**Проектирование интерактивной системы онлайн-оплаты коммунальных услуг**

### Основная цель проекта

Основной целью проекта является создание удобной, безопасной и масштабируемой веб-платформы для онлайн-оплаты коммунальных услуг. Система должна минимизировать временные затраты пользователей, повысить прозрачность начислений.

### Макет интерфейса

### 

### Рис 1. Главная страница

1. **Главная страница**: Главная страница служит точкой входа и предназначена для быстрого доступа к основным функциям. Основные элементы:
   * **Смена языка**: Поле для выбора языка интерфейса.
   * **Кнопка входа/регистрации**: Расположена в верхнем правом углу для доступа к личному кабинету.
   * **Информационный блок**: Краткие инструкции или новости (например, об изменении тарифов).  
     Дизайн главной страницы минималистичен, с акцентом на поле поиска и яркими кнопками для категорий услуг.

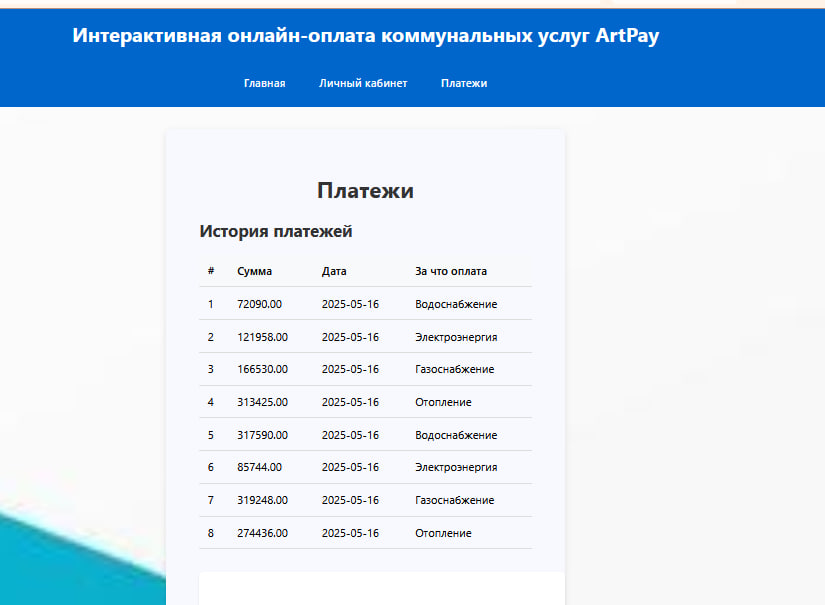


Рис 2. Просмотр платежей.

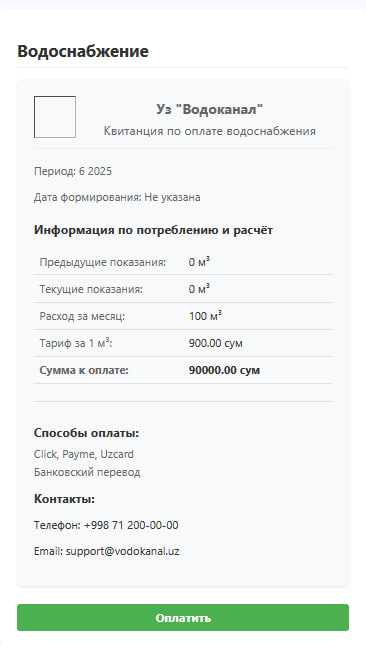


Рис 3. Просмотр квитанции

1. **Страница платежей**: Страница оплаты отображает информацию о выбранной услуге и позволяет завершить транзакцию. Ключевые элементы:
   * **Детали начисления**: Сумма, тип услуги, период, номер лицевого счета.
   * **Выбор способа оплаты**: Список доступных платежных систем (Click, Payme, Uzum Bank, Visa, Mastercard) с иконками.
   * **Форма оплаты**: Поля для ввода данных карты или перенаправление на платежный шлюз.
   * **Кнопка подтверждения**: Крупная кнопка "Оплатить" с указанием суммы.  
     Страница использует пошаговую структуру, чтобы пользователь мог легко проверить данные перед оплатой. Всплывающие подсказки помогают избежать ошибок при вводе.

