Санкт-Петербургский государственный университет

Кафедра системного программирования

Группа 21.Б07-мм

Добавление аутентификации через новые Identity-провайдеры в Miminet

Целовальникова Екатерина Андреевна

Отчёт по учебной практике
в форме «Решение»

Научный руководитель:
старший преподаватель кафедры СП И. В. Зеленчук

Санкт-Петербург
2024

Оглавление

Введение

Miminet [13] – это веб-эмулятор компьютерной сети, разрабатываемый для образовательных целей. Этот инструмент обеспечивает пользователей виртуальной средой, которая позволяет моделировать, анализировать и тестировать разнообразные конфигурации компьютерных сетей, и при этом не требует использования реального оборудования.

Miminet представляет собой графическое отображение IPMininet [10], широко используемого инструмента для создания виртуальных сетевых топологий в операционной системе Linux. IPMininet создает виртуальные сети с ограниченными ресурсами, предназначенные для обучения и тестирования различных аспектов сетевых технологий, обеспечивая быстроту и легкость в использовании.

В текущей версии Miminet реализованы механизмы аутентификации через Google и ВКонтакте. Однако с учетом положений Федерального закона «Об информации, информационных технологиях и о защите информации [20]», российские веб-ресурсы обязаны перейти к аутентификации пользователей через системы, зарегистрированные в России, к 1 января 2025 года.

В соответствии с новыми требованиями, предъявляемыми к аутентификации пользователей на российских платформах, доступ к сервисам будет возможен лишь при использовании номеров телефонов, портала «Госуслуги[1]» (ЕСИА), Единой биометрической системы[2] и прочих информационных систем, находящихся под российским контролем.

С учетом принятых нормативных актов, команда Miminet приняла решение о внедрении аутентификации через две новые выбранные платформы в веб-приложении. Этот шаг обусловлен стремлением Miminet предоставить пользователям более широкий выбор вариантов для аутентификации в приложении, соответствуя требованиям российского законодательства.

[1] https://www.gosuslugi.ru

[2] https://ebs.ru

1. Постановка задачи

Целью текущей учебной практики является интеграция механизмов аутентификации через две выбранные платформы в веб-приложение Miminet.

Для достижения цели были определены следующие задачи:

1. Провести обзор Identity-провайдеров
2. Изучить документацию выбранных Identity-провайдеров
3. Реализовать аутентификацию через выбранные Identity-провайдеры:
    3.1 Реализовать инициацию процесса аутентификации
    3.2 Реализовать обработку ответа аутентификации
4. Добавить соотвествующие кнопки на страницу входа Miminet
5. Провести тестирование добавленных функций

2. Обзор

2.1. Обзор Identity-провайдеров

В данном обзоре проведено сравнение Identity-провайдеров по следующим критериям:

- **Аудитория или количество пользователей:** Средняя аудитория или общее количество зарегистрированных пользователей, в зависимости от доступных данных.
- **Наличие документации:** Доступность документации для разработчиков.
- **Интеграция с внешними сервисами:** Возможности интеграции с API провайдера.
- **Ограничения на использование:** Требования и ограничения для различных типов организаций.

2.1.1. Госуслуги (ЕСИА)

Основные особенности:

- **Количество пользователей:** По итогам 2023 года количество пользователей портала "Госуслуги" достигло 109 миллионов человек [16].
- **Наличие документации:** ЕСИА предоставляет документацию [14], которая включает методические рекомендации по интеграции с REST API цифрового профиля.
- **Интеграция с внешними сервисами:** ЕСИА предоставляет REST API, который реализует механизм аутентификации пользователей на основе стандартов OAuth 2.0 и OpenID Connect 1.0 [6].

Ограничения на использование:

- **Ориентированность на государственные органы:** Процесс регистрации в ЕСИА ориентирован в первую очередь на государственные органы и организации [17].

- **Ограничения для частных организаций:** Процесс регистрации в ЕСИА предполагает, что организация обязательно должна быть юридическим лицом [19].

Ограничения для частных организаций делает регистрацию в системе невозможной для негосударственных организаций, в том числе для Miminet, которое не является юридическим лицом.

2.1.2. Одноклассники (OK)

Основные особенности:

- **Аудитория:** В четвертом квартале 2023 года среднемесячная аудитория Одноклассников в России составила 35 миллионов [15].

