Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования



Факультет информационных технологий

Кафедра «Инфокогнитивные технологии»

Направление подготовки: «Информатика и вычислительная техника»

Образовательная программа (профиль): «Веб-технологии»

ОТЧЕТ

по практике (учебной/прокатной)

на тему: «Мобильное приложение “Ваш помощник в магазине: сканируй, слушай, выбирай”»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | Выполнил:  Студент  Группа  Дата  Проверил  Преподаватель | | Суярков Владислав Валерьевич  243-321  25.06.25  Чернова Вера Михайловна |
|  |  |  |  | |

Москва

2025

Аннотация

В ходе практики была разработана концепция и прототип мобильного приложения, предназначенного для помощи покупателям в магазине. Основная идея приложения — возможность сканировать товар, получать аудиоописание и выбирать нужные позиции. В рамках проекта выполнена схема сайта на HTML, создана визуальная часть интерфейса в Figma, а также реализован демонстрационный чат-бот на языке Python, который имитирует взаимодействие с пользователем.

Общее задание на Проектной деятельности

Пояснение выбранной темы:

Тема проекта выбрана с целью создания удобного и доступного инструмента для покупателей, который облегчает процесс выбора товаров в магазине, особенно для людей с ограничениями по зрению или для тех, кто предпочитает аудиоформат информации.

Актуальность:

Современные технологии позволяют значительно улучшить опыт покупок, делая его более комфортным и инклюзивным. Сканирование товаров и получение голосовой информации помогает сэкономить время и повысить информированность покупателей.

Цели проекта:

Разработать прототип мобильного приложения, объединяющего функции сканирования, озвучивания и выбора товаров.

Создать удобный и интуитивно понятный интерфейс.

Реализовать демонстрационный чат-бот для имитации взаимодействия с пользователем.

Задачи:

Изучить существующие решения и определить требования к приложению.

Спроектировать структуру сайта и интерфейс.

Реализовать базовую версию сайта на HTML.

Создать визуальный прототип в Figma.

Разработать демонстрационный чат-бот на Python.

3. Планы работы по практике

Анализ существующих приложений и технологий для помощи покупателям.

Составление технического задания и требований к приложению.

Разработка схемы сайта на HTML.

Создание визуального прототипа интерфейса в Figma.

Программирование демонстрационного чат-бота на Python.

Тестирование и корректировка разработанных компонентов.

Подготовка отчёта и презентации результатов.

Содержательные главы

Аналитическая часть

Обзор рынка мобильных приложений для помощи покупателям.

На сегодняшний день, рынок мобильных приложений для помощи покупателям продолжает развиваться, чему способствуют доступность смартфонов и расширение покрытия мобильных сетей, включая переход на технологии пятого поколения (5G). Онлайн-шопинг и мобильный банкинг становятся стандартом для большинства пользователей, а готовность платить за цифровые сервисы растёт.

Примеры приложений:

Приложения для интернет-магазинов. Например, доступ к истории заказов, каталог и поиск товаров, оформление заказа и оплата.

Чат-боты и голосовые помощники. Они упрощают поиск, выбор и покупку товаров, а также предоставляют круглосуточную поддержку.

Приложения с элементами игры (геймификация). Например, бонусы, скидки и купоны за выполненные задания, которые делают покупки интереснее.

Анализ потребностей целевой аудитории (покупатели с ограничениями зрения, также могут быть и занятые пользователи и т.д.).

Целевая аудитория мобильного приложения включает несколько групп пользователей с различными потребностями, среди которых основное внимание уделяется покупателям с ограничениями зрения, а также занятым пользователям и другим категориям, предпочитающим аудиоформат информации.

Для более глубокого понимания потребностей целевой аудитории было проведено исследование, в ходе которого собрана статистика по уровню зрения различных групп населения. Представленная диаграмма (Рис.1) наглядно демонстрирует распределение пользователей в зависимости от степени зрительных возможностей — от людей с отличным зрением до тех, кто практически не способен функционировать без специальных средств коррекции.

