Введение

1. **Цель документа:** Установить и зафиксировать четкие однозначные требования к разрабатываемому продукту, установить взаимопонимание внутри команды во время разработки продукта и избежать двусмысленности. Документ предназначен для использования в качестве основы на всех этапах жизненного цикла продукта.
2. **Область применения системы:** Продукт будет доступен для использования в виде клиент-серверного веб-приложения, которое поможет пользователю осваивать команды для bash-терминала в GNU/Linux.
3. **Ссылки на другие документы/стандарты:** ISO/IEC/IEEE 29148:2018.

Общая характеристика системы

1. **Перспектива системы.** Разрабатываемая система является клиент-серверным веб-приложением с чат-ботом на основе LLM-модели, предназначенным для помощи в освоении bash-терминала в GNU/Linux. Клиентская часть работает в браузерах на ПК. Серверная часть развернута на сервере и обрабатывает поступающие запросы из клиентской части.
2. **Функции системы.** Продукт обладает следующими функциями:
   - Генерирование команды/набора команд для bash-терминала в GNU/Linux по запросу пользователя.
   - Разбор команды/набор команд для bash-терминала в GNU/Linux по запросу пользователя.
   - Регистрация/авторизация пользователя.
3. **Пользователи и заинтересованные стороны.**
   - Домашние пользователи дистрибутивов Linux.
   - Начинающие пользователи дистрибутивов Linux.
   - Разработчики и инженеры, работающие в дистрибутивах Linux.
   - Системные администраторы.
4. **Технологический стек.**
   - **Frontend:** React, Tailwind CSS, TypeScript, REST API.
   - **Backend:** Golang, Redis, RabbitMQ, REST API.
   - **ML-сервис:** Python, PyTorch, Transformers, Draccus, FastAPI, REST API
   - **База данных:** PostgreSQL.
   - **Инфраструктура:** Docker, Docker Compose, Nginx.
5. **Предположения и зависимости.** Предполгается, что:

- Пользователь знаком с Linux-дистрибутивами.
- Пользователь пользуется веб-приложением с декстопного браузера.
- Веб-приложение развернуто локально на ПК для демонстрации и на сервере для продакшена.
- ML-модель предобучена, и требуется её дообучить.

Ограничения:

- Чат-бот работает только с командами в GNU/Linux.
- Модель сжата до 16-bit для инференса и обучения.
- Ответ чат-бота не гарантирует 100% правильности и безопасности.

Функциональные требования

1. Регистрация с помощью электронной почты.
2. Обращение к чат-боту и получение ответ от него.
3. Новая генерация ответа чат-ботом, вместо старого.
4. При получении чат-ботом команды/набора команд:
    - Подробный рассказ, что делает каждая команда и за что отвечает каждый флаг и параметр в полученных командах.
    - Если команда/набор команд, содержат ошибки, то чат-бот указывает на них, исправляет, пишет рабочий вариант, а дальше выполняет, первый пункт.
    - Если команда/набор команд являются небезопасными, не актуальными или имеются альтернативы лучше, то чат-бот выполняет первый пункт и второй пункт (при необходимости), после чего он предупреждает пользователя о проблеме команды/набора команд и предлогает альтернативу, если такое возможно. Для альтернативы также требуется выполнить первый пункт.
    - Если пользователь ввел запрос на помощь с составлением команд, то чат-бот составляет команду/набор команд, решающий проблему пользователя, выполнив первый пункт.
    - Если пользователь ввел запрос, не связанный с помощью с bash-терминалом в GNU/Linux, то чат-бот отвечает ему, что он не может помочь с его проблемой, так как специализируется только на помощи с bash-терминалом в GNU/Linux.
5. Оценка ответа чат-бота (лайк/дизлайк), после чего, если ответ был оценен, он заносится в базу данных.
6. Обращение к своим прошлым чатам.
7. Возможность удаления старых чатов.

Нефункциональные требования

1. **Производительность:**
   - Продукт должен выдерживать работу 100 пользователей одновременно.
   - Время отклика сервиса должно составлять 1-2 секунды.
   - Пользователь должен получить ответ чат-бота в течение 7-10 секунд.
2. **Надежность.**
   - Среднее время безотказной работы ≥ 99% в месяц.
3. **Безопасность.**
   - Хранение паролей пользователей в виде хэшов, полученных с помощью метода SCRAM-SHA-256.
4. **Масштабируемость.**
   - Продукт должен иметь возможность добавления новых модулей и улучшения старых без переработки архитектуры.
5. **Поддерживаемость.**
   - Возможность изменять каждый модуль не влияя на работу остальных.
6. **Совместимость.**
   - Совместимость с браузерами Google Chrome, Firefox, Microsoft Edge, Yandex.