- **Наличие документации:** Компания VK предоставляет документацию [4] для разработчиков, описывающую использование API Одноклассников для аутентификации.

- **Интеграция с внешними сервисами:** При помощи протокола OAuth 2.0 можно получить доступ к API Одноклассников с сервера внешнего сайта или приложения.

2.1.3. Telegram

Основные особенности:

- **Аудитория:** В декабре 2023 года месячная аудитория Telegram в России составила 82.7 миллионов, а средняя дневная аудитория – 57 миллионов [1].

- **Наличие документации:** Telegram предоставляет документацию для разработчиков, описывающую использование Telegram Login Widget [9] для аутентификации.

- **Интеграция с внешними сервисами:** Telegram Login Widget использует механизмы, предоставляемые Telegram Bot API [8], для аутентификации пользователей на внешних платформах через использование их Telegram аккаунтов.

2.1.4. Mail.ru

Основные особенности:

- **Аудитория:** В декабре 2023 года месячная аудитория Mail.ru в России составила 76 миллионов, а средняя дневная аудитория – 24.6 миллионов [1].
- **Наличие документации:** Компания VK предоставляет документацию [3] для разработчиков, описывающую использование API Mail для аутентификации.
- **Интеграция с внешними сервисами:** При помощи протокола OAuth 2.0 можно получить доступ к API Mail с сервера внешнего сайта или приложения.

2.1.5. Яндекс

Основные особенности:

- **Аудитория:** В декабре 2023 года месячная аудитория компании Яндекс в России составила 99.8 миллионов, а средняя дневная аудитория – 68.5 миллионов [1].
- **Наличие документации:** Яндекс предоставляет документацию [18] для разработчиков, описывающую использование API Яндекс ID для аутентификации.
- **Интеграция с внешними сервисами:** API Яндекс ID позволяет настроить аутентификацию пользователя с использованием протокола OAuth 2.0 на сайте или в мобильном приложении.

В контексте реализации аутентификации в веб-приложении Miminet было принято решение использовать Яндекс и Telegram. Это решение было обусловлено значительным размером аудитории обоих сервисов, который является самым большим среди всех сравниваемых Identity-провайдеров.

2.2. Используемые инструменты

2.2.1. Flask

Miminet разработан на основе Flask [7] – гибкого веб-фреймворка для Python. Он обеспечивает обработку HTTP-запросов, определение маршрутов, создание шаблонов и управление сессиями.

2.2.2. Протокол OAuth 2.0

Протокол OAuth 2.0 [2] – стандарт авторизации, используемый для передачи авторизационных данных между клиентом и сервером. Применение API Яндекс ID дает возможность настройки процесса аутентификации пользователя с использованием данного протокола в рамках веб-приложения.

2.2.3. Telegram Login Widget

Telegram Login Widget – механизм, позволяющий веб-сервисам реализовать систему аутентификации пользователей через Telegram. Взаимодействие с Telegram Bot API требуется для получения согласия пользователя на предоставление данных и их последующей проверки на подлинность.

2.3. Интеграция с Яндекс через протокол OAuth 2.0

2.3.1. Основы протокола OAuth 2.0

OAuth 2.0 [2] представляет собой протокол авторизации, обеспечивающий сторонним приложениям доступ к защищенным данным поль-

зователя на веб-ресурсах без необходимости раскрытия учетных данных.

Протокол OAuth 2.0 определяет четыре роли:

- **Владелец ресурса (Resource Owner)** – пользователь, у которого есть защищенные данные на веб-ресурсе и который может дать доступ к этим данным стороннему приложению.
- **Стороннее приложение (Client)** – приложение, запрашивающее доступ к защищенным данным пользователя через сервер авторизации.
- **Сервер авторизации (Authorization Server)** – сервер, отвечающий за аутентификацию пользователей и выдачу токенов доступа сторонним приложениям.
- **Сервер ресурсов (Resource Server)** – сервер, который хранит защищенные данные пользователя и отвечает на запросы на доступ к этим данным от сторонних приложений.

2.3.2. Параметры приложения

Для получения OAuth-токенов необходимы следующие параметры [18], предоставленные при регистрации приложения на сервере авторизации Яндекс:

- **Client ID** — уникальный идентификатор приложения, выданный Яндексом во время регистрации. Используется для идентификации приложения в процессе запроса авторизации.
- **Client Secret** — секретный ключ приложения, предоставленный Яндексом вместе с Client ID. Используется для обмена авторизационного кода на токен доступа.
- **Redirect URI** — предварительно зарегистрированный URI, на который Яндекс перенаправит пользователя после успешной авторизации. По этому адресу будет отправлен запрос с авторизационным кодом, который затем обменивается на токен доступа.

