|  |
| --- |
| Техническое задание на закупку  Услуг по разработке мобильного приложения «Помощник абонента» (My Ucell) |
|  |
|  |
| для нужд ООО «COSCOM» |

Оглавление

[**Общие сведения** 2](#_Toc87357968)

[**Требования по комплектации** 3](#_Toc87357969)

[**Общие требования к участнику** 3](#_Toc87357970)

[**Требования к обслуживанию и эксплуатации товара** 3](#_Toc87357971)

[**Требования на соответствие товара нормативным документам в области технического регулирования** 3](#_Toc87357972)

[**Документы для разработки** 4](#_Toc87357973)

[**Дополнительные требования** 4](#_Toc87357974)

[**Требования к API решению** 5](#_Toc87357975)

[**Требования к функциональным характеристикам** 5](#_Toc87357976)

[**Архитектура решения** 5](#_Toc87357977)

[**Требования к интеграции** 7](#_Toc87357978)

[**Требования к безопасности** 8](#_Toc87357979)

[**Требования к отчетности и логированию** 17](#_Toc87357980)

[**Структура и описание административного интерфейса** 17](#_Toc87357981)

[**Требование в команде участника** 21](#_Toc87357982)

[**Оплата труда** 21](#_Toc87357983)

[**Сроки выполнения работ** 21](#_Toc87357984)

[**Результаты проекта** 22](#_Toc87357985)

[**Требование по техническому обучению персонала** 22](#_Toc87357986)

[**Запуск мобильного, платежного приложения** 22](#_Toc87357987)

[**Требование к запуску на MVP (обозначено \* выше по тексту)** 23](#_Toc87357988)

Перечень принятых сокращений

|  |  |  |
| --- | --- | --- |
| № | Сокращение | Расшифровка сокращения |
| 1 | Аккаунт | Учетная запись пользователя |
| 2 | Face ID | сканер объёмно-пространственной формы лица человека |
| 3 | OTP (one time password) | одноразовый пароль, отправленный посредством SMS действителен только для одного сеанса авторизации. |
| 4 | Username | В системе у каждого пользователя может быть уникальное имя.  указать свой username для дальнейшего удобства переводов средств. |
| 5 | SMS, СМС | технология приёма и передачи коротких текстовых сообщений с помощью сотового телефона |
| 6 | USSD | стандартный сервис в сетях GSM, позволяющий организовать интерактивное взаимодействие между абонентом сети и сервисным приложением в режиме передачи коротких сообщений |
| 7 | SMS - информирование | это услуга, при помощи которой держатель банковской карты имеет возможность посредством SMS-сообщений получать на мобильный телефон оповещения об операциях на своем карточном счете. |
| 8 | БПК | Банковская пластиковая карта |
| 9 | \* | Обязательное требование при запуске MVP |

# **Общие сведения**

Данное техническое задание является первым этапом проекта: «Разработка собственного мобильного платежного приложения и платежного инструмента»

* 1. Цель разработки данного мобильного приложения

Основная цель — запуск мобильного приложения с персональным кабинетом и с базой мерчантов, партнеров (банков, страховых компаний и пр), функционирующей системой обработки платежей, наработанная партнерская база. В рамках данной задачи необходимо разработать, запустить и обеспечить интеграцию с билингом компании и комплексной процессинговой платформы для обработки платежей пластиковых карт МПС Uzcard, Хумо.

* 1. Основание для реализации проекта, в рамках которого разрабатывается Мобильное приложение

Цели и преимущества от реализации проекта:

• развитие фин. тех направление на базе компании

• предоставить абонентам мобильный персональный кабинет

• дополнительный канал монетизации

• возможность реализации программу лояльности для абонентов компании

# **Требования по комплектации**

Решение должно иметь полноценную комплектацию программной части для возможности полноценного предоставления сервиса и услуг.

# **Общие требования к участнику**

Поставщику необходимо заполнить файл «Перечень требований (рус).xlsx» на соответствие техническим требованиям для проекта «Услуг по разработке мобильного приложения «Помощник абонента» (My Ucell). Поставщик не ограничен в предоставлении дополнительных материалов по техническому решению для платформы.

Заказчик в лице ООО «Coscom» имеет право в одностороннем порядке решать соответствие поставщика техническим требованиям, указанным в файле «Перечень требований (рус).xlsx» без объяснения причин.

# **Требования к обслуживанию и эксплуатации товара**

Поддержка программного комплекса, а также его сервисное обслуживание должно осуществляться в рамках заключенного договора. А так же для развития проекта в дальнейшем, все исходные должны быть переданы разработчикам компании.

# **Требования на соответствие товара нормативным документам в области технического регулирования**

Программный комплекс решения должен соответствовать всем имеющимся нормативным требованиям безопасности в соответствии с законами РУз об информационной безопасности, а так же, соответствовать регламенту об информационной безопасности утвержденному внутри компании.

Группы пользователей. [\*](#_Требование_к_запуску)

В рамках проекта предусмотрены следующие группы пользователей:

Администратор — администратор может добавлять/удалять/изменять существующих пользователей. Назначать или изменять роли пользователей. контролировать учетные записи и элементы системы. Управлять модулями и документации.

Клиент — пользователь: любой человек.

Данная группа пользователей представляет основную массу пользователей проекта.

# **Документы для разработки**

Ожидается предоставление технической, проектной документации, содержащей:

• детальную конфигурацию и технические характеристики оборудования необходимого для развертки решения на виртуальной среде и программного комплекса;

• схемы взаимодействия элементов решения в сети;

• перечень (спецификацию) рекомендуемого оборудования необходимого для развертки решения на виртуальной среде предлагаемого программного обеспечения;

• функциональное описание (описание модулей, функциональных блоков, программного обеспечения);

• техническое описание решения HLD/LLD;

• проектное решение исходя из требований законодательства РУз;

• программу и методику приёмочных испытаний.

• полноценное руководство по эксплуатации внедряемого решения Site Folder.

