Лабораторная работа 1

Постановка задача:

1. *Выбор* объектно-ориентированного языка и технологии для реализации системы межбанковский сообщений.
2. *Создание git-репозитория*. Наличие файла .gitignore для выбранной вами технологии обязательно (см. коллекцию). Попадание в репозиторий артефактов сборки (.dll, .obj, .class, .jar и т.д.) и служебных файлов среды разработки (.suo, .user и т.д.) недопустимо.
3. *Разработка* приложение «Управление финансовой системой», которая позволяет Пользователю выбрать любой банк, который зарегистрирован в системе, авторизоваться в нём под любой ролью и выполнить действия, разрешенные данной роли.

Требования к системе:

Приложение обязано быть спроектировано с использование паттернов проектирования (порождающих, структурных и поведенческих) и следовать принципам SOLID. Сам проект должен иметь логичную и чистую архитектуру.

Приложение должно выполнять следующие функции*:

1. Имитировать работу банковской системы с возможностями работы с вкладами клиентов:

- создание;
- хранение;
- снятие;
- перевод;
- накопление;
- блокировка;
- заморозка.

2. Реализовать возможности выдачи кредитов и рассрочек с индивидуальным и фиксированным процентом переплаты по следующим условиям:

- 3 месяца;
- 6 месяцев;
- 12 месяцев;
- 24 месяца;
- Более 24 месяцев.

4. Реализовать функционал зарплатного проекта для предприятия;

5. Реализовать возможность авторизации пользователей с ролями (приведены обязательные роли, но можно дополнить):

- *Клиент*
  - может зарегистрироваться в системе (требуется согласие менеджера);
  - может взаимодействовать со счетами (открывать, закрывать и т.д.);
  - оформлять кредиты и рассрочки (требуется согласие менеджера);
  - подать заявку на зарплатный проект от предприятия.
- *Оператор*
  - может просматривать статистику по движениям средств пользователей и 1 раз отменить** действие по счёту (любой перевод кроме снятия наличных);
  - подтверждает Заявку на зарплатный проект после получения данных от предприятия.
- *Менеджер*
  - функционал оператора;
  - подтверждение кредитов и рассрочек;
  - отмена** операций произведенным специалистом стороннего предприятия;

- *Специалист стороннего предприятия*
    - подача документов на зарплатный проект;
    - запрос на перевод средств другому предприятию или сотруднику его предприятия.
- *Администратор*
    - Просмотр всех логов действий (логи могут быть в отдельном файле и зашифрованы);
    - Отмена** всех действий пользователей.

6. Обязательные абстракции: Банк, пользователь, предприятие, счёт, кредит, рассрочка, перевод. Минимальные данные по клиенту:

- ФИО;
- серия и номер паспорта;
- идентификационный номер;
- телефон;
- email;
- предусмотреть возможность работы с иностранными клиентами (желательно отдельный механизм).

Минимальные данные по предприятию:

- тип (ИП, ООО, ЗАО и т.д.);
- юридическое название;
- УНП;
- БИК банка;
- юридический адрес;

Для демонстрации работы необходимо наполнить систему: 3 банками, 10 предприятиями и 100 клиентами (во всех банках). У предприятия только один банк и много счетов, у клиента может быть много банков и много счетов. Банк может являться

предприятием. Разрешается демонстрация работы в консоли, но в таком случае необходимо продумать понятный и простой UX.

*Данный материал предоставляет лишь ***минимальные*** и ***неполные*** требования к функционалу проекта.
**Для упрощения работы, можно отменить только *два* последних действия по *каждому* пользователю.

Сдача работы:

– предоставить документацию по проекту (UML диаграммы и небольшую пояснительную записку);

– провести декомпозицию предметной области и разработать диаграмму классов (class diagram);

– провести анализ взаимодействия объектов внутри системы и разработать диаграмму последовательности (sequence diagram);

- провести анализ сценариев взаимодействия пользователей с системой и разработать диаграмму вариантов использования (use case diagram);
- разработать и предоставить рабочее ПО.