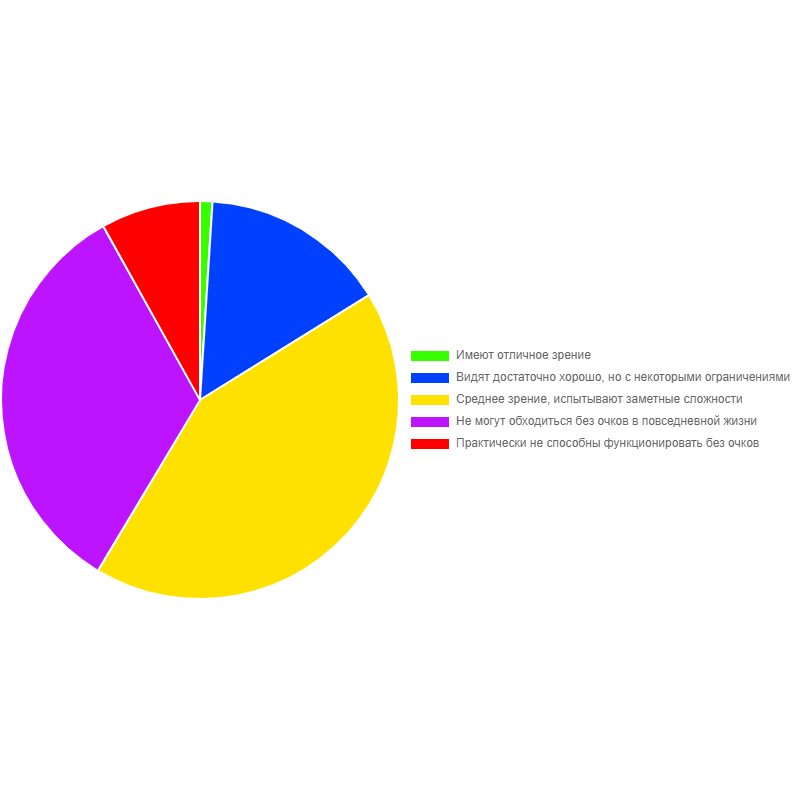


Рисунок 1 - статистика людей с разным уровнем зрения в России

Для более полного понимания причин и масштабов проблем со зрением среди взрослого населения Российской Федерации была проанализирована общая заболеваемость глаз и его придаточного аппарата. На следующей диаграмме (Рис.2) отражено распределение различных заболеваний, включая нарушения рефракции, катаракту, глаукому и другие патологии.

