Проектирование информационных систем
Домашнее задание № 3

ФИО: Леонов В.В.
TG:@leery_corsair
Группа М23–524

18 октября 2023

Содержание

1 Постановка задачи

Перед вами на экране лицо виртуального персонажа с богатой мимикой, способное выражать различные мимические комплексы на основе FACS: осуществлять движения мышц в области лба, бровей, глаз, рта, губ, подбородка, носогубных складок, щёк. Персонаж может поддерживать беседу и выражать различные эмоции в ответ на реплики пользователя. Мимика, интонация и реплики персонажа определяются моделью или человеком (API для управления предоставляется).

Необходимо разработать ТЗ к клиентской части приложения (максимально полно).

2 Требования к клиентской части

2.1 Функциональные требования

Функциональные требования являются важной частью спецификации приложения и определяют, какие функции и возможности должны быть реализованы в приложении. Деление требований на пользовательские и требования домена (FACS — Facial Action Coding System) может помочь структурировать их для лучшего понимания и управления.

Пользовательские требования описывают, как пользователи в качестве отдельных сущностей будут взаимодействовать с приложением и какие функции будут доступны:

1. Регистрация и аутентификация (многофакторная) пользователей.
2. Вход и выход из системы.
3. Создание и редактирование профилей пользователей.
4. Взаимодействие с другими пользователями (обмен сообщениями).

5. Управление настройками аккаунта.

6. Конфигурирование параметров уведомлений и оповещений.

Функциональные требования, связанные с FACS-подсистемой:

1. С точки зрения персонажа и его кастомизации:

    1.1. Наличие различных анимаций движения мышц в области лба, бровей, глаз, рта, губ, подбородка, носогубных складок, щёк персонажа.

    1.2. Наличие редактора виртуального персонажа (части лица, прическа, голос, раса/человекоподобные существа/животные, визуальный стиль: реализм, мультипликация, аниме и т.д.) и фонового изображения.

    1.3. Возможность импорта/экспорта конфигурационных файлов.

    1.4. Поддержка множества персонажей.

2. С точки зрения действий персонажа:

    2.1. Наличие различных сцериев взаимодействия: повествование за заданную тему, диалог и игровой формат.

    2.2. Наличие персонализированного психологического образа виртуального персонажа, на основе которого строятся ответные реакции пользователю.

    2.3. Возможность построения ответов с помощью преднастроенной системы, так с дополнительным использованием сети Интернет.

    2.4. Возможность построения ответов на основе контекста.

    2.5. Возможность построения повторного альтернативного ответа.

2.6. Наличие фильтрации шокирующего и 18+ контента, а также опции детского режима.

2.7. Синхронизация лицевых анимаций и реплик.

3. С точки зрения взаимодействия персонажа и пользователя:

3.1. Поддержка взаимодействия с помощью жестов.

3.2. Поддержка взаимодействия с помощью голоса.

3.3. Поддержка взаимодействия с помощью текста.

3.4. Поддержка взаимодействия на различных языках.

3.5. Поддержка интеграции со сторонними сервисами с помощью API.

3.6. Поддержка множественных диалогов с сохранением истории запросов/ответов.

2.1.1 Организация потоков данных

Источником первичных входных данных для системы служат действия пользователя в настольном или мобильном приложении, далее они преобразуются и отправляются на сервер в виде HTTP-запросов по сети Интернет.

Выходные данные из системы выводятся пользователю на экране настольного или мобильного приложения, которое получает данные от сервера по сети Интернет в результате HTTP-запросов и преобразует их для отображения пользователю.

2.2 Требования к интерфейсу

Требования к интерфейсу клиентского приложения включают в себя спецификации и ожидания по дизайну, внешнему виду и поведению пользовательского интерфейса (UI). Эти требования помогают обеспечить удовлетворение потребностей пользователей, создать приятное взаимодействие и улучшить общий опыт:

1. Поддержка мультиязычности интерфейса, в том числе поддержка разных форматов дат, валют и других локальных особенностей.

2. Поддержка персонализации: наличие как светлой, так и темной темы, а также режима цветовой слепоты; добавление, удаление и перемещение виджетов на экране.

3. Соответствие единству стилистической схемы, шрифтов, иконок и досок настроения.

4. Наличие плавных анимаций для компонентов интерфейса и надежная обратная связь при взаимодействии с пользователем.

5. Адаптивность под различные платформы (Desktop, Mobile) с корректным масштабированием.

6. Поддержка широкоформатных устройств (вплоть до 32:9), а также устройств с повышенным разрешением (вплоть до 8K).

7. Интуитивная система навигации со всплывающими подсказками.

2.3 Системные требования

2.3.1 Общесистемные требования

Разрабатываемое клиентское приложение должно быть полным по следующим критериям:

1. Возможность масштабироваться горизонтально и вертикально, что позволит увеличивать производительность с ростом нагрузки.

2. Устойчивость к высоким нагрузкам — обработка большого количества запросов и операций без серьезных сбоев или деградации производительности.

3. Кэширование — применение кеширования данных на стороне клиента, чтобы уменьшить нагрузку на сервер и сократить время отклика.

4. Асинхронные операции и многозадачность для эффективной обработки множества запросов одновременно.

5. Поддержка всех основных операционных систем: Windows, Linux, MacOS, Android и IOS.

6. Наличие режима функционирования с потреблением ограниченного числа ресурсов, а также энергоэффективного режима для мобильных устройств.

2.3.2 Контроль входной информации

Контроль входной информации — это процесс проверки, фильтрации и обработки данных, которые поступают от пользователя или других источников на стороне клиента перед их передачей на сервер. Это важный аспект безопасности и надежности клиент-серверных приложений. Необходимо обеспечить следующие меры:

1. Валидация — верификация данных, вводимых пользователями, на соответствие ожидаемым форматам и значениям.

2. Санитизация данных — фильтрация входных данных, чтобы удалить или экранировать потенциально опасные символы/скрипты.

3. Обработка ошибок — обеспечение обработки ошибок и некорректных данных, чтобы предотвратить сбои приложения из-за неправильных данных.

4. Защита от межсайтовой подделки запросов — реализация токенов CSRF, чтобы предотвратить отправку злоумышленниками фальшивых запросов от имени пользователя.

5. Все данные на сервер передаются в зашифрованном виде.

2.3.3 Контроль выходной информации

Контроль выходной информации важен для обеспечения безопасности и конфиденциальности данных, которые передаются пользователям или другим системам. Следует реализовать следующие методы и практики контроля выходной информации:

1. Экранирование данных — перед выводом данных на экран или передачей данных на клиентскую сторону, необходимо удостовериться, что все потенциально опасные символы экранированы. Это предотвратит атаки типа XSS.

2. Защита от атак CSRF — при передаче данных на клиентскую сторону, необходимо удостовериться, что выполняется защита от атак CSRF, чтобы предотвратить изменение данных без согласия пользователя.

3. Контроль доступа к API — при взаимодействии с внешними API, необходимо удостовериться, что доступ к API ограничен и проверяется на наличие соответствующих разрешений.

4. Управление кешированием — при кешировании данных на стороне клиента, необходимо удостовериться, что данные в кеше обновляются при изменении на сервере и не сохраняют конфиденциальную информацию.

5. Логирование безопасности — ведение журнала событий безопасности, чтобы отслеживать все события, связанные с выходной информацией, и быстро реагировать на возможные инциденты.