отдельных модулей или дополнительной функциональности на других языках, например, Kotlin или Python). После обсуждения с куратором и по согласованию с ним студент может выбрать любые комбинации технологий и фреймворков, но стоит учитывать, что куратор может быть незнаком с какими-то из них.

Список технологий, используемых в некоторых проектах компании:

Spring Core,Spring Data,AOP,Spring Boot,Spring Security,MVC,Spring Cloud, Microservices, Data processing(Spark, Apache Flume), CDI/EJB, Apache CXF, RedHat Fuse, Oracle eCommerce (ATG, Endeca...), JBoss, Apache Tomcat, Embedded Tomcat H2 (in-memory), Websphere Application Server 9.0, React, Angular/NPM/Webpack, typescript, JSF/ExtJs, JSP/Servlets, JavaScript / TypeScript / HTML / Sass / CSS/LESS/jQuery/GraphQL, SOAP/REST, JDBC, XML/XSLT/XSD,Bash scripting, Oracle Database, Gradle, Git, Maven, Subversion, Jenkins, Grafana, GitlabCI, Sonar, UNIX shell, Bootstrap, Formbased authorisation/JWT, Drools, ARS, DOM, PL SQL, Oracle AQ, IBM MQ, JMS, OracleDB ,Apache Camel, Docker, Microsoft T-Sql, WebDriver (Java), Oracle ATG, python, mapstruct, Lombok, Groovy, openshift, EC2, S3, Mongo, Casandra, hazelcast ,Junit, testing, Selenium, Mockito, Selenid, Prometheus, Kubernetes , Helm, Kibana, AWS, Apache Camel, Go, K8s, ETCD, PostgreSql, Wicket, blockchain, Web Flux, JPA/Hibernate

Задание состоит из 2 частей, каждая часть должна быть реализована с использованием разных неповторяющихся технологий.

Предлагаемый вариант стека технологий для 1 реализации части на основании лекций:

* IDE - Any (Eclipse, IDEA)
* Tomcat
* DB – MySQL
* Maven
* JPA
* Spring Framework (кроме Boot, Data)
* JSP

Для 2 части:

* Maven
* AS – WildFly
* EJB
* JSF
* MQ (для уведомлений от сервера)
* WebServices (для обмена данных между клиентом и сервером)

Подключение любого фреймворка/библиотеки принимаются в зачет только при условии выполнения пунктов, описанных выше.

Рекомендуем использовать последнюю стабильную версию технологий, обязательных к использованию.

Плюсом будет использование следующих технологий: Selenium, Sonar, Angular/React (например, админка для табло), Docker, Microservices, использования доступного cloud для развертывания приложения (например, <https://cloud.yandex.ru/> - 1 месяц бесплатного использования) и/или наличие «killer features».