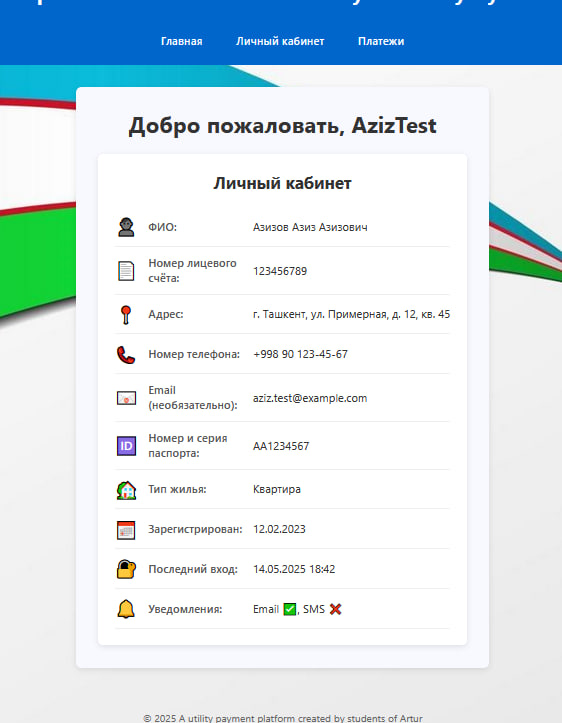


Рис 4. Просмотр платежей.

1. **Личный кабинет**:  
   Личный кабинет предоставляет пользователю доступ к персонализированным функциям. Основные разделы:
   * **Тип жилья**: позволяет корректнее производить вычисления при подсчете коммунальных услуг.
   * **Дата регистрации**: Позволяет подсчитать с какого числа пользователь зарегистрировался в системе.
   * **Настройки**: Управление уведомлениями (email, SMS, push), смена пароля, выбор языка интерфейса.
   * **Профиль**: Отображение личных данных (имя, email, номер телефона) с возможностью редактирования.  
     Дизайн личного кабинета организован в виде вкладок для удобной навигации, с акцентом на историю платежей как наиболее востребованную функцию.

**Разработка основных модулей системы**

### Модуль авторизации и регистрации пользователей

### 

### Рис 5. Позволяет зарегистрировать пользователя.

### При введение всех нужных данных пользователь будет добавлен в систему. Однако в случае если пользователь хочет воспользоваться другим способом он может использовать кнопку ниже.

### 

### Рис 6. Альтернативная регистрация.

### В данном случае пользователь может войти в систему посредством входа через другие государственные платформы.

### Теперь рассмотрим процедуру входа.

### 

### Рис 7. Процедура входа.

### В случае если перейти на страницу входа перед пользователем появится окно на котором будет предложено ввести логин и пароль после чего войти на платформу. В случае если пользователь еще не регистрировался он может нажать на кнопку регистрации и его перекинет на страницу регистрации.

### 

### Рис 8. Процедура входа сотрудника

### Если пользователь нажмет на кнопку войти как сотрудник ему будет предложено ввести данные ключа сотрудника для входа. Если данная кнопка нажата случайно то пользователь может нажать на кнопку войти как пользователь.

### Модуль обработки платежей и интеграции с платежными системами

### 

### Рис 9. Пример квитанции.

1. **Функциональность модуля**: Модуль отвечает за обработку платежей и взаимодействие с платежными шлюзами. Основные функции:
   * **Способ оплаты**: Пользователь вводит данные карты.
   * **Обработка транзакций**: Проверка начислений, отправка запроса на платеж, получение подтверждения и обновление статуса в базе данных.
   * **Генерация квитанций**: Формирование PDF-квитанции с деталями платежа (сумма, дата, услуга, лицевой счет).