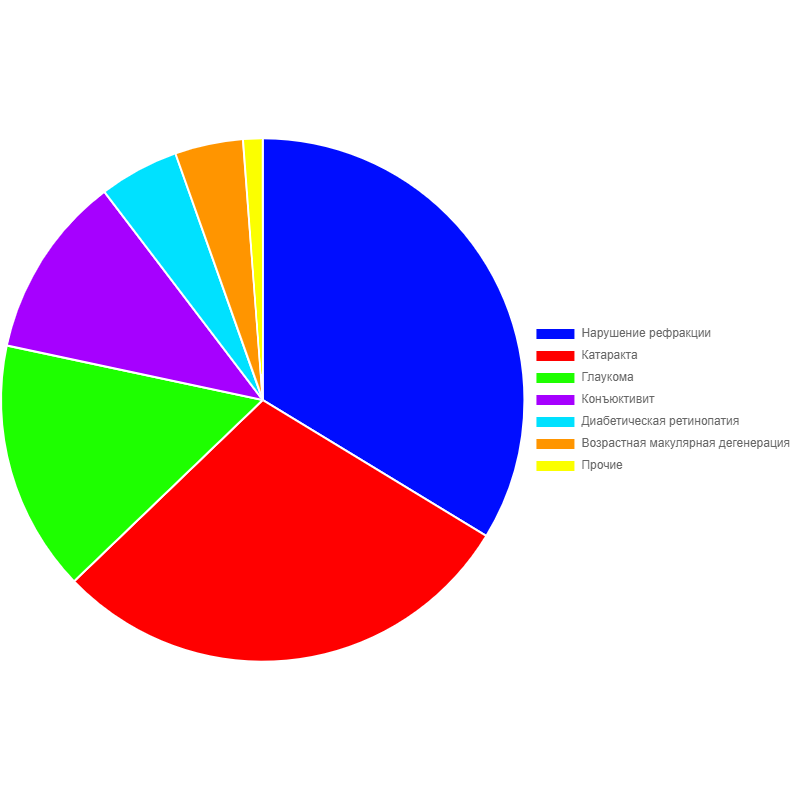


Рисунок 2 - Общая заболеваемость болезнями глаза и его придаточного аппарата взрослого населения РФ

Изучение технологий сканирования штрих-кодов и преобразования текста в речь.

В современных мобильных приложениях для помощи пользователям с ограничениями зрения ключевую роль играют технологии сканирования штрих-кодов и преобразования текста в речь (Text-to-Speech, TTS). Эти технологии обеспечивают быструю и удобную передачу информации о товарах, делая процесс выбора более доступным и автономным.

Технологии сканирования штрих-кодов

Сканирование штрих-кодов основано на использовании камеры устройства и специализированного программного обеспечения для распознавания различных типов кодов (например, UPC, EAN, QR-коды). Современные алгоритмы обеспечивают высокую скорость и точность считывания даже при плохом освещении или частичном повреждении кода. Интеграция таких решений позволяет мгновенно получать информацию о товаре, его цене и характеристиках.

Технологии преобразования текста в речь (TTS)

TTS-системы преобразуют цифровой текст в естественное звучание речи, что особенно важно для пользователей с нарушениями зрения. Современные движки TTS поддерживают различные языки, интонации и темпы речи, обеспечивая комфортное восприятие информации. Использование нейросетевых моделей позволяет достигать высокой естественности звучания и адаптировать голос под предпочтения пользователя.

Интеграция и применение

Объединение технологий сканирования штрих-кодов и TTS в одном приложении создаёт мощный инструмент для помощи покупателям с ограничениями зрения. Пользователь может просто навести камеру на товар, после чего приложение автоматически озвучит его описание и характеристики, что значительно упрощает процесс выбора и повышает уровень автономии.

Определение функциональных требований к приложению.

Для создания эффективного и инклюзивного мобильного приложения, ориентированного на пользователей с нарушениями зрения, необходимо четко сформулировать функциональные требования, обеспечивающие удобство, доступность и надежность работы. Основные функциональные требования включают:

Сканирование штрих-кодов

Возможность распознавания различных типов штрих-кодов (UPC, EAN, QR-коды).

Высокая точность и скорость считывания при различных условиях освещения и частичном повреждении кода.

Автоматическое обнаружение и фокусировка на штрих-коде с помощью камеры устройства.

Озвучивание информации (Text-to-Speech)

Преобразование текстовой информации о товаре в речь с использованием естественных голосов.

Поддержка нескольких языков и возможность выбора голоса и скорости озвучивания.

Возможность повторного воспроизведения и паузы озвучивания.

Пользовательский интерфейс

Интуитивно понятный и минималистичный дизайн с крупными элементами управления.

Поддержка экранных читалок и жестов для пользователей с нарушениями зрения.

Возможность настройки контрастности и размера шрифта.

Дополнительные функции

Сохранение истории сканированных товаров для быстрого доступа.

Возможность получения подробной информации о составе, сроках годности и рекомендациях по использованию.

Интеграция с внешними базами данных для расширения ассортимента товаров.

Производительность и безопасность

Быстрая обработка данных без задержек.

Защита персональных данных и конфиденциальность пользователя.

Поддержка работы в офлайн-режиме для базовых функций.

Выводы по выбору архитектуры и инструментов разработки.

При разработке мобильного приложения для помощи людям с нарушениями зрения важно подобрать архитектуру и инструменты, обеспечивающие масштабируемость, производительность и удобство поддержки, а также интеграцию с необходимыми технологиями сканирования и преобразования текста в речь.