Требование к языку разработки и архитектуры

• Android – Kotlin

• IOS – Swift

• Backend – java/python

• Поддержка микросервисов и управление API

# **Дополнительные требования**

• Поддержка персонального кабинета абонента, с выводом на экран: балансы, статус, текущие показания билинговой информации. [\*](#_Требование_к_запуску)

• Чат-бот – возможность пользователю приложения создать чат с оператором, для быстрого решения возникшего вопроса. На стороне оператора в веб приложении приходит уведомление о новом сообщении от пользователя приложения, в котором оба могут общаться. Функция поддержки отправки файла (картинка, видео размером не более 10мб). Все чаты с пользователями должны логироваться и оставаться в системе, а роль «администратор» иметь возможность просматривать переписки с пользователями но номеру абоненту или по логину оператора. Данная функция должна существенно снизить нагрузку на CallCenter по вопросам работы с приложением.

• Поддержка рекламных банеров, с регулируемым или автоматическим обновлением, через админ панель. [\*](#_Требование_к_запуску)

• Возможность смены ТП [\*](#_Требование_к_запуску)

• Возможность воспользоваться услугой «резервная сим карта»

• Возможность выгрузки или просмотра детализации звонков. [\*](#_Требование_к_запуску)

• Поддерживать регистрацию банковских карт. [\*](#_Требование_к_запуску)

• Поддерживать списание средств с банковских крт. [\*](#_Требование_к_запуску)

• Предоставлять возможность P2P переводов с одной банковской карты на другую. [\*](#_Требование_к_запуску)

• Предоставлять возможность блокировку\разблокировку банковской карты.

• Поддерживать сверку операций, проведенных по банковские карты.

• Поддерживать проверку баланса пластиковой карты. [\*](#_Требование_к_запуску)

• Поддерживать отмену транзакции по банковской карте.

• Назначение бонусов, управление системой лояльности - система должна поддерживать функционал вознаграждение пользователей в виде баллов за каждую проведенную банковскую операцию, данный функционал должен быть гибким[\*](#_Требование_к_запуску).

Возможные виды бонусов:

1. Пользователь получает баллы за использование приложение каждый день(раз в сутки);

2. Пользователь получает баллы за оплату определённых сервисов и\или мерчантов;

3. Пользователь за баллы может оплатить услуги\пакеты компании;

\* Система должна поддерживать назначение количество баллов и их стоимости через панель администратора на каждого сервисов и\или мерчанта отдельно.

А также система должна поддерживать назначение процентов кешбека на каждый мерчант отдельно через панель администратора.

!!!Доступность решение 99.95% при нагрузке.

# **Требования к API решению**

Решение должно предоставлять API для интеграции с внешними (3rd party) системами.

Решение должно поддерживать API методы для интеграции с PGW.

Решение должно поддерживать универсальный API методом для интеграции с другими платежными системами.

Требования к надежности.

• Приложение должно работать и поддерживаться 24/7. [\*](#_Требование_к_запуску)

• Сервера должны находиться в балансере, в случае отказа принимать нагрузку на последующий. [\*](#_Требование_к_запуску)

*Общие требования к дизайну экранов пользователя.*

Приложение должно обладать современным дизайном, а также продуманным и удобным, с точки зрения пользователя. [\*](#_Требование_к_запуску)

# **Требования к функциональным характеристикам**

Доступ к приложению предоставляется по четырехзначному паролю или по опечатку пальцев или по FaceID (если она поддерживается устройством пользователя) [\*](#_Требование_к_запуску)

Банковские данные, персональные данные не должны оставаться в памяти оборудования пользователя. [\*](#_Требование_к_запуску)

# **Архитектура решения**

Система должна эффективно сочетать возможности традиционных и одностраничных веб -приложений, т.е., использовать рендеринг статического контента на стороне сервера и динамическую загрузку данных для реализации интерактивного поведения.

Архитектура системы состоит из следующих основных компонентов:

Клиент (слой клиента) – это интерфейсный (обычно графический) компонент системы, предоставляемый конечному пользователю. Этот уровень не должен иметь прямых связей с базой данных (по требованиям безопасности и масштабируемости), быть нагруженным основной бизнес-логикой (по требованиям масштабируемости) и хранить состояние приложения (по требованиям надёжности).

На этот уровень обычно выносится только простейшая бизнес-логика: интерфейс авторизации, проверка вводимых значений на допустимость и соответствие формату, уже загруженными на клиентское устройство.

Сервер приложений (средний слой, связующий слой) располагается на втором уровне, на нём сосредоточена большая часть бизнес-логики. Реализация данного компонента обеспечивается связующим программным обеспечением.

Центральным компонентом сервера приложений является веб-сервер – программное обеспечение, предназначенное для приема HTTP-запросов от клиентов (обычно веб-браузеров) и выдающее им HTTP-ответы, как правило, вместе с HTML-страницей, изображением, файлом, медиа-потоком или другими данными. Основным назначением веб-сервера является генерирование статического и динамического контента для загрузки на устройства пользователей.

Сервер приложений должен обеспечивать возможность добавления к нему дополнительных экземпляров (микросервисы) в целях горизонтального масштабирования производительности программного комплекса без необходимости внесения изменений в программный код приложения.

Файловый сервер – должен предоставлять приложению ссылки на документы, графику и прочие файлы. Он нужен для того, чтобы уменьшить объем приложения и подгружать все файлы онлайн в кэш приложения.

Основные модули мобильного приложения:

Модуль Cabinet [\*](#_Требование_к_запуску)

Персональный кабинет – этот модуль должен предоставлять билинговые данные по клиенту:

• Балансы абонента в разрезе тарифного плана

• Текущий тарифный план

• Управление тарифным планом (смена)

• Текущие услуги

• Управление услугами

• Просмотр/получение детализации звонков

• Предоставление текущего статуса

• Управление резервной сим картой

Модуль user [\*](#_Требование_к_запуску)

Пользователь – это модуль должен содержать данные пользователей, с функциями создания пользователя и персонализации приложения. Этот модуль должен решать задачи такие как: регистрация и восстановление аккаунта (с последующим удалением сохраненных карт) внутри системы, добавление карт оплаты; виджеты быстрой оплаты мобильной связи, списка сохраненных платежей и недавних получателей; управление данными и настройкой личного кабинета пользователя и устройства, настройки безопасности

Модуль card [\*](#_Требование_к_запуску)

Карта – в этом модуле должно быть управление картой, добавление, управление правилами доступа для других пользователей этой же карты, выбор основной карты, вывод и скрытие отображения баланса, блокировка и разблокирование возможности использования карты; должно быть получение отчета по поступлениям средств и их расходу за выбранные периоды;

Модуль payment [\*](#_Требование_к_запуску)

Оплата – в этом модуле должна возможность оплаты по различным категориям, как «Мобильные операторы», «Интернет провайдеры» и т.д., должен быть возможность поиска и сортировки по категориям, создания авто-платежей с возможностью выбора временного промежутка для постоянных оплат сервисов и услуг; перевод средств и их запрос у другого пользователя используя номер карты, пользовательское имя или сгенерированный временный QR-код, с возможностью его отправки через сторонние приложения.