2.3.3. Процедура авторизации приложения

Приложения запрашивают OAuth-токены в соответствии со следующей схемой:

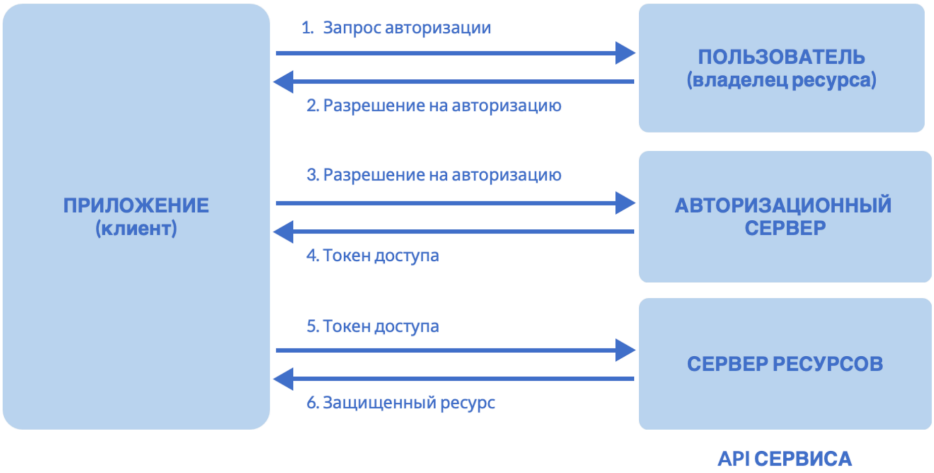

Рис. 1: Процедура авторизации через OAuth 2.0

1. Приложение перенаправляет пользователя на страницу Яндекса для аутентификации и запроса на предоставление разрешений для доступа к его данным.

2. После аутентификации и предоставления пользователем необходимых разрешений, Яндекс отправляет данные обратно в приложение, включая авторизационный код.

3. Приложение использует полученный авторизационный код для запроса токена доступа у сервера авторизации Яндекса, предоставляя свой Client ID и Client Secret.

4. Сервер авторизации Яндекса проверяет авторизационный код и, если все в порядке, выдает токен доступа приложению.

5. Приложение использует полученный токен доступа для запроса защищенных данных пользователя у сервера ресурсов Яндекса.

2.4. Интеграция с Telegram через Login Widget

2.4.1. Создание и настройка бота для виджета

Для активации Telegram Login Widget [9] необходимо создать Telegram бота, который инициирует процесс аутентификации пользователя и авторизует приложение для доступа к данным пользователя в Telegram. Процесс создания и настройки бота осуществляется через `@BotFather`.

2.4.2. Привязка домена к боту

Для активации виджета необходимо связать домен веб-сервиса с ботом, используя команду `/setdomain` в `@BotFather`. После того как пользователь успешно аутентифицируется в Telegram с использованием виджета, бот получает информацию с полем `connected_website`, которое содержит доменное имя сайта, где была инициирована аутентификация.

2.4.3. Настройка виджета

Настройка виджета включает определение имени бота, стиля кнопки и типа авторизации, используемых на веб-сервисе, и осуществляется на специальной странице.