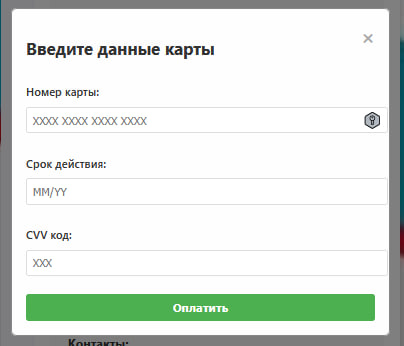


Рис 10. Ввод данных для оплаты.

### Модуль управления пользователями

### 

### Рис 11. Вид на панель администратора.

1. **Функциональность модуля**. Основные функции:
   * **Создание тестовых пользователей и квитанций**: Отправка сообщений о новых начислениях, статусе платежей и напоминаниях об оплате через email, SMS или push-уведомления.
   * **Просмотр пользователей**: Отображение и фильтрация пользователей.
   * **Отчеты для администраторов**: Аналитика по количеству платежей, популярным услугам и активности пользователей.
   * **Изменение данных**: Позволяет изменять данные пользователей.

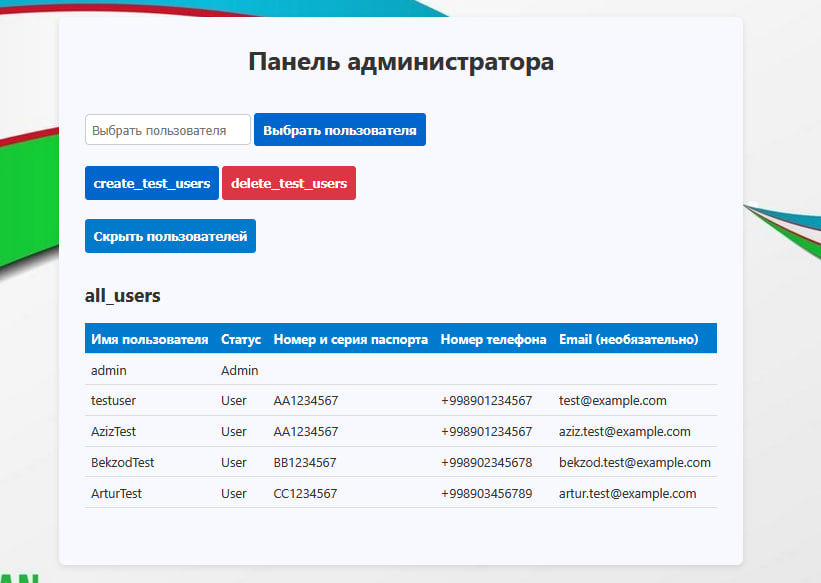


Рис 12. Просмотр пользователей.

Позволяет просматривать в виде таблицы данные пользователей системы.