Архитектура приложения

Модульная архитектура (например, MVVM или Clean Architecture) позволит разделить логику представления, бизнес-логику и работу с данными, что облегчит тестирование и поддержку.

Использование слоев абстракции обеспечит гибкость при замене или обновлении компонентов (например, движков TTS или библиотек для сканирования).

Поддержка офлайн-режима требует реализации локального кэширования данных и обработки запросов без подключения к сети.

Инструменты и технологии разработки

Кроссплатформенные фреймворки (Flutter, React Native) позволят создать единое приложение для iOS и Android с минимальными затратами ресурсов.

Для сканирования штрих-кодов рекомендуется использовать проверенные библиотеки, например, ZXing, ML Kit от Google или встроенные API платформ.

Для преобразования текста в речь можно интегрировать нативные TTS-сервисы (Android TTS, iOS AVSpeechSynthesizer) или сторонние облачные решения с поддержкой нескольких языков и качественных голосов.

Инструменты для обеспечения доступности (Accessibility APIs) должны быть использованы для поддержки экранных читалок и жестов.

Средства тестирования и обеспечения качества

Автоматизированное тестирование UI и функциональности для проверки доступности и корректной работы всех функций.

Инструменты мониторинга производительности и сбора отзывов пользователей для постоянного улучшения приложения.

Проектная часть

Разработка структуры сайта: описание страниц, навигации, основных элементов (схема сайта на HTML).

Данный сайт всего лишь представляет схему, того, как сайт будет оформлен, и какие технологии будут использованы. Реализована страница будет по следующей схеме (Рис. 3).

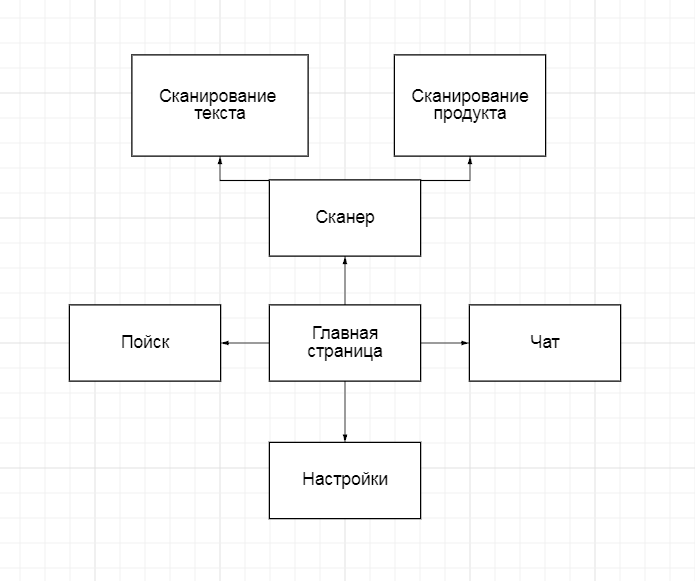


Рисунок 3 – Блок схема страниц приложения.

Для визуального дизайна приложения будет использована интернет-программа Figma. Главное в дизайне: Достаточно крупные буквы и иконки, дизайн не слишком сложный, чтобы всё легче было разместить по всему экрану, более желательно чёрные буквы на ярком фоне, поскольку, лучше всего видны. В будущем планируется добавить, возможность шрифта по Брайлю, для перевода текста приложения в специальный шрифт для слепых. Переключатель планируется добавить в главное меню. Чтобы можно было переключить оттуда и обратно.

На языке HTML будет ещё выполнен сайт Лендинг, в котором будут описаны все преимущества данного приложения, а также, что в этом приложении будет реализовано

Реализация демонстрационного чат-бота на Python: описание логики работы, сценариев взаимодействия, используемых библиотек.

Логика работы

Инициализация: при создании объекта бота инициализируются словарь истории покупок, переменная частоты покупок и словарь с информацией о продуктах.

Приветствие: бот приветствует пользователя и сообщает о начале взаимодействия.

Сбор данных: бот запрашивает у пользователя, как часто он делает покупки, предлагая три варианта выбора.

Персонализированные предложения: в зависимости от выбранной частоты покупок бот выводит соответствующие вопросы и рекомендации.