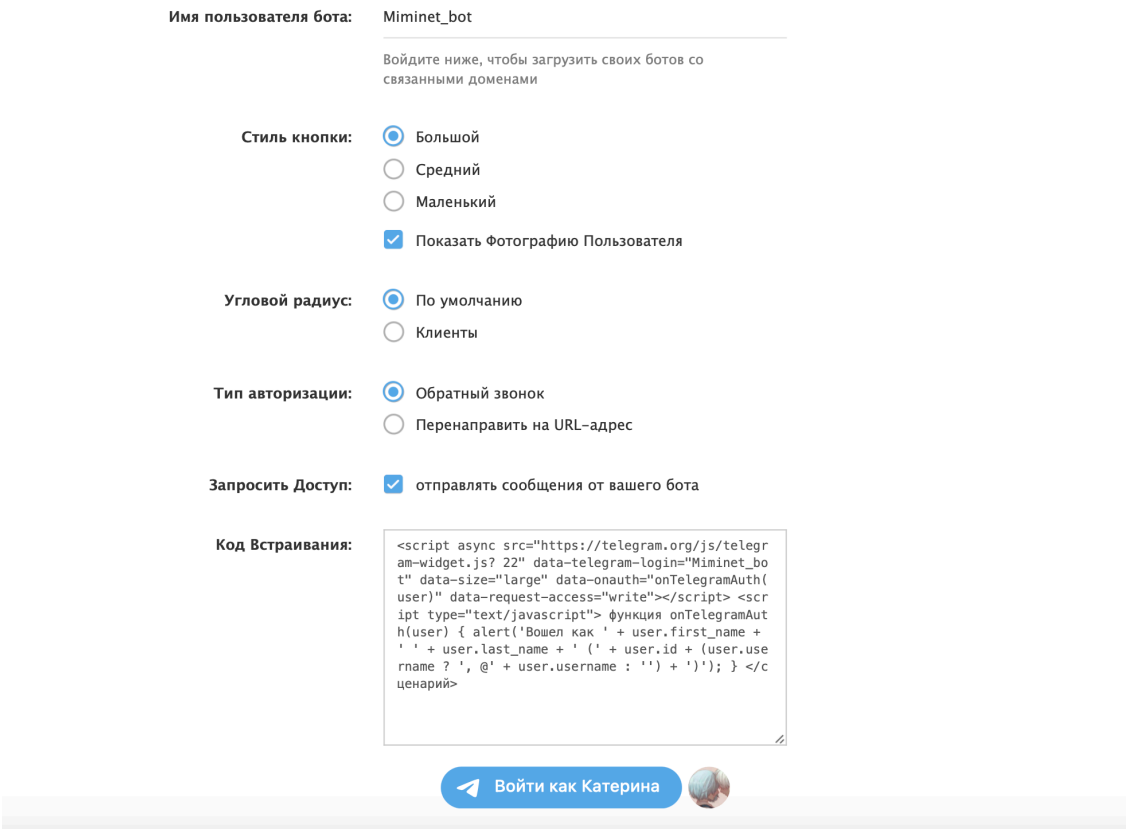

Рис. 2: Настройка Telegram Login Widget

В дополнение к основным данным, виджет также передает расширенную информацию о пользователе:

- **ID пользователя** – уникальный идентификатор пользователя в Telegram.
- **Дата аутентификации** – время, когда пользователь прошел аутентификацию в Telegram, представленное в формате Unix time.
- **Хэш** – специально сформированная подпись, используемая для проверки данных, полученных от Telegram.

3. Ход работы

3.1. Реализация аутентификации через Яндекс в Miminet

3.1.1. Инициация процесса аутентификации

Начало процесса аутентификации через Яндекс в Miminet реализовано с использованием функции `yandex_login`. Данная функция инициирует процесс аутентификации пользователя, перенаправляя его на страницу Яндекса для входа в систему и предоставления согласия на доступ приложения Miminet к его данным на Яндексе.

Листинг 1: Функция `yandex_login()`

```
def yandex_login(yandex_json=yandex_json):
    yandex_session = OAuth2Session(
        yandex_json["web"]["client_id"],
        redirect_uri=yandex_json["web"]["redirect_uris"][0]
    )

    authorization_url, state = yandex_session.authorization_url(
        yandex_json["web"]["auth_uri"], access_type="offline",
        prompt="consent"
    )
    session["state"] = state

    return redirect(authorization_url)
```

В процессе работы функции `yandex_login` выполняются следующие действия:

- **Инициирование сеанса OAuth 2.0:** Инициируется сеанс с применением библиотеки `OAuth2Session`, который будет использоваться для взаимодействия с Яндексом.
- **Извлечение параметров для аутентификации:** Из конфигу-

рационного JSON-файла Яндекса извлекаются необходимые параметры, такие как идентификатор клиента и URL перенаправления, которые будут использоваться для получения кода авторизации и последующего получения токена доступа.

- **Создание URL авторизации:** Создается URL для перенаправления пользователя на страницу аутентификации Яндекса, включая параметры, необходимые для запроса кода авторизации для приложения Miminet.
- **Установка состояния сеанса:** Уникальное состояние сеанса сохраняется в сессии пользователя для обеспечения безопасности процесса аутентификации.
- **Перенаправление на страницу аутентификации Яндекса:** Пользователь перенаправляется на страницу аутентификации Яндекса для входа в систему и подтверждения доступа приложения Miminet к его учетной записи.

