Варианты тем:

«Разработка АРМ бухгалтера по учету аренды помещение в бюджетном/государственном учреждении»

«Система управления бух учетом арендной платы помещений»

Функционал:

1. Прием данных с помощью предоставленного удобного интерфейса, абстрагированного от таблиц, колонок и столбцов, как в Excel. Вместо этого – работа с сущностями (арендатор, договор, компания).
2. Автоматизация расчётов, связанных с арендой помещений (арендной платой).
3. Система уведомлений для контроля за сроками оплаты (просроченные, не просроченные)
4. Наглядное представление информации по арендаторам, договорам, счетам.
5. Гибкая настройка шаблонов, позволяющая использовать приложение без его привязки к конкретному учреждению.
6. Быстрое предоставление необходимой информации с помощью поиска
7. Архивация данных. Экспорт для переноса. Автоматический бэкап для защиты от потери.
8. Объединение счетов за оплату в единые ведомости и ордер. (без необходимости copy-paste из файла в файл).
9. Вывод данных в Excel: от простого счета для оплаты по договору до объединенных в единый мемориальный ордер документов.
10. Поддержка различных форматов для вывода данных. (.xsl – Excel 2003, xslx).
11. Быстрый доступ к истории оплаты каждого арендаторы

Подробнее о том, как приложение решает поставленные задачи и реализует функционал.

Рассмотреть процесс ручной работы и описать преимущества, которые даст автоматизация.

Целесообразность написания десктоп приложения.

Нецелесообразность использования Excel и макросов VBA.

Потоки данных

Управляющее воздействие: стандарт «Аренда», налоговый кодекс, земельный кодекс, постановление, инструкции и законы.

Механизмы: экономист

Выходные данные: готовые Excel-файлы, сформированные по определенному шаблону

Входные данные: договор с арендатором

Рабочий процесс пользователя и программы

Алгоритм учета аренды помещений:

1. Экономист получает подписанный договор с арендатором.
2. Создание нового арендатора в разделе «Список арендаторов»
3. Создание сущности корпуса в разделе «Список корпусов»
4. Внесение информации о договоре в программу
   1. Нажать кнопку «Новый договор» в разделе «Договоры»
      1. Выбрать арендатора
      2. Выбрать корпус
      3. Заполнить информацию об арендованном помещении

Создание нового договора осуществляется в отдельной вкладке/окне.  
На этом этапе заполняются все поля на основе сведений о договоре. В выпадающем списке, при помощи поиска, можно выбрать предварительно созданные для договора сущности **(\*)**.  
После создания договора, он появляется в разделе «Договоры»

В списке договоров можно перейти к подробной информации о конкретном договоре, а также произвести расчет арендной платы, коммунальных услуг, земельного налога и аренды оборудования.  
Расчет производится помесячно, для каждого месяца формируется свой Excel-документ для распечатки счёта.  
  
При оплате счёта арендатором, экономист помечает соответствующий месяц погашенным.  
В случае неуплаты, месяц помечается как просроченный, и идет автоматический расчет пени.  
За некоторое количество дней до окончания срока аренды срабатывает система уведомлений.

Для каждого месяца есть возможность сформировать обобщающие документы – накопительную ведомость, мемориальный ордер и т.д.   
Они охватывают различные данные по всем договорам за весь месяц.

В профиле для каждого арендатора есть соответствующая для него история договоров, его счетов и оплат. Таким образом, при необходимости, можно быстро проверить арендатора либо выдать по нему всю информацию.

**(\*)** Для удобного и быстрого занесения договоров в БД мы предварительно создаем отдельные сущности – отдельные компоненты договора. Например, арендатор, корпус.

Преимущество такого подхода состоит в возможности использовать повторяющиеся сущности от одного договора к другому, без необходимости их повторного ввода.  
Занеся однажды арендатора в список арендаторов, нам не придется заново вводить данные об этом арендаторе при последующем заключении новых договоров с этим арендатором.  
Это позволяет ускорить работу в дальнейшем.

Базовая теоретико-аналитическая часть

Анализ аналогов

1С: Предприятие

БД

Надо ли хранить данные в «Коммунальные услуги» типа «Начислено», если мы можем рассчитать их без хранения.

Счет скорей всего должен выступать в качестве таблицы БД, т.к. для его составления нужны реквизиты.  
А вот накопительная ведомость, мемориальный ордер и ведомость начисления – вычисляются исходя из множества уже существующих таблиц. То есть они просто объединяют в себе данные других таблиц и вычисление происходит на их основе, поэтому скорей всего не нужны в виде отдельных сущностей.  
Дебет и кредит тот же, вычисляется так же исходя из других таблиц. Имеет ли смысл выделять их в новые таблицы?  
Можно конечно выделить дебет и кредит в таблицу Счет, и сделать из нее просто общий сборник данных для всего экспорта во все счета и ведомости.