# **Требования к интеграции**

Интеграция должна учитывать особенности работы с новым биллингом и их архитектурой, а также с банковскими, процессинговыми, платежными системами. [\*](#_Требование_к_запуску)

**Требования к проекту и программному обеспечению:**

Язык интерфейса приложения: русский, узбекский, английский [\*](#_Требование_к_запуску)

Разработанное программное обеспечение (далее ПО) должно быть:

- совместимо с устройствами на базе ОС Android 4.0 (и выше), iOS 10 и выше телефон, планшет;

Веб-часть администратора системы

- Главная страница [\*](#_Требование_к_запуску)

- Подсистема история транзакции [\*](#_Требование_к_запуску)

- Подсистема отчет

- Подсистема Скоринг

- Управление марчантами

- Управление бонусированием для мерчантов

- Иметь возможность изменять/добавлять бандлы [\*](#_Требование_к_запуску)

Функционал приложения off-line. [\*](#_Требование_к_запуску)

В Offline режиме должна работать следующий список функций

.

• Проверка аккаунта;

• Регистрация пользователей;

• Авторизация пользователей;

• Активация аккаунта;

• Восстановление пароля;

• Удаление аккаунта;

• Изменение пароля;

• Удаление карт ;

• Вкл/отк. Сервисов;

• Информация о карте;

• Выбор основной карты;

• Отправить запрос средств;

• Отправить и добавить запрос оплаты;

• Получить запрос оплаты;

• Получить список всех запрос оплаты;

• Проверка карты;

• Оплата услуг поставщика;

• Сохранение платежа (избранное)

• Обращение к службе поддержки.

В Offline режиме должна быть возможность выполнять USSD команды. [\*](#_Требование_к_запуску)

# **Требования к безопасности**

Все компоненты приложения должны быть определены и проверены на их необходимость в использовании.

• Доступ к приложению должен быть через Логин/пароль и ни не должны автоматически заполнятся и хранится.

• Все компоненты, такие как библиотеки, модули и внешние системы, не являющиеся частью приложения, но необходимые для работы приложения, определены.

• Все меры безопасности (включая библиотеки обращающиеся к сервисам безопасности) применены централизованно.

• Компоненты сегрегированы друг от друга с помощью определенных мер безопасности, к примеру сетевой сегрегации, правил на файерволах, и т.д.

• В коде на стороне клиента не хранится информация о бизнес логике, секретных ключей или другой запатентованной информации.

• Все компоненты приложения, библиотеки, модули, фреймворки, платформа и операционные системы не имеют известных уязвимостей.

• Все страницы и ресурсы для доступа должны требовать аутентификацию, за исключением тех, которые предназначены для общего доступа (принцип проверки каждого разрешения на доступ механизмом управления доступом).

• Приложение не заполняет информацию об учетных записях автоматически, либо как скрытые поля, аргументы в URL, запросы Ajax, или в формах, т.к. это подразумевает передачу в открытом тексте, обратимом или подверженном дешифрации методе хранения. Одноразовый код, ограниченный по времени допустим в качестве подстановки, для защиты формы смены паролей.

• Все контроли аутентификации применяются на стороне сервера.

• Все контроли аутентификации имеют отказоустойчивость мер защиты для обеспечения невозможности входа злоумышленников.

• Поля ввода паролей позволяют или рекомендуют использование парольных фраз, а также не препятствуют вводу из менеджеров паролей, длинных фраз паролей или особенно комплексных паролей.

• Все функции идентификации учетной записи (к примеру обновления профиля, смена забытого пароля, выключение / утеря токена, и т.д.), которые помогают заново получить доступ к учетной записи, должны быть как минимум устойчивы к атакам, как и основной механизм аутентификации приложения.

• Функционал по смене пароля должен включать подтверждение старого пароля, ввод нового пароля с его подтверждением.

• Все решения аутентификации должны заноситься в журнал аутентификации, без хранения конфиденциальной информации, к примеру паролей, или идентификаторов сессий. Журналирование должно включать все необходимые для расследования безопасности метаданные.

• Пароли учетных записей хэшируются с помощью односторонних функций с использованием “соли”, а также учтен фактор требуемых временных затрат на перебор паролей или попыток проведения атак на поиск аргументов односторонних функций.

• Информация об учетных записях, а также все страницы/функции, требующие от пользователя ввода подобной информации, используют шифрованные каналы передачи данных.

• Функция смены забытого пароля и другие пути его восстановления не должны открыто показывать действующий пароль, а также новый пароль не отсылается пользователю в открытом виде.

• Получение скрытой информации невозможно с помощью использования функционала сброса пароля, входа в приложение или смены забытого пароля.

• Пароли, установленные по умолчанию, не используются для структуры приложения или любого компонента, используемого приложением (к примеру комбинация admin/password).

• Применены меры по предотвращению автоматизации перебора паролей, тестирования скомпрометированных учетных записей, атак направленных на блокирование учетных записей.

• Вся информация учетных записей, имеющая отношение к аутентификации для доступа сервисов внешних для приложения, зашифрована и хранится в защищенном хранилище.

• Забытые пароли и другие пути их восстановления используют одноразовые пароли или любые другие токены, оповещения на мобильном телефоне, или другие автономные средства восстановления. Использование случайно сгенерированного значения в почтовом или СМС сообщении должно быть только крайней мерой.

• Блокирование учетной записи должно быть разделено на “жесткое” и “мягкое” блокирование, и также не быть взаимно исключаемым. Если учетная запись “мягко” временно заблокирована в следствии атаки перебора паролей, она не должна сбрасывать “жесткую” блокировку.

• Вопросы, основанные на знании известной информации (также известные как “секретные вопросы”) не должны нарушать требования конфиденциальности личной жизни и должны адекватно защищать учетные записи от восстановления пароля злоумышленником.

• Приложения имеющие особую ценность для Компании должны иметь конфигурацию, позволяющую установить запрет на использование определенного количества уже использованных паролей.

