4.4. ***Алгоритм работы сервиса***

В системе реализован паттерн API-Gateway, его суть заключается в том, что клиент отправляет запросы не в несколько разных МС с бизнес-логикой, а только в один МС (gateway), чтобы он уже перенаправлял запросы в другие МСы. Главная задача этого МС - инкапсулировать сложную логику всей внутренней системы, предоставив клиенту простой и понятный API. Таким образом все данные поступающие в систему, изначально проходят через Gateway, который в свою очередь отправляет параметры в нужный МС. Общение на прямую возможно только с базой данных и Kafka.

4.4.1. ***Прескоринг и формирование кредитных предложений***

1. Микросервис ***application***получает сообщение LoanApplicationRequestDTO от потребителя.
2. Микросервис ***application*** осуществляет прескоринг заявки в соответствии с правилами валидации **T\_01**.
   1. Если прескоринг прошёл успешно, то вызывается микросервис ***deal*** с сообщением LoanApplicationRequestDTO без изменений.
   2. Иначе появится ошибка **ERR\_1**.
3. Микросервис ***deal***сохраняет сообщение LoanApplicationRequestDTO в БД в соответствии с  **T\_02**
4. Микросервис ***deal***отправляет запрос в микросервис ***credit-conveyor***на генерацию кредитных предложений, передавая LoanApplicationRequestDTO без изменений
5. Микросервис ***credit-conveyor*** формирует 4 кредитных предложения LoanOfferDTO на основании всех возможных комбинаций полей isInsuranceEnabled и isSalaryClient, т.е. клиенту предлагается 4 вида предложений (от худшего в лучшему):   
     I. Без страхового пакета и клиент не является зарплатным (false-false)  
    II. Без страхового пакета и клиент является зарплатным (false-true)  
   III. Со страховым пакетом и клиент не является зарплатным ( true-false)  
   IV. Со страховым пакетом и клиент является зарплатным (true-true)
6. Микросервис ***credit-conveyor***отправляет сгенерированные кредитные предложения LoanOfferDTO в микросервис ***deal***, которые передает полученные значение в МС ***application***.
7. Микросервис ***application*** в свою очередь отправляет клиенту 4 подготовленных кредитных предложения и формируется ответ с HTTP = 200 с 4 предложениями в теле запроса.

При формировании кредитного предложения логика формирования **ставки** следующая:  Б(базовая ставка, 15%) - СЖ(страхование жизни, 4%) - ЗК(зарплатный клиент 1%).

**Стоимость страховки** рассчитывается по формуле: Б(базовая цена, 10000), если С (сумма займа) до 200 000 и Б+(С\*0.05), если С > 200 000.

4.4.2. ***Получение выбранного кредитного предложения***

1. Микросервис ***application*** получает сообщение LoanOfferDTO от потребителя, в котором передается только одно выбранное кредитное предложение и передает сообщение без изменений в микросервис ***deal***
2. Микросервис ***deal***обновляет статус заявки (APPROVED) и сохраняет сообщение LoanOfferDTO в БД в соответствии с  **T\_03**
3. Микросервис ***deal***вызывает микросервис ***dossier***передавая applicationId
4. Микросервис ***dossier***формирует письмо в соответствии с **T\_04**, с ссылкой на завершение регистрациии формируется ответ с HTTP = 200 с пустым телом запроса.

4.4.3. ***Скоринг***

1. Микросервис ***deal***получает сообщение ScoringDataDTO (FinishRegistrationRequestDTO) от потребителя.
2. В БД производится поиск заявки по полю applicationId из входящего сообщения
3. Сообщение ScoringDataDTO обогащается данными из найденной в БД заявки по applicationId, в соответствии с  **T\_05**
4. Микросервис ***deal*** передает обогащенное сообщение LoanApplicationRequestDTO (LoanApplicationRequestDTO обогащается данными из FinishRegistrationRequestDTO и получаем ScoringDataDTO согласно таблице из бизнес-параметров) в микросервис ***credit-conveyor***
5. Микросервис ***credit-conveyor***осуществляет скоринг заявки в соответствии с правилами валидации **T\_07**
   1. Если скоринг прошел успешно, то вызывается микросервис ***deal*** с сообщением CreditDTO без изменений и переходим к п.6
   2. Иначе появится ошибка **ERR\_1**.
6. Микросервис ***deal***обогащает сохраненную в БД по applicationId заявку из сообщения CreditDTO в соответствии с  **T\_06**
7. Если сообщение успешно сохранилось в БД, то микросервис ***deal***вызывает ***dossier***, передавая applicationId.
8. Микросервис ***dossier***формирует письмо в соответствии с **T\_04,** формируется ответ с HTTP = 200 с пустым телом запроса.

4.4.4. ***Отправка документов потребителю***

1. Микросервис ***deal***получает запрос от потребителя, на создание документов. В запросе передается applicationID.
2. Микросервис ***deal***отправляет запрос в базу данных, на изменения статуса заявки (Application/status/PREPARE\_DOCUMENTS)
3. Микросервис ***deal***отправляет запрос на отправку документов в микросервис ***dossier***, передавая обогащенный ApplicationDTO.
4. Микросервис ***dossier***обращается к МС **deal**с запросом на обновление статуса заявки (ApplicationStatusHistoryDTO/status/DOCUMENT\_CREATED).
5. Микросервис ***deal***отправляет запрос в базу данных, на изменения статуса заявки (Application/status/DOCUMENT\_CREATED)
6. Микросервис ***dossier***формирует письмо в соответствии с **T\_04**, отправляя клиенту на почту документы для подписания и ссылку на согласие с условиями и формируется ответ с HTTP = 200 с пустым телом запроса.  
         a. Если клиент согласен с условиями, МС ***deal***получает запрос с applicationID от потребителя, на получение кода и ссылки на подписания документов  
         b. Иначе появится ошибка **ERR\_1**.

4.4.5. ***Подписание документов и выдача кредита***

1. Микросервис ***deal***отправляет запрос в базу данных, на обновления поля Application/SES-code
2. Микросервис ***deal***отправляет запрос в микросервис ***dossier***передавая обогащенный ApplicationDTO.
3. Микросервис ***dossier***отправляет код и ссылку на подписание документов на почту клиента, в соответствии с **T\_04**и формируется ответ с HTTP = 200 с пустым телом запроса.
4. Микросервис ***deal***получает от клиента запрос с SES-code и проверяет его.  
         а. Если полученный SES-code совпадает с отправленным кодом клиенту, то:      
                   I. МС ***deal***отправляет запрос в БД на обновление статуса заявки (Application/status/DOCUMENT\_SIGNED)  
                  II. МС ***deal***отправляет запрос в БД на обновление статусы заявки (Application/status/CREDIT\_ISSUED и Credit/credit\_status/ISSUED)  
                 III. МС ***deal*** вызывает МС ***dossier***, который, в свою очередь, отправляет уведомление на почту клиента, в соответствии с **T\_04**, о выдаче кредита и формируется ответ с HTTP = 200 с пустым телом запроса.  
         b. Иначе появится ошибка **ERR\_1**.