3.1.2. Обработка ответа аутентификации

После инициирования процесса аутентификации через Яндекс, следующим шагом является использование функции `yandex_callback`. Данная функция предназначена для обработки ответа аутентификации пользователя, полученного на странице Яндекса, и последующей авторизации его в системе Miminet.

Листинг 2: Функция `yandex_callback`, 1 часть

```
def yandex_callback(yandex_json=yandex_json):
    state = session.get("state")
    if not state:
        return redirect(url_for("login_index"))
    try:
        yandex_session = OAuth2Session(
            yandex_json["web"]["client_id"],
            redirect_uri=yandex_json["web"]["redirect_uris"][0],
```

```
        state=state,
    )
    yandex_session.fetch_token(
        yandex_json["web"]["token_uri"],
        authorization_response=request.url,
        client_secret=yandex_json["web"]["client_secret"],
    )
    user_info_response =
        yandex_session.get("https://login.yandex.ru/info")
    user_info_response.raise_for_status()
    id_info = user_info_response.json()
    ...
```

В первой части функции `yandex_callback` выполняются следующие действия:

- **Проверка состояния сеанса:** Проверяется наличие сохраненного состояния сеанса аутентификации.
  - В случае отсутствия состояния сеансаспроисходит перенаправление пользователя на страницу входа для повторной аутентификации.
- **Инициирование сеанса OAuth 2.0:** С использованием библиотеки `OAuth2Session` инициируется сеанс OAuth 2.0. Этот шаг включает передачу параметров, таких как идентификатор клиента, URL перенаправления и сохраненное состояние сеанса, и является ключевым в процессе авторизации приложения Miminet Яндексом.
- **Обмен кода авторизации на токен доступа:** С помощью сеанса OAuth 2.0 запрашивается у Яндекса токен доступа, предоставляя полученный ранее код авторизации и секрет клиента. Этот токен подтверждает авторизацию приложения Miminet для доступа к данным пользователя.

- **Получение информации о пользователе:** После успешного получения токена доступа делается запрос к API Яндекса ID для получения данных о профиле пользователя.

Листинг 3: Функция yandex_callback, 2 часть

```
def yandex_callback(yandex_json=yandex_json):
    ...
    user = User.query.filter_by(
        yandex_id=id_info.get("id")).first()
    if user is None:
        try:
            new_user = User(
                nick=id_info.get("login", ""),
                yandex_id=id_info.get("id"),
                email=id_info.get("default_email", ""),
            )
            db.session.add(new_user)
            db.session.commit()
        except SQLAlchemyError as e:
            db.session.rollback()
            return redirect(url_for("login_index"))
        user = User.query.filter_by(
            yandex_id=id_info.get("id")).first()
    login_user(user, remember=True)
    return redirect_next_url(fallback=url_for("home"))
```

Во второй части функции `yandex_callback` выполняются следующие действия:

- **Проверка наличия пользователя в базе данных:** Проверяется наличие пользователя в базе данных Miminet, используя уникальный идентификатор, полученный от Яндекса.

- **Регистрация нового пользователя:** Если пользователь не найден, создается новая учетная запись с данными, полученными от

Яндекса, и добавляется в базу данных Miminet.

  - В случае возникновения ошибки при регистрации происходит откат транзакции и перенаправление на страницу входа.

- **Авторизация пользователя в системе:** После подтверждения наличия пользователя в базе данных или создания новой учетной записи, осуществляется вход пользователя в систему Miminet и перенаправление его на домашнюю страницу.

3.2. Реализация аутентификации через Telegram в Miminet

3.2.1. Инициация процесса аутентификации

Начало процесса аутентификации через Telegram в Miminet реализовано с использованием Telegram Login Widget. Данный виджет инициирует процесс аутентификации пользователя и авторизирует Miminet для доступа к данным пользователя в Telegram.