Обработка пользовательских запросов: в цикле бот принимает команды и вопросы пользователя, поддерживая следующие сценарии:

• Запрос информации о продукте (команда «расскажи про <продукт>») — бот выводит описание продукта из предопределенного словаря.

• Добавление продукта в историю покупок (команда «я покупаю <продукт>») — бот увеличивает счетчик покупок данного продукта.

• Запрос наиболее часто покупаемых продуктов — бот выводит топ-5 продуктов из истории покупок.

• Команда «выход» — завершение работы бота.

Обработка некорректных или непонятных запросов с выводом подсказки.

Сценарии взаимодействия

Пользователь запускает программу, видит приветственное сообщение.

Отвечает на вопрос о частоте покупок, выбирая вариант 1, 2 или 3.

Получает список предложенных вопросов и рекомендаций.

Вводит запросы, например:

• «расскажи про молоко» — бот выводит описание молока.

• «я покупаю хлеб» — бот добавляет хлеб в историю.

• «какие продукты я покупаю чаще всего?» — бот показывает статистику покупок.

При вводе «выход» программа завершает работу.

Используемые библиотеки

time — используется для создания пауз (time.sleep) между сообщениями, что улучшает восприятие диалога и делает общение более естественным.

Встроенные средства Python — класс, словари, функции ввода/вывода, сортировка.

В будущем планируется демонстрация интеграции основных функций: сканирование (эмуляция), озвучивание информации, выбор товара

Результаты. Доступ к результатам.

Резульаты проекта: сайт-схема и лэндинг, выполненный на языке HTML, чат-бот для демонстрации возможностей вывода информации, выполненный на языке Python, а также визуальная демонстрация, выполненная в интернет-программе Figma. Причём выполненные работы соответствуют описанному в проектной части данной работы.

[Лендинг](Лэндинг.HTML)

[Сайт-схема](Сайт%20схема/index.HTML)

[Демонстрационный чат-бот](ChatDemo.py)

[Демонстрационный дизайн](https://www.figma.com/proto/uw7z7BsPpYACOyT4uhFKkq/HelperAppDemo?node-id=2-2&p=f&t=bLbZeqXxgKq0wag4-1&scaling=scale-down&content-scaling=fixed&page-id=0%3A1&starting-point-node-id=2%3A2)

Листинг программ.

Сайт лэндинг.

HTML (Лэндинг.HTML):

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1" />

<title>Ваш помощник в магазине: Сканируй, слушай, выбирай</title>

<link rel="stylesheet" href="styleland.css">

</head>

<body>

<header>

<h1>Ваш помощник в магазине</h1>

<p>Сканируй, слушай, выбирай — легко и удобно!</p>

</header>

<div class="slider-container" aria-label="Основные функции приложения">

<div class="slide active" role="group" aria-roledescription="slide" aria-label="Сканирование текста и озвучивание вслух">

<img src="https://images.unsplash.com/photo-1585079542023-cd3e7a2d5f3f?auto=format&fit=crop&w=600&q=80" alt="Сканирование текста на упаковке" />

<h2>Сканирование текста и озвучивание вслух</h2>

<p>Наведите камеру на упаковку товара, и приложение прочитает всю информацию вслух. Это позволит быстро получить нужные данные без необходимости читать мелкий текст.</p>

</div>

<div class="slide" role="group" aria-roledescription="slide" aria-label="Распознавание продуктов по фотографии с высокой точностью">

<img src="https://images.unsplash.com/photo-1567306226416-28f0efdc88ce?auto=format&fit=crop&w=600&q=80" alt="Распознавание продукта по фото" />

<h2>Распознавание продуктов по фотографии с высокой точностью</h2>

<p>Сделайте фото товара, и приложение мгновенно определит его, предоставив подробную информацию и помогая сделать правильный выбор.</p>

</div>

<div class="slide" role="group" aria-roledescription="slide" aria-label="Возможности для слепых и слабовидящих">