Интерфейсы

1. **Пользовательский интерфейс.**

Веб-приложение должно содержать 4 страницы: Главная страница с описанием продукта, страница с чат-ботом, страница авторизации и страница регистрации.

Страница авторизации:

- Отображение ошибки при вводе некорректной электронной почты во время авторизации.
- Отображение ошибки при вводе некорректного пароля во время авторизации.
- Отображение ошибки при вводе неизвестного юзернейма во время авторизации.
- Ввод пароля должен скрываться символом «*», но должна иметься возможность отключить это.
- При нажатии кнопки «Войти» после успешной аутентификации пользователь должен попасть на страницу с чат-ботом.
- При нажатии кнопки «Назад» пользователь должен попасть на главную страницу.

Страница регистрации:

- Отображение сообщения о том, что код отправлен на почту при регистрации.
- Отображение ошибки при вводе некорректного кода при регистрации.
- Отображение ошибки при вводе существующего юзернейма при регистрации.
- Ввод пароля должен скрываться символом «*», но должна иметься возможность отключить это.
- При нажатии кнопки «Завершить регистрацию» пользователь должен попасть на страницу с чат-ботом.
- При нажатии кнопки «Назад» пользователь должен попасть на главную страницу.

Главная страница:

- Информация о том, какую проблему решает наше клиент-сервисное веб-приложение.
- При нажатии кнопки «Регистрация» пользователь должен попасть на страницу регистрации.
- При нажатии кнопки «Авторизация» пользователь должен попасть на страницу авторизации.

Страница с чат-ботом:

- В центре страницы располагается поле для ввода запроса, которое представлено прямоугольником с закругленными краями.
- Над полем для ввода запроса должна быть надпись «О чем сегодня Вам хочется узнать?»
- При появлении текста в поле для ввода запроса должна появиться кнопка «Отправить».
- При нажатии на кнопку «Отправить» страница должна динамически обновиться и стать похожей на диалог в мессенджере, где с правой стороны страницы располагаются сообщения пользователя, а с левой стороны страницы располагаются ответы чат-бота. Запрос должен быть отправлен модели, сгенерированный ответ который должен быть отправлен обратно на клиентский сервер.
- После генерации первого ответа в новом чате он заносится в базу данных.
- С левой стороны страницы располагается вертикальное поле, внизу которого расположена кнопка выхода из аккаунта, а сверху расположена кнопка создания нового чата. Между ними расположены старые чаты.
- При наведении курсора на старый чат появляется иконка корзины, при нажатии на которую появляется предупреждающее сообщение: если выбрать «Да», то чат удаляется, если выбрать «Нет», то сообщение пропадает без каких-либо дополнительных действий.

- При нажатии кнопки выхода из аккаунта пользователь должен получить предупреждающее сообщение, в котором при нажатии кнопки «Да» пользователь попадает на главную страницу, а при нажатии кнопки «Нет» — предупреждающее сообщение пропадает без каких-либо дополнительных действий.
- При нажатии кнопки создания нового чата страницы должна динамически обновиться и появится поле для ввода запроса по середине экрана с текстом над ним, как описано в первом и втором пунктах.

2. **Программные интерфейсы (API).**

Система должна предоставлять REST API для операций с пользователями, которое предназначено для использования через протокол HTTPS. Запросы и ответы должны быть в формате JSON.

HTTP-запросы:

- **POST /users** Входные данные: name (строковый тип данных), surname (строковый тип данных), email (строковый тип данных), password (строковый тип данных).
  Выходные данные: temp_id (беззнаковый целочисленный тип данных).
- **POST /users/{temp_id}/verifications** Входные данные: code (строковый тип данных).
  Выходные данные: access_token (строковый тип данных) и refresh_token (строковый тип данных). Хранение refresh_token должно происходить в HttpOnly cookie, а access_token в памяти JS, передаются в заголовке.
- **POST /auth/login.** Входные данные: email (строковый тип данных), password (строковый тип данных).
  Выходные данные: access_token (строковый тип данных, срок действия 60 минут), refresh_token (строковый тип данных, срок действия 2 недели). Хранение refresh_token должно происходить в HttpOnly cookie, а access_token в памяти JS, передаются в заголовке.