Листинг 4: Telegram Login Widget

```
<script async
  src="https://telegram.org/js/telegram-widget.js?22"
  data-telegram-login="Miminet_bot"
  data-*
  data-onauth="onTelegramAuth(user)">
</script>
<script type="text/javascript">
  function onTelegramAuth(user) {
    var fullName = user.first_name;
    ...
    window.location.href = "{{url_for('tg_callback')}}?user="
      + encodeURIComponent(JSON.stringify(user));
  }
</script>
```

В представленном скрипте выполняются следующие действия:

- **Загрузка Telegram Login Widget:** Инициируется загрузка виджета с официального сайта Telegram, который предоставляет интерфейс для аутентификации пользователя и его последующей авторизации на передачу данных в Miminet.
- **Настройка виджета:** Виджет настраивается с использованием параметров, таких как имя бота, размер и радиус кнопки, а также обработчик события `data-onauth`, который активируется после того, как пользователь аутентифицировался и авторизовал передачу своих данных из Telegram.
- **Вызов функции обработки данных пользователя:** Функция `onTelegramAuth` вызывается автоматически после того, как пользователь аутентифицировался в Telegram и дал согласие на передачу своих данных. Функция обрабатывает эти данные и готовит их к дальнейшему использованию.
- **Перенаправление на функцию обратного вызова:** Функция `onTelegramAuth` перенаправляет пользователя на URL обработчика входа в Miminet (`tg_callback`), передавая данные пользователя как параметры запроса.

3.2.2. Проверка данных авторизации Miminet в Telegram

Для обеспечения безопасности передачи данных авторизации Miminet в Telegram используется функция `check_tg_authorization`, которая обеспечивает проверку целостности и актуальности полученной информации.

Листинг 5: Функция `check_tg_authorization`

```
def check_tg_authorization(auth_data, tg_json=tg_json):
    BOT_TOKEN = tg_json["token"]
    check_hash = auth_data["hash"]
    del auth_data["hash"]
```

```
data_check_arr = [f"{key}={value}" for key,
    value in auth_data.items()]
data_check_arr.sort()
data_check_string = "\n".join(data_check_arr)
secret_key = hashlib.sha256(BOT_TOKEN.encode()).digest()
hash_result = hmac.new(
    secret_key, data_check_string.encode(), hashlib.sha256
).hexdigest()
if hash_result != check_hash:
    raise Exception("...")
if (time.time() - int(auth_data["auth_date"])) > 86400:
    raise Exception("...")
return auth_data
```

Данная функция выполняет следующие шаги:

- **Извлечение данных и удаление хэша:** Полученные данные авторизации и хэш проверяются на наличие целостности и актуальности. Хэш удаляется из данных для дальнейшей обработки.
- **Формирование строки для проверки**: Данные авторизации сортируются и объединяются в строку для последующей проверки.
- **Вычисление хэша:** Вычисляется хэш полученных данных с использованием секретного ключа, основанного на токене бота Telegram. Для этого применяется алгоритм хэширования HMAC-SHA-256.
- **Проверка хэша:** Полученный хэш сравнивается с хэшем, полученным от Telegram, чтобы убедиться в подлинности данных.
  - Если хэши не совпадают, функция выбрасывает исключение, указывая на несоответствие данных.
- **Проверка актуальности данных:** Проверяется, что данные авторизации были получены в течение последних 24 часов.

- Если данные устарели, функция выбрасывает исключение, указывая на истечение срока действия данных.

- **Возврат данных:** Если данные прошли все проверки, функция возвращает информацию для дальнейшего использования в приложении Miminet.

3.2.3. Обработка ответа аутентификации

После инициирования процесса аутентификации через Telegram, следующим шагом является использование функции `tg_callback`. Данная функция предназначена для обработки ответа аутентификации пользователя в Telegram и последующей авторизации его в системе Miminet.

Листинг 6: Функция `tg_callback`

```
def tg_callback():
    user_json = request.args.get("user")
    if not user_json:
        return redirect(url_for("login_index"))
    user_data = json.loads(user_json)
    try:
        auth_data = check_tg_authorization(user_data)
    except Exception as e:
        return redirect(url_for("login_index"))
    user = User.query.filter(
        (User.tg_id == str(auth_data.get("id", "")))
        | (User.email == auth_data.get("username", ""))
    ).first()
    if user is None:
        try:
            new_user = User(
                nick=auth_data.get("first_name", ""),
                tg_id=str(auth_data.get("id", "")),
                email=auth_data.get("username", ""),
            )
```

```
            db.session.add(new_user)
            db.session.commit()
        except SQLAlchemyError as e:
            db.session.rollback()
            return redirect(url_for("login_index"))
    login_user(user, remember=True)
    return redirect_next_url(fallback=url_for("home"))
```

- **Проверка наличия пользователя в базе данных и регистрация нового пользователя:** Процесс проверки наличия пользователя в базе данных регистрации нового пользователя в Miminet аналогичен описанному в функции для Яндекса, включая обработку возможных ошибок и перенаправление на страницу входа в случае их возникновения.