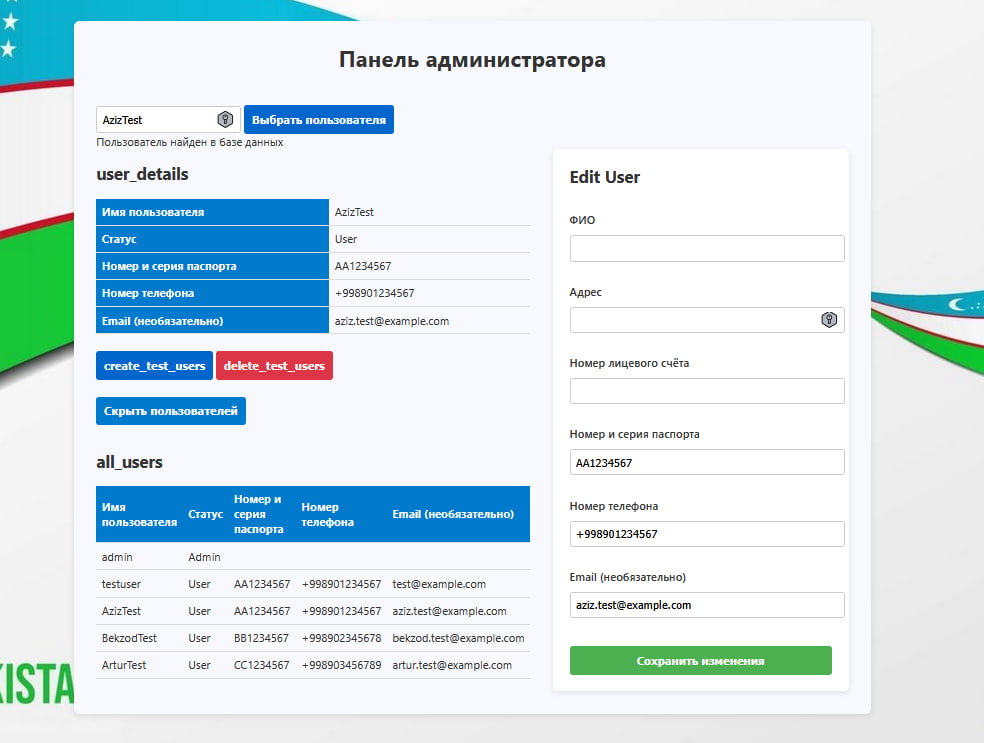


Рис 13. Изменение данных пользователя.

Позволяет изменять определенные данные в случае если они устарели.

3.2. **Тестирование системы**

Тестирование системы онлайн-оплаты коммунальных услуг является критическим этапом, обеспечивающим ее надежность, безопасность и удобство использования. Тестирование проводилось в трех направлениях: функциональное, производительности и безопасности, а также пользовательское, чтобы гарантировать соответствие требованиям и ожиданиям пользователей в Узбекистане.

### Функциональное тестирование

1. **Цель**:  
   Проверка корректной работы всех функций системы, включая регистрацию, оплату, уведомления и отчеты.
2. **Методы**: **Ручное тестирование**: Проверялись сценарии, такие как регистрация нового пользователя, оплата услуги по QR-коду, восстановление пароля и экспорт истории платежей.
3. **Сценарии тестирования**:
   * Регистрация с корректными и некорректными данными (например, неверный формат email).
   * Оплата через Visa, включая случаи сбоев (недостаток средств, тайм-аут).
4. **Результаты:** Все функции работали корректно при нормальных условиях. Выявлены мелкие ошибки, такие как некорректное отображение ошибок валидации на узбекском языке, которые были исправлены.

### Тестирование производительности и безопасности

1. **Производительность**:
   * **Цель**: Проверка способности системы обрабатывать высокую нагрузку и обеспечивать быстрый отклик.
   * **Методы**: Использовался инструмент JMeter для симуляции 10 одновременных пользователей. Тестировались сценарии: поиск начислений, оплата и генерация отчетов.
   * **Результаты**: Среднее время отклика составило 0,8 секунды при пиковой нагрузке. При нагрузке свыше 8 пользователей наблюдалось замедление до 2 секунд, что потребовало оптимизации кэширования Redis. После доработки система стабильно обрабатывала 10 запросов с откликом менее 1 секунды.

3.5. **Инструкция по базе данных.**

### Создание таблиц

Таблицы создаются с помощью SQL-запроса CREATE TABLE IF NOT EXISTS в функции init\_db(). Всего создается пять таблиц: users, bills, payments, utility\_data и waste\_receipts. Ниже приведен разбор каждой таблицы.

#### Таблица users

Столбцы:

* id: Уникальный идентификатор пользователя (автоинкремент, первичный ключ).
* username: Имя пользователя (уникальное, не NULL).
* password: Хэш пароля (не NULL).
* passport: Номер и серия паспорта (опционально).
* phone: Номер телефона (опционально).
* email: Электронная почта (опционально).
* is\_staff: Флаг, указывающий, является ли пользователь сотрудником (0 или 1, по умолчанию 0).
* address: Адрес пользователя (опционально).
* full\_name: Полное имя пользователя (опционально).
* account\_number: Номер лицевого счета (опционально).