<img src="https://images.unsplash.com/photo-1504384308090-c894fdcc538d?auto=format&fit=crop&w=600&q=80" alt="Доступность для слепых и слабовидящих" />

<h2>Возможности для слепых и слабовидящих</h2>

<p>Интуитивный интерфейс и голосовое управление разработаны специально для пользователей с нарушениями зрения, чтобы сделать покупки максимально комфортными и независимыми.</p>

</div>

<div class="slider-controls">

<button aria-label="Предыдущая функция" id="prevBtn">&#10094;</button>

<button aria-label="Следующая функция" id="nextBtn">&#10095;</button>

</div>

</div>

<section class="advantages" aria-label="Преимущества приложения">

<h2>Преимущества приложения</h2>

<ul>

<li>Экономит время при выборе и покупке товаров</li>

<li>Помогает ориентироваться в магазине без посторонней помощи</li>

<li>Делает покупки доступными для людей с нарушениями зрения</li>

<li>Интуитивно понятный и простой в использовании интерфейс</li>

<li>Голосовое управление для удобства и безопасности</li>

</ul>

</section>

<footer>

&copy; 2024 Ваш помощник в магазине. Все права защищены.

</footer>

<script>

(function() {

const slides = document.querySelectorAll('.slide');

const prevBtn = document.getElementById('prevBtn');

const nextBtn = document.getElementById('nextBtn');

let currentIndex = 0;

function showSlide(index) {

slides.forEach((slide, i) => {

slide.classList.toggle('active', i === index);

});

}

prevBtn.addEventListener('click', () => {

currentIndex = (currentIndex - 1 + slides.length) % slides.length;

showSlide(currentIndex);

});

nextBtn.addEventListener('click', () => {

currentIndex = (currentIndex + 1) % slides.length;

showSlide(currentIndex);

});

})();

</script>

</body>

</html>

CSS (styleland.css):

body {

font-family: "Segoe UI", Tahoma, Geneva, Verdana, sans-serif;

margin: 0;

background: #f9f9f9;

color: #333;

display: flex;

flex-direction: column;

align-items: center;

padding: 20px;

}

header {

text-align: center;

margin-bottom: 40px;

}

header h1 {

font-size: 2.5rem;

margin-bottom: 10px;

color: #2c3e50;

}

header p {

font-size: 1.2rem;

color: #555;

}

.slider-container {

position: relative;

max-width: 600px;

background: #fff;

border-radius: 10px;

box-shadow: 0 4px 12px rgba(0,0,0,0.1);

overflow: hidden;

margin-bottom: 50px;

}

.slide {

display: none;

padding: 20px;

text-align: center;

}

.slide.active {

display: block;

}

.slide img {

max-width: 100%;

height: 220px;

object-fit: contain;

margin-bottom: 20px;

border-radius: 8px;

background: #eee;

}

.slide h2 {

font-size: 1.5rem;

margin-bottom: 12px;

color: #34495e;

}

.slide p {

font-size: 1.1rem;

line-height: 1.5;

color: #555;

max-width: 520px;

margin: 0 auto;

}

.slider-controls {

position: absolute;

top: 50%;

width: 100%;

display: flex;

justify-content: space-between;

transform: translateY(-50%);

pointer-events: none;

}

.slider-controls button {

pointer-events: all;

background: rgba(44, 62, 80, 0.7);

border: none;

color: white;

font-size: 2rem;

padding: 10px 16px;

cursor: pointer;

border-radius: 50%;

user-select: none;

transition: background 0.3s;

}

.slider-controls button:hover {

background: rgba(44, 62, 80, 0.9);

}

section.advantages {

max-width: 700px;

background: #fff;

border-radius: 10px;

padding: 30px 40px;

box-shadow: 0 4px 12px rgba(0,0,0,0.1);

color: #2c3e50;

}

section.advantages h2 {

text-align: center;

margin-bottom: 25px;

font-size: 2rem;

}

section.advantages ul {

list-style: none;

padding-left: 0;

font-size: 1.1rem;

line-height: 1.6;

}

section.advantages ul li {

margin-bottom: 15px;

padding-left: 20px;

position: relative;