Вопрос – выводить счет на основе арендатора или договора? Где будет кнопка вывода счета?

Сотрудник напрямую не контактирует с арендатором – только с договором.  
И сущность арендатора позволяет посмотреть список его договором. Но не сформировать счет.  
Но можно выбрать его договор, и уже исходя из конкретного договора – вывести счет.  
У одного счета – лишь 1 договора. У 1 договора – множество счетов(за каждый месяц)

Настроено подключение приложения к СУБД h2 во встроенном режиме. В зависимости проекта папки lib установлен JDBC драйвер, обеспечивающий стандартизированный интерфейс для взаимодействия приложения к СУБД.  
Возможно в скором времени использование hibernate для более быстрого и удобного доступа к БД.

Для использования методов драйвера был выделен отдельный класс Database, который включает в себя логику работы с СУБД, а именно – подключение к ней.  
Планируется использовать порождающий паттерн проектирования «Одиночка», чтобы подключение к БД происходило единожды во всем приложении.

Информация для подключения к СУБД вынесена в отдельный конфигурационный database.properties файл, чтобы отделить логику работы программы от входящих данных, а также для удобного доступа к конфигурации проекта, без необходимости изменять код приложения.

На первое время для теста реализованы методы для добавления и получения простой выборки с БД для проверки работоспособности подключения.  
Но кроме этого, методы сделаны статическими и выделены в модель приложения – класс RenterModel, с задумкой на проектирование приложения по архитектуре MVC.  
Класс RenterModel будет охватывать всех методы для работы с данными арендаторов.

Логика работы вывода списка по нажатию на кнопку «Показать» выделена в метод UpdateListRenters, который добавлен также в логику для «Добавить» и «Удалить» арендаторов.  
UpdateListRenters занимается обновлением списка арендаторов для пользователя, через получение данных из RenterModel и удаление старых.

Созданы директории для контроллеров, моделей и представлений. Предполагается использовать архитектуру MVC.

Для конструирования меню и сайдбара написаны контроллеры MenuController и SidebarController, который содержат в себе порядок построения элементов для этих компонентов GUI.   
Предполагается заменить логику конструирования в файлы fxml.  
(Конец 23.10.2018)

ПТ - 26.10.2018

Осуществлен перенос элементов интерфейса из RenterController в отдельный файл представления renter.fxml. Таким образом контроллер стал более чистым и удобным для написания логики работы с сущностями арендаторов. Но пришлось изменить методы – их параметры и статичность, также добавить сеттер для добавления ссылки на БД к объекту класса.  
В виду вышеперечисленных изменений, соответственно, изменен вызов контроллера из SidebarController, так как в текущей реализации нам необходимо создавать объект класса RenterController, в отличии от предыдущей статической версии работы.

Из главного класса EconomistWorkstation удалена часть, ответственная за отображение страницы Renter, чтобы в скором времени, при необходимости, создать настоящую стартовую страницу приложения.

В планах расширить логику и представления для новых полей таблицы Renter, чтобы получить полностью сформированную сущность арендаторов.  
Таким образом логика (функционал создания арендаторов) будет завершена.  
Текущего уровня представления будет достаточно. Потом интерфейс будет улучшен.

Также надо сделать вывод арендаторов в списке арендаторов без кнопки показать.  
Потом сделать конкретную страницу об арендаторе для операций CRUD.

Ускорься - пора сдавать 2 главы первые, теор часть и про БД.   
Исследуй аналоги, алгоритм действий и расписывай. Потом прогу допишешь.

ПН 29.10.18

Убрана кнопка «Показать» и обеспечен вывод арендаторов.

ВТ 30.10.18

Так. Я перенес логику создания формы в отдельный контроллер RenterFormController и представление RenterForm для более удобной работы с кодом за счет декомпозиции сложной задачи и в виду некоторых сложностей, которые препятствовали объединению формы и списка арендаторов, так как форма предполагалась быть в отдельном окне.  
Объединить в теории можно, но будет большая путаница и нагромождение условных операторов.

Для каждого арендатора добавлена кнопка удаления с соответствующей обработкой.

Реализован порождающий паттерн проектирования «Одиночка» (Singleton) для единого доступа к базе данных из любого класса, без необходимости передавать объект базы данных от класса к классу через параметры.

СР 31.10.18

Создал новый класс (структуру) для хранения сущностей типа Renter.  
Добавлены поля для создания полноценной сущности арендатора Renter (не только имя)  
В RenterModel добавлена логика для добавления сущности Renter в БД, а также удаление её по id.  
RenterController. RenterFormController связывается с полями Renter.fxml.