- **POST /auth/logout.** Входные данные: access_token (строковый тип данных) в заголовке, refresh_token удаляется из cookie.
- **POST /chats.** Входные данные: access_token (строковый тип данных) в заголовке, message (строковый тип данных).

  Выходные данные: uuid (строковый тип данных), bot_message (строковый тип данных).
- **GET /chats/{uuid}.** Входные данные: uuid (строковый тип данных). Выходные данные: JSON-файл, содержащий сообщения пользователя и сообщения чат-бота.
- **DELETE /chats/{uuid}.** Входные данные: uuid (строковый тип данных).
- **POST /chats/{uuid}/bot-messages.** Входные данные: uuid (строковый тип данных), message (сообщение, отправленное пользователем, строковый тип данных).
- **PUT /chats/{uuid}/bot-messages.** Входные данные: uuid (строковый тип данных), message (сообщение, отправленное пользователем, строковый тип данных).
- **POST /chats/{uuid}/bot-messages/evaluations.** Входные данные: message (строковый тип данных), evaluation (оценка сообщения чат-ботом, логический тип данных).

3. **Аппаратные интерфейсы.**

   Сервер для LLM-модели:

   - **RAM:** Минимально 128Gb, рекомендуемо 256Gb.
   - **Storage:** Минимально 50Gb, рекомендуемо 150Gb.
   - **OS:** Linux (Ubuntu/Arch/CentOS).
   - **Видеокарта:** NVIDIA A100 40GB, 1 шт. минимально, 2 шт. рекомендуемо .
   - **CPU:** 16–24 ядра с частотой более 2.5 ГГц минимально, 32+ ядер с частотой более 3 ГГц.

- **Скорость сети:** не менее 1 Gbps.

Хранение данных в базе данных

1. **Таблица users.** Таблицы содержит информацию о зарегистрированных пользователях. Связь «один ко многим» с таблицами chats и messages.
   Колонки:
   - id — id пользователя, целочисленный беззнаковый тип данных.
   - name — имя пользователя, строковый тип данных.
   - surname — фамилия пользователя, строковый тип данных.
   - email — электронная почта пользователя, строковый тип данных.
   - password — хэш SCRAM-SHA-256, строковый тип данных.
2. **Таблица chats.** Таблица содержит информацию о всех созданных чатах. Связь «один к одному» с таблицей users и «один ко многим» с таблицей messages.
   Колонки:
   - id — id чата, целочисленный беззнаковый тип данных.
   - uuid — uuid чата, строковый тип данных.
   - user_id — id пользователя, которому принадлежит данный чат.
3. **Таблица messages.**
   Таблица содержит информацию о всех сообщениях пользователя и чат-бота в чатах. Связь «один к одному» с таблицами users и chats.
   Колонки:
   - id — id сообщения, целочисленный беззнаковый тип данных.
   - chat_id — id чата, которому принадлежит сообщение, целочисленный беззнаковый тип данных.
   - user_id — id пользователя, которму принадлежит сообщение, целочисленный беззнаковый тип данных. Может быть NULL, если сообщение принадлежит чат-боту.
   - message — текст сообщения, строковый тип данных.
   - grade — оценка, присвоенная сообщению, отправленным чат-ботом, логический тип данных. Имеет значение NULL для сообщений пользователей или неоцененных сообщений чат-бота.

4. **Таблица tokens.**

   Таблица содержит долгосрочные JWT-токены для обновления краткосрочных JWT-токенов. Связь «один к одному» с таблицей users.

   Колонки:

   - id — id JWT-токена, целочисленный беззнаковый тип данных.
   - refresh_token — JWT-токен, строковый тип данных, может быть NULL при истечении срока.
   - creation_data — дата создания JWT-токена.
   - validity_period — срок действия JWT-токена.

5. **Таблица good_answers.**

   Таблица содержит ответы чат-бота, которые были помечены пользователем лайком.

   Колонки:

   - id — id ответа, целочисленный безнаковый тип данных.
   - content — содержание ответа чат-бота, строковый тип данных.
   - model — имя LLM-модели.

6. **Таблица bad_answers.**

   Таблица содержит ответы чат-бота, которые были помечены пользователем дизлайком.

   Колонки:

   1. id — id ответа, целочисленный безнаковый тип данных.
   2. content — содержание ответа чат-бота, строковый тип данных.
   3. model — имя LLM-модели.