}

section.advantages ul li::before {

content: "✔";

position: absolute;

left: 0;

color: #27ae60;

font-weight: bold;

}

footer {

margin-top: 50px;

text-align: center;

font-size: 1rem;

color: #777;

}

@media (max-width: 640px) {

.slider-container {

max-width: 90vw;

}

section.advantages {

padding: 20px;

max-width: 90vw;

}

}

Сайт схема. chat.HTML

<!-- chat.html -->

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="UTF-8" />

<title>Чат о продуктах</title>

<link rel="stylesheet" href="stylegen2.css">

</head>

<body>

<h1>Чат о продуктах</h1>

<p>Здесь пользователь может общаться с помощником, задавать вопросы и получать рекомендации по продуктам.</p>

<p>Поддерживается ввод текста и голосовые сообщения.</p>

<p><a href="index.html">← Вернуться на главную</a></p>

</body>

</html>

index.HTML

<!-- index.html - Главная страница -->

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="UTF-8" />

<title>ВАШ ПОМОЩНИК В МАГАЗИНЕ</title>

<link rel="stylesheet" href="styleindex.css">

</head>

<body>

<div class="container">

<button id="scanner" onclick="location.href='scanner.html'">Сканер продукта / Распознавание текста</button>

<button id="settings" onclick="location.href='settings.html'">Настройки</button>

<button id="chat" onclick="location.href='chat.html'">Чат о продуктах</button>

<button id="search" onclick="location.href='search.html'">Поиск продуктов (текст/голос)</button>

</div>

</body>

</html>

scanner.HTML

<!-- scanner.html -->

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="UTF-8" />

<title>Сканер продукта</title>

<link rel="stylesheet" href="stylegen2.css">

</head>

<body>

<h1>Сканер продукта / Распознавание текста</h1>

<p>Здесь находится функционал камеры для сканирования штрихкодов и распознавания текста на упаковке продукта.</p>

<p>После сканирования приложение покажет информацию о продукте.</p>

<p><a href="index.html">← Вернуться на главную</a></p>

</body>

</html>

search.HTML

<!-- search.html -->

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="UTF-8" />

<title>Поиск продуктов</title>

<link rel="stylesheet" href="stylegen1.css">

</head>

<body>

<h1>Поиск продуктов в магазине</h1>

<p>Введите название продукта или используйте голосовой поиск.</p>

<input type="text" placeholder="Введите название продукта" />

<button>Голосовой поиск 🎤</button>

<p>Результаты поиска будут отображены здесь.</p>

<p><a href="index.html">← Вернуться на главную</a></p>

</body>

</html>

settings.HTML

<!-- settings.html -->

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="UTF-8" />

<title>Настройки</title>

<link rel="stylesheet" href="stylegen2.css">

</head>

<body>

<h1>Настройки</h1>

<p>Здесь можно управлять профилем пользователя, выбрать язык, настроить голосового помощника и другие параметры приложения.</p>

<p><a href="index.html">← Вернуться на главную</a></p>

</body>

</html>

stylegen1.css

body {

font-family: Arial, sans-serif;

padding: 20px;

background: #fafafa;

}

a {

text-decoration: none;

color: #007bff;

}

input[type="text"] {

padding: 8px;

width: 300px;

font-size: 16px;

margin-right: 10px;

}

button {

padding: 8px 12px;

font-size: 16px;

cursor: pointer;

border: none;

border-radius: 5px;

background-color: #007bff;

color: white;

}

button:hover {

background-color: #0056b3;

}

stylegen2.css

body {

font-family: Arial, sans-serif;

padding: 20px;

background: #fafafa;

}

a {

text-decoration: none;

color: #007bff;

}

styleindex.css

body, html {

margin: 0; padding: 0; height: 100%;

font-family: Arial, sans-serif;

display: flex;

justify-content: center;

align-items: center;

background: #f0f0f0;

}

.container {

position: relative;

width: 400px;

height: 400px;

background: white;

border: 2px solid #333;

border-radius: 10px;

box-shadow: 0 0 10px rgba(0,0,0,0.1);