Таблица bills

Столбцы:

* id: Уникальный идентификатор счета (автоинкремент, первичный ключ).
* user\_id: Идентификатор пользователя, которому принадлежит счет (не NULL, внешний ключ на users.id).
* amount: Сумма счета (не NULL, вещественное число).
* due\_date: Срок оплаты (не NULL, текстовая строка в формате даты).
* paid: Флаг, указывающий, оплачен ли счет (0 или 1, по умолчанию 0).
* utility\_type: Тип коммунальной услуги (не NULL, например, "water", "electricity").
* month: Месяц счета (опционально, целое число).
* year: Год счета (опционально, целое число).

Зависимости:

* Внешний ключ user\_id ссылается на id таблицы users.

Таблица payments

Столбцы:

* id: Уникальный идентификатор платежа (автоинкремент, первичный ключ).
* user\_id: Идентификатор пользователя, совершившего платеж (не NULL, внешний ключ на users.id).
* bill\_id: Идентификатор счета, который был оплачен (не NULL, внешний ключ на bills.id).
* amount: Сумма платежа (не NULL, вещественное число).
* payment\_date: Дата платежа (не NULL, текстовая строка в формате даты).

Зависимости:

* Внешний ключ user\_id ссылается на id таблицы users.
* Внешний ключ bill\_id ссылается на id таблицы bills.

Таблица utility\_data

Столбцы:

* id: Уникальный идентификатор записи (автоинкремент, первичный ключ).
* user\_id: Идентификатор пользователя (не NULL, внешний ключ на users.id).
* utility\_type: Тип коммунальной услуги (не NULL, например, "water", "electricity").
* month: Месяц данных (не NULL, целое число).
* year: Год данных (не NULL, целое число).
* previous\_reading: Предыдущие показания счетчика (опционально, вещественное число).
* current\_reading: Текущие показания счетчика (опционально, вещественное число).
* consumption: Потребление (опционально, вещественное число).
* tariff: Тариф за единицу потребления (опционально, вещественное число).
* amount: Итоговая сумма (опционально, вещественное число).
* paid: Флаг, указывающий, оплачена ли услуга (0 или 1, по умолчанию 0).
* generation\_date: Дата генерации данных (опционально, текстовая строка).
* adjustments: Корректировки (опционально, текст).

Зависимости:

* Внешний ключ user\_id ссылается на id таблицы users.

Таблица waste\_receipts

Столбцы:

* id: Уникальный идентификатор квитанции (автоинкремент, первичный ключ).
* user\_id: Идентификатор пользователя (не NULL, внешний ключ на users.id).
* receipt\_number: Номер квитанции (не NULL, текст).
* amount: Сумма квитанции (не NULL, вещественное число).
* payment\_date: Дата платежа (не NULL, текстовая строка в формате даты).
* paid: Флаг, указывающий, оплачена ли квитанция (0 или 1, по умолчанию 0).

Зависимости:

* Внешний ключ user\_id ссылается на id таблицы users.

### Зависимости между таблицами

Таблицы связаны через внешние ключи, что обеспечивает целостность данных. Вот как они зависят друг от друга:

* **Таблица users**:
  + Является родительской таблицей.
  + Связана с таблицами bills, payments, utility\_data и waste\_receipts через столбец user\_id, который ссылается на users.id.
  + Удаление записи в users потребует удаления или обработки связанных записей в дочерних таблицах (в зависимости от настроек каскадного удаления, которые в коде не указаны).
* **Таблица bills**:
  + Зависит от users через user\_id (внешний ключ на users.id).
  + Связана с payments через bill\_id, который ссылается на bills.id.
  + Хранит информацию о счетах за коммунальные услуги.
* **Таблица payments**:
  + Зависит от users через user\_id (внешний ключ на users.id).
  + Зависит от bills через bill\_id (внешний ключ на bills.id).
  + Хранит историю платежей по счетам.
* **Таблица utility\_data**:
  + Зависит от users через user\_id (внешний ключ на users.id).
  + Не имеет прямой связи с bills или payments через внешние ключи, но связана с ними логически через user\_id, utility\_type, month и year (используется в запросах для объединения данных).
* **Таблица waste\_receipts**:
  + Зависит от users через user\_id (внешний ключ на users.id).
  + Хранит данные о квитанциях за вывоз отходов, не связана напрямую с другими таблицами, кроме users.