• Для особо важных транзакций есть возможность прохождения дополнительной аутентификации, использования дополнительного атрибута аутентификации или подпись транзакции.

• Внедрены меры блокировки использования слабых паролей или парольных фраз.

• Пользователи могут запросить и использовать подтверждение с использованием одноразовых паролей, дополнительных факторов аутентификации, биометрических (Touch ID или похожих) методов аутентификации, или эквивалентного многофакторного механизма аутентификации.

• Административные интерфейсы не должны быть доступны для не доверенных сторон.

• Приложение должно быть совместимо с парольными менеджерами третьих сторон или используемых в браузерах, за исключением случаев, запрещенных политиками.

• Менеджеры сессий, специально изменённые или написанные для приложения, не используются или устойчивы ко всем известным атакам на управление сессиями.

• Сессии должны становиться недействительными после выхода пользователя.

• Сессии должны становиться недействительными после определенного периода бездействия.

• Сессии должны становиться недействительными после определенного максимального периода времени независимо от того активная сессия или она находится в состоянии бездействия.

• Все страницы для которых требуется аутентификация имеют легко и визуально доступный функционал для выхода из приложения.

• Идентификатор сессии никогда не должен разглашаться в составе URL, сообщениям об ошибках, или логах. Это требование также включает подтверждение того, что приложение не поддерживает возможность перезапись сессионных куки через URL.

• Все успешные попытки новой или повторной аутентификации должны генерировать новую сессию и новый идентификатор сессии.

• Только идентификатор сессии, сгенерированный приложением должен распознаваться приложением как активный.

• Идентификатор сессии должен быть достаточно длинным, случайным и уникальным во всей базе активных сессий.

• Идентификатор сессии, хранящийся в куки должен иметь путь, установленный в адекватно ограниченном значении для приложения, и токены сессий аутентификации должны иметь флаги “HttpOnly” и “secure”.

• Список активных сессий должен отображаться в профиле пользователя и другом месте приложения. Пользователь должен иметь возможность удаления активной сессии.

• Используется принцип наименьших привилегий – пользователи могут использовать функции, файлы данных, URL, сервисы и т.п. только на основании полученной авторизации для их использования. Это подразумевает защиту от спуфинга и несанкционированного повышения привилегий.

• Доступ к конфиденциальной информации защищен, таким образом пользователю доступны только авторизованные объекты или данные (к примеру, защита от изменения пользователем параметров для просмотра или изменения учетной записи другого пользователя).

• Возможность просмотра каталога выключена, за исключением случаев, когда подобная возможность требуется. Также, приложения не должны разрешать раскрытие метаданных файлов или каталогов, таких как Thumbs.db, .DS\_Store, .git или .svn .

• Контроли доступа имеют отказоустойчивость с учетом требований безопасности.

• Похожие правила контроля доступа на уровне презентации должны обрабатываться на стороне сервера.

• Все атрибуты учетных записей или данных, а также политики, используемые контролями доступа, не могут меняться пользователя без авторизации.

• Все события по контролю доступа могут быть занесены в журнал событий и все неуспешные события должны журналироваться.

• Приложение или его структура использует устойчивый и случайный токен для предотвращения межсайтовой подделки запроса или предоставляет другой механизм защиты транзакций.

• Система имеет защиту от суммарного или продолжительного доступа к защищенным функциям, ресурсам или данным. К примеру, можно ограничить количество изменений в час или запретить удаление всех базы для одного пользователя.

• Приложение имеет функцию дополнительной авторизации (к примеру, адаптивная аутентификация или подтверждение для всех приложений, а также полное разделение полномочий для приложений, имеющих особую ценность для Компании, как мера защиты от мошенничества на основании предыдущих случаев или анализа рисков.

• Приложение корректно обрабатывает запросы авторизации в зависимости от контекста, таким образом, не позволяя неавторизованные изменения через модификацию параметров запроса.

• Среда в которой исполняется приложение не подвержена атакам переполнения буфера или применяет меры защиты от подобных атак.

• Неуспешные проверки ввода данных на стороне сервера, которые проявляются в отказе запроса должны журналироваться.

• Проверки ввода данных должны проходить на стороне сервера.

• Все SQL запросы, HQL, OSQL, NOSQL и сохраненные процедуры, а также вызов сохранённых процедур защищены использованием заранее подготовленных запросов или контролем параметров набора запросов, таким образом предотвращая атаки с использованием SQL инъекций.

• Приложение не подвержено атаке с использованием LDAP инъекций, либо применяет меры защит от подобных атак.

• Приложение не подвержено атаке с использованием инъекций в командную строку, либо применяет меры защит от подобных атак.

• Приложение не подвержено атакам инъекции удалённых или локальных файлов при использовании контента в виде пути к файлу.

• Приложение не подвержено общеизвестным атакам на XML, таким как модификация параметров запроса XPath, инъекции XML и т.п.

• Все переменные строчного типа, помещенные в код страницы HTML или другого веб приложения, проходят процедуру корректной кодировки в зависимости от контекста либо вручную, либо с помощью шаблонов, для предотвращения атак XSS.

• Если приложение позволяет автоматическое присваивание большого количества параметров модели (также известное как автоматическое привязывания переменных) из входящего запроса, поля, которые могут быть конфиденциальными с точки зрения безопасности (“accountBalance”, “role” или “password”) защищены от несанкционированного автоматического привязывания.

• Приложение защищено от атак загрязнения параметров HTTP, в частности если приложение не различает источник параметров запроса (GET, POST, куки, заголовки, среды обработки, и т.п.).

• Проверки на стороне клиента приложения должны использоваться как вторая линия защиты, в дополнение к проверке на стороне сервера.

• Все вводимые данные должны проверяться, включая источники ввода данных (к примеру, REST запросы, параметры запросов, заголовки HTTP, пакетные файлы, RSS источники и т.п.) с использованием позитивных проверок (белые списки), затем менее предпочтительных форм проверок, таких как серые списки (устранение известных плохих запросов), или отказ в обработке вводимых данных (черные списки).

• Структурированные данные должны строго проверяться на соответствие определённой схеме, включая разрешенные символы, длину и шаблон (к примеру, номер банковской карты или телефона, или проверка соотносящихся полей, к примеру совпадение номера почтового индекса с адресом).

• Неструктурированные данные должны проходить проверку для обеспечения мер безопасности, включая разрешенные символы, длину, также символы потенциально опасные для работы должны исключаться исходя из контекста (к примеру, имена в кодировке Unicode или апострофы, как-то ねこ или O'Hara).