- **Авторизация пользователя в системе:** После подтверждения наличия пользователя в базе данных или создания новой учетной записи, осуществляется вход пользователя в систему Miminet и перенаправление его на домашнюю страницу.

3.3. Интерфейс страницы авторизации в Miminet

На изображении ниже представлен интерфейс страницы авторизации веб-приложения Miminet, который предлагает пользователям варианты входа через Яндекс и Telegram.

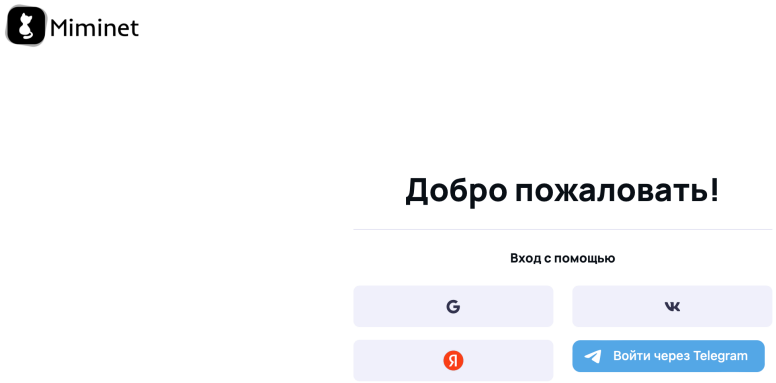

Рис. 3: Страница авторизации в Miminet

После консультации с научным руководителем было решено сохранить текущий дизайн кнопки авторизации через Яндекс, дополнив его визуальным элементом в виде иконки Яндекса. Также были внесены изменения, включая добавление цветовой подсветки кнопки при наведении курсора.

Telegram Login Widget интегрируется на веб-сервис с использованием элемента iframe, что ограничивает возможности его кастомизации со стороны разработчиков веб-приложений. В связи с этим было принято решение о сохранении стандартного вида виджета Telegram с целью обеспечения его корректной функциональности и совместимости.

4. Тестирование

Для тестирования использовался фреймворк Pytest [11] и дополнительные инструменты, такие как mocker и фикстуры.

Pytest представляет собой платформу для тестирования Python, позволяющую писать различные типы тестов программного обеспечения, включая модульные тесты, интеграционные тесты, сквозные тесты и функциональные тесты.

Фикстуры [12] в PyTest обеспечивают механизм для настройки тестового окружения, предоставляя необходимые ресурсы и конфигурации для выполнения тестов. Они могут быть использованы для инициализации экземпляров приложения, подключения к базе данных или предоставления фиктивных данных для тестирования.

Mocker [5] в PyTest используется для создания моков - фиктивных объектов, которые имитируют поведение реальных объектов в тестах. Это помогает изолировать тестируемый код от внешних зависимостей, таких как внешние сервисы или базы данных.

4.1. Тестирование аутентификации через Яндекс

В ходе тестирования были рассмотрены следующие аспекты:

- **Запрос к авторизационному серверу Яндекса:** Проверка процесса инициирования аутентификации пользователя с использованием сервера авторизации Яндекса.
- **Запрос к серверу ресурсов Яндекса:** Проверка процесса получения данных пользователя через сервер ресурсов Яндекса.
- **Установка состояния в сессии пользователя:** Проверка механизма сохранения состояния аутентификации в сессии пользователя.
- **Обработка данных пользователя в Miminet:** Проверка способности системы обрабатывать информацию о пользователе, по-

лученную от Яндекса, и создавать соответствующую запись в базе данных для новых пользователей.

- **Перенаправления на указанные URL:** Проверка механизмов перенаправления пользователя на необходимые страницы после аутентификации или в случае возникновения ошибок.

- **Обработка исключений и ошибок:** Проверка способности системы реагировать на исключения и ошибки, возникающие в процессе аутентификации через Яндекс.

4.2. Тестирование аутентификации через Telegram

В ходе тестирования были рассмотрены следующие аспекты:

- **Обработка данных авторизации Telegram:** Проверка механизмов обработки данных авторизации, полученных от Telegram, с использованием сравнения хешей и даты авторизации.