• HTML код сформированный редакторами WYSIWYG или похожим ПО, должен проходить проверку и обработан в соответствии с функцией проверки ввода данных и их кодировки.

• Там, где приложение использует шаблоны авто-исключений и авто-исключения выключены, вывод данных должен вручную кодироваться в зависимости от контекста для предотвращения XSS.

• Там, где данные передаются из одного контекста DOM в другой, процедура передачи использует безопасные Javascipt методы, к примеру, innerText или .val .

• Во время обработки JSON в браузере на клиенте необходимо использовать JSON.parse. eval() не должен использоваться для обработки JSON на клиенте.

• Данные аутентификации должны очищаться на клиенте, как-то DOM браузерa, после удаления сессии.

• Все криптографические модули имеют отказоустойчивость с учетом требований безопасности, и ошибки обрабатываются с условием невозможности атаки Padding Oracle.

• Все числа, названия файлов, GUID, строки, сгенерированные случайным образом с использованием проверенного модуля генерации случайных чисел, в тех случаях где данные значения, полученные случайной генерацией не должны быть предугаданы злоумышленниками.

• Криптографические алгоритмы, используемые приложением проверены на соответствие адекватному стандарту.

• Существует политика управления криптографическими ключами (как-то генерация, распределение, отзыв и т.п.). Жизненный цикл ключа адекватно обеспечен.

• Конфиденциальная и личная информация хранится в зашифрованном виде.

• Для предотвращения атак на сброс памяти, пароли или ключи, содержащиеся в памяти, перезаписаны нулями, как только необходимость в них отпадает.

• Все ключи и пароли должны быть заменяемы, и сгенерированы или заменены во время установки приложения.

• Приложение не должно выводить конфиденциальную информацию, помогающую злоумышленнику, в сообщениях об ошибках или данных отслеживания стека, включая идентификатор сессии, версии приложения или его структуры, а также личную информацию.

• Логика обработки ошибки должна по умолчанию запрещать доступ к себе в контролях безопасности.

• Меры безопасности журналируют успешные и особенно неуспешные события, определенные как имеющие отношение к безопасности.

• Каждое событие в журнале должно включать необходимую информацию для детализированного расследования событий по времени их происхождения.

• Все события, которые включают данные из не доверенных источников не должны исполняться в виде кода при использовании ПО для просмотра журналов аудита.

• Журналы событий безопасности защищены от изменения и несанкционированного доступа.

• В соответствии с законодательством, приложение на журналирует конфиденциальную информацию и личную информацию, или информацию, которая может помочь злоумышленнику, включая идентификатор сессии, пароли, хэши или токены API.

• Журнал событий имеет возможность предотвращения отказа от транзакций.

• Источники времени синхронизированы на корректное время и зону времени.

• Все формы содержащие конфиденциальную информацию отключают возможность кэширования на стороне клиента, включая функционал авто-заполнения полей.

• Вся конфиденциальная информация отсылается через основное сообщение HTML или его заголовок, таким образом информация не передается через URL.

• Приложение устанавливает достаточно мер предотвращения кэширования заголовков, так что любая конфиденциальная или личная информация не должна кэшироваться на диске основными браузерами (к примеру, в Chrome можно просмотреть кэш зайдя на about:cache).

• На сервере, все кэшированные или временные сохраненные копии конфиденциальных данных защищены от несанкционированного доступа или удаляются после санкционированного доступа к ним.

• Приложение уменьшает количество параметров в запросе, к примеру скрытые поля, переменные Ajax, куки и значения заголовков.

• Данные хранимые на стороне клиента (к примеру, локальное хранилище HTML5, хранилище сессий, база данных индексов, обычные куки или Flash куки) не содержат конфиденциальной или личной информации.

• Доступ к данным должен журналироваться, если данные находятся под защитой законодательства или там, где это необходимо.

• Конфиденциальная информация, хранимая в памяти, перезаписывается нулями, как только необходимость в ней отпадает, для предотвращения атак на сохранение памяти.

• Для каждого TLS сертификата может быть построен путь до доверенного центра, а также каждый сертификат действителен.

• Для всех каналов передачи данных (внешних и внутренних) используемых для аутентификации или передачи конфиденциальной информации или функций используется TLS, а также существует отказоустойчивость с учетом требований безопасности, таким образом предотвращая использования нешифрованных каналов. Наиболее сильная альтернатива должна иметь предпочтение при выборе алгоритма шифрования.

• Все подключения ко внешним системам, которые имеют отношение к конфиденциальной информации или функциям должны быть аутентифицированы

• Технология HPKP должна применяться для публичных ключей в производственной и резервной среде.

• Заголовки HTTP Strict Transport Security должны включаться во все запросы и для всех доменов нижнего уровня, к примеру: Strict-Transport-Security: max-age=15724800; includeSubdomains.

• Для предотвращения записи трафика злоумышленниками рекомендуется конфигурация Perfect forward secrecy.

• Должа быть настроена возможность правильного отзыва сертификатов, к примеру, метод Online Certificate Status Protocol (OCSP) Stapling.

• Должны использоваться только устойчивые алгоритмы, шифры и протоколы, во всей иерархии сертификатов, включая корневые и посреднические сертификаты для выбранного центра сертификации.

• Настройки TLS должны соответствовать установленным рекомендованным практикам, особенно при уменьшении безопасности отдельных настроек, шифров или алгоритмов.

• Приложение должно принимать только определенный набор требуемых методов HTTP запросов, к примеру, GET или POST принимаются, а не используемые методы (TRACE, PUT, и DELETE) явно заблокированы.

• Каждый HTTP запрос содержит заголовок определяющий тип контента и безопасный набор символов (к примеру, UTF-8, ISO 8859-1).

• Заголовки HHTP, добавленные доверенными прокси или SSO устройствами, к примеру, токеном хранителя, проходят аутентификацию в приложении.

• Подходящий заголовок X-FRAME-OPTIONS используется для сайтов где контент не должен просматриваться в X- Frame третьей стороны.

• Заголовки HTTP или любая часть ответа HTTP не должна выдавать детализированной информации о версиях компонентов системы.

• Все ответы API содержат X-Content-Type-Options: nosniff и Content-Disposition: attachment; filename="api.json" (или другое имя файла в зависимости от контента).

• Политика безопасности контента (CSPv2) настроена для помощи в устранении общеизвестных DOM, XSS, JSON и JavaScript инъекций-уязвимостей.

• Заголовок X-XSS-Protection: 1; mode=block должен применяться для применения фильтров браузера для отраженных XSS атак.

• Приложение должно обрабатывать бизнес логику только в последовательном порядке, со всеми шагами, обрабатывающимися в реальном времени, а также не должны обрабатываться шаги вне порядка, пропущенные шаги, шаги обработки от стороннего пользователя, либо слишком быстро поданные запросы транзакций.

• Приложение должно иметь ограничения, основанные на бизнес логике, которые оно обеспечивает для каждого отдельного пользователя, с помощью оповещений или автоматизированных ответных мер на нешаблонные или автоматизированные атаки.

• Редиректы или переводы URL должны производится только на адреса, помещенные в белые списки, либо показывать предупреждение до редиректа на страницы с потенциально не доверенным контентом.

• Для предотвращения атак просмотра каталогов, включения локальных файлов, неправильной обработки MIME типа файла, отраженной загрузки файла, и инъекции команд операционной системы, не доверенные файлы, загружаемые в приложение не должны использовать прямые команды ввода/вывода.

• Файлы, полученные из не доверенных источников должны проверяться на совпадение ожидаемого типа файла и сканироваться антивирусным ПО для предотвращения загрузки общеизвестного вредоносного контента.

• Для предотвращения атак с использованием уязвимостей удаленного / локального исполнения кода, данные, которым нельзя доверять, не должны использоваться в функционале включения, отображения или загрузчика классов.

• Для предотвращения атак с использованием произвольного удалённого контента, данные, которым нельзя доверять, не должны использоваться в кросс-доменном разделении ресурсов (CORS).

• Файлы, полученные из не доверенных источников должны храниться вне корневого каталога веб ресурса, с ограниченными разрешениями, а также предпочтительно с усиленными проверками.

• Веб сервер или сервер с приложением должен быть настроен на отказ в доступе для удаленных ресурсов или систем внешних для веб-сервера или сервера приложения.

• Код приложения не должен исполнять файлы, полученные от не доверенных источников.

• Не рекомендуется использовать технологии на стороне клиента, которым нельзя доверять, или которые являются не безопасными или устаревшими, к примеру плагины NSAPI, Flash, Shockwave, Active-X, Silverlight, NACL, или Java апплеты.

• Идентификаторы, к которым есть доступ из других приложений и хранимые на устройствах, к примеру, UDID или IMEI не должны использоваться в качестве токенов аутентификации.

• Мобильные приложения не должны хранить конфиденциальные данные в местах с разделенными ресурсами (SD карты или общие папки) без применения шифрования.

• Конфиденциальные данные не должны храниться не защищенными на устройствах, даже в зонах, защищенных системой, к примеру Цепочки ключей.

• Секретные ключи, токены API или пароли должны динамично генерироваться на устройстве.

• Мобильное приложение должно предотвращать утечку конфиденциальной информации (к примеру, сохранение скриншотов приложения на момент его нахождения в неактивном состоянии на экране или запись информации в консоли.

• Приложение требует минимальные разрешения для функциональности или требуемых ресурсов.

• Код особо важный для приложения должен храниться в памяти случайным образом (к примеру, ASLR)

• Приложение не использует исполнение особо важного функционала, вызовов или контента с помощью других приложений на устройстве.

• Для предотвращения записи памяти, конфиденциальная информация, хранящаяся в памяти должна перезаписываться нулями, как только необходимость в ней отпадает.

• Приложение проверяет ввод данных при исполнении особо важного функционала, вызовов или контента.

• Единый стиль кодировки должен использоваться на сторонах сервера и клиента.

• Доступ к функционалу администрирования и управления веб сервисом, обслуживающим приложение должен быть ограничен администраторами веб сервиса.

• Схемы XML или JSON должны использоваться для проверки ввода данных.

• Любой ввод данных ограничен необходимым размером.

• SOAP сервисы на основе веб технологий дольны соответствовать требованиям Web Services-Interoperability (WS-I) основного профиля. В частности, это означает шифрование TLS.

• Использование аутентификации и авторизации на основе сессий должно соответствовать требованиям данного чек листа. К примеру, необходимо избегать использования статичных ключей API и т.п.

• Сервис REST должен быть защищен от подделки межсайтовых запросов при помощи: ORIGIN проверок, шаблонов двойного предоставления куки, одноразовых кодов CSRF, и проверок направляющего ресурса.

• Сервис REST должен прямо проверять входящий заголовок Content-Type на соответствие того, что ожидается, к примеру, application/xml или application/json.

• Полезная нагрузка сообщения должна быть подписана для обеспечения надежной передачи данных между клиентом и сервисом, используя JSON Web Signing или WS-Security для SOAP запросов.

• Альтернативные или менее безопасные каналы передачи данных не должны использоваться.

• Все компоненты должны иметь последние версии, обновления и настройки безопасности. Это включает удаление или отключение ненужных настроек или папок, к примеру, тестовые приложения, техническую документацию и учетные записи, установленные по умолчанию.

• Каналы передачи данных между компонентами, к примеру сервером приложения и сервером базы данных, должны быть зашифрованы, особенно когда компоненты расположены в отдельных контейнерах или системах.

• Все компоненты приложения, сервисы и серверы используют отдельные сервисные учетные записи с наименьшими привилегиями.

• Интерфейсы отладки на уровне приложения, такие как USB или серийные порты должны быть отключены.

• Криптографические ключи должны быть уникальными для каждого отдельного устройства.

# **Требования к отчетности и логированию**

Для оптимизации работы приложение, оценки его эффективности и получения маркетинговых метрик будет необходима следующая отчетность:

* Количество новых и текущих пользователей
* Логирование действий пользователей (все действия) [\*](#_Требование_к_запуску)
* Количество обращений к приложению (в целом и в среднем) за запрашиваемый период времени[\*](#_Требование_к_запуску)
* Количество переходов по каждой команде за запрашиваемый период
* Количество активаций сервисов через приложения процент удачных платежей и ошибок[\*](#_Требование_к_запуску)
* Количество смены ТП
* Количество запросов и их успешность по детализации звонков
* Детальный отчет по всем действиям: [\*](#_Требование_к_запуску)
* Оплата услуг (отчёт по QR коду отдельно);
* P2P перевод;
* Отмененные транзакции;
* Перерегистрация;
* Информация по запросам;
* Рейтинг лояльности.