- **Обработка данных пользователя в Miminet:** Проверка способности системы обрабатывать информацию о пользователе, полученную от Telegram, и создавать соответствующую запись в базе данных для новых пользователей.

- **Перенаправления на указанные URL:** Проверка механизмов перенаправления пользователя на необходимые страницы после аутентификации или в случае возникновения ошибок.

- **Обработка исключений и ошибок:** Проверка способности системы реагировать на возникающие исключения и ошибки в процессе аутентификации через Telegram.

Заключение

В процессе учебной практики были выполнены следующие задачи:

1. Проведен обзор Identity-провайдеров
2. Изучена документация выбранных Identity-провайдеров
3. Реализована аутентификацию через выбранные Identity-провайдеры:
   - 3.1 Реализована инициацию процесса аутентификации
   - 3.2 Реализована обработка ответа аутентификации
4. Добавлены соотвествующие кнопки на страницу входа Miminet
5. Проведено тестирование добавленных функций

Программный код выполненной работы доступен на GitHub [3] [4]

[3] https://github.com/mimi-net/miminet/pull/104

[4] https://github.com/mimi-net/miminet/tree/Katerina/

Список литературы

[1] Mediascope Cross Web, Аудитория СМИ, интернет, декабрь 2023 года, Россия 0+, возраст 12+. — URL: https://mediascope.net/data/ (дата обращения: May 1, 2024).

[2] OAuth 2.0. — URL: https://oauth.net/2/ (дата обращения: May 1, 2024).

[3] OAuth Mail. — URL: https://yandex.ru/dev/id/doc/ru/ (дата обращения: May 1, 2024).

[4] OAuth OK. — URL: https://yandex.ru/dev/id/doc/ru/ (дата обращения: May 1, 2024).

[5] Oliveira Bruno. Pytest mocker. — URL: https://pytest-mock.readthedocs.io/en/latest/ (дата обращения: May 1, 2024).

[6] OpenID Connect 1.0. — URL: https://openid.net/specs/openid-connect-core-1_0.html (дата обращения: May 1, 2024).

[7] Pallets. Flask's documentation. — URL: https://flask.palletsprojects.com/en/3.0.x/ (дата обращения: May 1, 2024).

[8] Telegram Bot API. — URL: https://core.telegram.org/bots (дата обращения: May 1, 2024).

[9] Telegram Login Widget. — URL: https://core.telegram.org/widgets/login (дата обращения: May 1, 2024).

[10] Tilmans Olivier. IPMininet. — URL: https://ipmininet.readthedocs.io/en/latest/ (дата обращения: May 1, 2024).

[11] holger krekel, pytest-dev team. Pytest. — URL: https://docs.pytest.org/en/8.2.x/ (дата обращения: May 1, 2024).

[12] holger krekel, pytest-dev team. Pytest fixtures. — URL: https://docs.pytest.org/en/6.2.x/fixture.html (дата обращения: May 1, 2024).

[13] Зеленчук И. В. Miminet. — URL: https://miminet.ru (дата обращения: May 1, 2024).

[14] Методические рекомендации по интеграции с REST API Цифрового профиля. — URL: https://digital.gov.ru/ru/documents/7166/ (дата обращения: May 1, 2024).

[15] Пресс-релиз VK по результатам за 12 мес. 2023. — URL: https://vk.company/ru/investors/results/ (дата обращения: May 1, 2024).

[16] РФ Министерство связи и массовых коммуникаций. Статистика количества пользователей портала «Госуслуги». — URL: https://digital.gov.ru/ru/events/49139/ (дата обращения: May 1, 2024).

[17] Регламент информационного взаимодействия Участников с Оператором ЕСИА и Оператором эксплуатации инфраструктуры электронного правительства. — URL: https://digital.gov.ru/ru/documents/4244/ (дата обращения: May 1, 2024).

[18] Руководство по интеграции с API Яндекс ID. — URL: https://yandex.ru/dev/id/doc/ru/ (дата обращения: May 1, 2024).

[19] Руководство пользователя ЕСИА. — URL: https://digital.gov.ru/ru/documents/6182/ (дата обращения: May 1, 2024).

[20] Федеральный закон от 31.07.2023 № 406-ФЗ. — URL: http://publication.pravo.gov.ru/document/0001202307310022?index=1 (дата обращения: May 1, 2024).