Необходима возможность экспорта статистики в excel файл.

!!! Должна быть настроена ELK система, для мониторинга и поиска проблем, для дальнейшего решения. [\*](#_Требование_к_запуску)

# **Структура и описание административного интерфейса**

Требования к интерфейсу

Административный интерфейс должен быть интуитивно понятен и реализован на узбекском\русском языке. Главная страница должна содержать следующие рабочие области:

1. Навигационная панель по разделам системы; [\*](#_Требование_к_запуску)

2. Главная область (отражающая открытый функционал);

3. Панель ссылок; [\*](#_Требование_к_запуску)

4. Строка поиска; [\*](#_Требование_к_запуску)

5. Кнопки быстрого доступа к сообщениям и уведомлениям;

6. Администратор системы должен иметь возможность управлять мерчантами а именно: [\*](#_Требование_к_запуску)

- Добавление мерчанта

- Удаление мерчанта

- Отключение мерчанта

- Включение мерчанта,

- Применять бал, который дает мерчант при использовании пользователем

SMS уведомления

Функционал должен для получения OTP. Получив OTP, должен быть возможность использовать его, для подключения SMS- информирование. Ещё должен быть подключение и отключение SMS информирования с использованием уникального номера транзакции.

Платёжный инструмент должен выполнять следующий список функций

* **Проверка аккаунта.** Проверяет существует ли пользователь в системе. Если номер телефона существует и зарегистрирован в хранилище и возвращает информацию об активации, иначе возвращает информацию о не существовании.
* **Регистрация пользователей.** Регистрация нового пользователя в системе
* **Авторизация пользователей.** С помощью метода проверяется авторизация пользователя и устройство, которое выполняет эту операцию. Если авторизация пользователя прошла успешно, но устройство не зарегистрировано в хранилище отправляется OTP к номеру пользователя.
* **Активация аккаунта.** Активация устройства. В данном методе с помощью выше приведенной OTP зарегистрируется устройство в хранилище.
* **Проверка пароля пользователя.** Проверяется правильность пароля регистрированного пользователя в хранилище.
* **Восстановление пароля.** Проверяется логин пользователя на существование и совпадение нового пароля со старым. После проверки данных отправляется OTP к номеру пользователя для восстановления пароля.
* **Удаление аккаунта.** Удаляются настройки, услуги, платежные карты и аккаунт пользователя, кроме транзакции по карты.
* **Обновление данных пользователя.** Изменение информации пользователя в хранилище
* **Изменение пароля.** Предназначен для изменения текущего пароля с новым.
* **Удаление карт при восстановлении пароля аккаунта.** Метод предназначен на согласие пользователя для удаления всех карт пользователя при восстановлении пароля.
* **Статус согласия пользователя на удаление карт при восстановлении пароля аккаунта**. Метод предназначен для просмотра статуса, согласен ли пользователь удалить всех карты при восстановлении пароля.
* **Добавление виджета.** Добавить и сохранить местоположения виджетов. С помощью метода сохраняется последовательность виджетов в хранилище.
* **Получение виджетов.** Метод предназначен для получения расположения виджетов.
* **Все сервисы.** Метод предназначен для получения всех услуги системы из хранилища.
* **Все включенные сервисы пользователя.** Метод предназначен для получения всех подключенных услуг пользователя.
* **Вкл/отк. сервисов.** Метод предназначен для включения/отключения системных услуг пользователя.
* **Все устройства**. Метод предназначен для получения всех устройств пользователя, которые были зарегистрированы в хранилище.
* **Все доверенные устройства.**  Метод предназначен для получения всех доверенных устройств пользователя, которые были зарегистрированы в хранилище.
* **Удаление устройства**. С помощью метода удаляется выбранное устройство пользователя
* **Служба поддержки.** Пользователь отправляет свои отзывы (комментарии) о системе.
* **Получение все транзакции по картам пользователя**. Просмотр истории транзакции по картам в приложении.
* **Создание карты**. Метод предназначен для создания и регистрации платежной карты. Если добавляемая карта не подключена к услуге СМС-информирование, тогда данная карта не создается в системе.
* **Активация карты**. Метод выполняет активацию по полученного OTP пароля для добавления карты пользователя.
* **Информация о карте.** Метод предназначен для получения информацию о созданных картах в системе с балансом карты.
* **Статистика карты**. Метод предназначен для получения общих приходов и расходов по регистрированным картам пользователя с начало месяца по текущий день.
* **Обновление параметров карты**. Метод предназначен для изменения настроек карты пользователя
* **Удаление карты**. Метод удаляет выбранные карты пользователя из системы.
* **Выбор основной карты**. С помощью этого метода прикрепляется статус «основная карта» к одной из зарегистрированных карт пользователя.
* **Данные по приходам и расходам по карте**. Получение данных по приходам и расходам по карте пользователя с начала месяца, которая не была реализована вне системы, и записывает в базу данных.
* **Получение номер карты пользователя**. Получение маскированного номера выбранной карты пользователя. Номер карты передается в следующий виде: 8600ХХ\*\*\*\*ХХХХХХ.
* **Показать баланс карты.** Показать/скрыть баланс карты.
* **Последние транзакции по карте.** Метод предназначен для получения последние 4 транзакции по всем картам пользователя.
* **Отчет по приходам.** С помощью метода получает все приходные транзакции по выбранной карте пользователя (последние 80 шт).
* **Отчет по расходам.** Метод получает все расходные транзакции по выбранной карте пользователя (последние 80 шт).
* **Получение информацию о карте.** Результат запроса отображается для настройки карты в системе
* **Создание карты по номеру.** Добавляет платежную карту пользователя по номеру абонента и по параметрам карты. По параметрам запроса проверяется пользователь, номер карты и ее срок, если карта чужая, отправляется OTP в ответе запроса, иначе добавляется карта в систему (в базу данных). Если в параметрах запроса есть OTP параметр тоже, то тогда будет активироваться после проверки параметра запроса и активируется карта в базе данных.
* **Отчеты по транзакциям, осуществленные в системе.** Получает все транзакции пользователя в определенном количестве, которые были реализованы в системе.
* **Сохранить платёж.** Предназначен для сохранения транзакции для последующих действий
* **Изменить сохраненного платежа.** С помощью метода изменяется название сохранённой транзакции
* **Удалить сохраненный платеж.** Удаляет сохранённые транзакции и переводы, раннее сохранённых транзакций.
* **Получить все сохраненные платежи .** Метод предназначен для получения всех сохраненных транзакций из базы. Если транзакций не существует, то отправляет в ответе пустой лист транзакций.
* **Виджет «Сохранённые платежи».** Метод получает список сохранённых платежей, в ответе содержатся данные, которые позволяют повторять сохранённые платежи, с возможностью редактирования полей.
* **Получить все сохраненные платежи по P2P.** Метод получает все транзакции по переводам денежных средств с карты на карту из базы. Если не существует транзакций - отправляет в ответе пустой лист транзакций.
* **Отправить запрос платить другу.** Метод получает запрос об оплате и сравнивается с OTP. Если OTP правильно, тогда находит карту пользователя из базы и создается объект запроса, все его запросы платить другу и удаляются. После этого добавляется запрос в базу и сравнивается смс-инфо номер карты с телефоном номера пользователя. Если не совпадает, то проверяет количество OTP пользователя, если проверка прошла успешно, отправляет OTP. Если телефонный номер совпадает, отправляет ответ на запрос.
* **Получить запрос платить другу.** Метод проверяет на существование пользователя и после этого проверяются параметры запроса. После успешной проверки создается объект транзакции для получения средств «платит друг». Метод отправляет параметры в Gate для снятия денег и отправляет ответ на запрос.
* **Отправить и добавить запрос оплаты.** После выбора поставщика услуг, заполняются поля, например провайдер, логин и сумма. После нажатия кнопки «платит друг» на данный URL уходит запрос, в ответ возвращается ID транзакции, которая закрепляется в другом GET методе и присваивается к окончанию URL
* **Получить запрос оплаты.** Передача ссылки с полученным значением transactionId, в котором хранятся данные, переданного платежа и набор правил. Response передаёт информацию на 3х языках. Aditionalinfo – вложенный объект содержит информацию обо всех операторах, при неверном вводе кода оператора выводится уведомление допущенной ошибки.
* **Получить список всех запрос оплаты.** Метод предоставляет возможность получения всех запросов об оплате.
* **Проверка карта держателя по вводу PAN номера карты.** Пользователь указывает номер карты, на которую собирается осуществить перевод, после ввода 16-значеного PAN номера карты, следует запрос и ответ.
* **Оформление перевода.** Осуществление перевода денежных средств с карту на карту
* **Перевод средств.** Параметр «uniques» используется при транзакциях. На момент создания данных транзакции, выдаёт значение (время unix), далее по значению.
* **Информация о торговой точке.** Метод выводит все данные по каждому поставщику услуг, логотипы, название, поля, правила, если например это мобильные операторы. Каждый поставщик имеет свой параметр «merchantId» по которому производится транзакция.
* **Проверка правил полей каждого поставщика.** Проверяются данные о торговой точке
* **Оплата услуг поставщика.** Оплата услуг поставщика. (переданный платёж)

# **Требование в команде участника**

* Возможная работа команды на месте заказчика. (для быстрой коммуникации с внутренними разработчиками)
* Backend приложения должен быть написан на Java или Python с использованием микросервисных архитектур
* Frontend (web) должен быть написан на reactjs или angular
* Android приложение должно быть написано на Kotlin
* IOS приложение должно быть написано на Swift
* Deployment всех приложений должен осуществляться на стороне заказчика, на оборудовании заказчика.

# **Оплата труда**

Учитывая динамическое развитие проекта, и внесение постоянных изменений или улучшений, проект предусматривает внесение изменений в ходе разработки приложений. Ввиду всего этого предусматривается следующий способ оплаты: T&M оплата разработки за определенную часть работы.

# **Сроки выполнения работ**

Срок оказания Услуг по разработке мобильных приложений – до 31 мая 2022.  
  
Процесс внедрения мобильного приложения должен включать в себя следующие пункты:

* Утверждение дизайна мобильного приложения (UX & UI); [\*](#_Требование_к_запуску)
* Установка необходимого оборудования и ПО у Заказчика; [\*](#_Требование_к_запуску)
* Интеграция с системами Заказчика; [\*](#_Требование_к_запуску)
* Кастомизация мобильного приложения; [\*](#_Требование_к_запуску)
* Публикация мобильного приложения в соответствующих магазинах распространения приложений и на веб-сайте Заказчика. [\*](#_Требование_к_запуску)

# **Результаты проекта**

Результатом проекта будет являться:

* Решение, успешно развернутое в промышленной среде Заказчика;
* Документация для администраторов решения.
* Доступность решение 99.95% при нагрузке.
* Запуск 31 мая 2022

Критерием успеха будет являться коммерческий запуск комплекса и размещение мобильного приложения в соответствующих местах продаж (Google Play и AppStore). Приложение должно быть размещено в Google Play и AppStore усилиями разработчика через аккаунты Оператора.

# **Требование по техническому обучению персонала**

Необходимо провести обучение персонала компании заказчика в количестве 5 человек.

4 человека для разработки и дальнейшей поддержки продукта. Ознакомление разработчиков с технической частью продукта, использованными в ходе разработки библиотеками и их особенностями, структурой и организации кода, обучению механизму деплоя в продакшн(если имеется свой).

1 человек для администрирования и решения оперативных вопросов. Администрирование продукта, решение и выявление проблем пользуясь описанным в п.12 инструментами.

# **Запуск мобильного, платежного приложения**

Запуск мобильного, платежного приложения должна производиться в нескольких этапах;

* Предварительный запуск системы (тестирование) перед вводом на эксплуатацию – в данном этапе после запуска проводится полное тестирование системы и его функционала а так же производится стресс тест на отказоустойчивость системы
* Нагрузочное тестирование
* Доработка и исправление багов системы по завершению тестирования – в данном этапе исправляется все обнаруженные на раннем этапе неисправности и выполняется доработка\оптимизация системы.
* Финальное тестирование системы – После исправление всех неисправностей и оптимизации системы производится финальное (повторное) тестирование системы и его функционала.
* Запуск платёжной системы – Если в предыдущем этапе проблем не обнаружено считается что система готова на полноценный запуск в эксплуатацию

# **Требование к запуску на MVP (обозначено \* выше по тексту)**

* Запуск проекта 31.05.2022