}

button {

position: absolute;

padding: 15px 20px;

font-size: 16px;

cursor: pointer;

border: none;

border-radius: 5px;

background-color: #007bff;

color: white;

transition: background-color 0.3s;

}

button:hover {

background-color: #0056b3;

}

/\* Расположение кнопок \*/

#scanner {

top: 20px;

left: 50%;

transform: translateX(-50%);

width: 200px;

}

#settings {

top: 50%;

left: 20px;

transform: translateY(-50%);

width: 140px;

}

#chat {

top: 50%;

right: 20px;

transform: translateY(-50%);

width: 140px;

}

#search {

bottom: 20px;

left: 50%;

transform: translateX(-50%);

width: 200px;

}

Демонстрационный чат-бот для приложения.

import time

class ShopAssistantBot:

def \_\_init\_\_(self):

self.purchase\_history = {} # {product\_name: count}

self.purchase\_frequency = None # 'часто', 'редко', 'иногда'

self.products\_info = {

'молоко': 'Молоко — полезный напиток, богатый кальцием и витаминами.',

'хлеб': 'Хлеб — основной продукт питания, бывает разных сортов и вкусов.',

'яблоко': 'Яблоко — фрукт, богатый клетчаткой и витаминами.',

'сыр': 'Сыр — молочный продукт с высоким содержанием белка и кальция.',

'вода': 'Вода — необходима для поддержания водного баланса организма.'

}

def greet(self):

print("Здравствуйте! Я ваш помощник в магазине.")

time.sleep(1)

print("Давайте начнем с нескольких вопросов, чтобы лучше помочь вам.")

time.sleep(1)

def ask\_purchase\_frequency(self):

print("Как часто вы делаете покупки в магазине?")

print("1 - часто (несколько раз в неделю)")

print("2 - иногда (раз в неделю или две)")

print("3 - редко (реже раза в две недели)")

while True:

choice = input("Выберите 1, 2 или 3: ").strip()

if choice == '1':

self.purchase\_frequency = 'часто'

break

elif choice == '2':

self.purchase\_frequency = 'иногда'

break

elif choice == '3':

self.purchase\_frequency = 'редко'

break

else:

print("Пожалуйста, выберите 1, 2 или 3.")

def suggest\_questions(self):

print("\nИсходя из того, что вы делаете покупки", self.purchase\_frequency + ", могу предложить следующие вопросы:")

if self.purchase\_frequency == 'часто':

print("- Какие продукты вы покупаете чаще всего?")

print("- Хотите узнать о новых акциях и скидках?")

elif self.purchase\_frequency == 'иногда':

print("- Могу подсказать, что сейчас в наличии и по акции.")

print("- Хотите рекомендации по продуктам для дома?")

else: # редко

print("- Могу помочь составить список необходимых продуктов.")

print("- Хотите узнать о товарах с длительным сроком хранения?")

print("Или задайте мне вопрос о продукте (например, 'Расскажи про молоко').")

def handle\_user\_input(self):

while True:

user\_input = input("\nВаш вопрос или команда (введите 'выход' для завершения): ").strip().lower()

if user\_input == 'выход':

print("Спасибо, что воспользовались помощником. Хороших покупок!")

break

elif user\_input.startswith('расскажи про '):

product = user\_input.replace('расскажи про ', '').strip()

self.tell\_about\_product(product)

elif user\_input.startswith('я покупаю '):

product = user\_input.replace('я покупаю ', '').strip()

self.add\_to\_history(product)

elif user\_input == 'какие продукты я покупаю чаще всего?':

self.most\_frequent\_products()

else:

print("Извините, я не понял запрос. Попробуйте сформулировать по-другому или введите 'выход'.")

def tell\_about\_product(self, product):

info = self.products\_info.get(product)

if info:

print(f"Информация о продукте '{product}': {info}")

else:

print(f"Извините, у меня нет информации о продукте '{product}'.")

def add\_to\_history(self, product):

if product in self.purchase\_history:

self.purchase\_history[product] += 1

else:

self.purchase\_history[product] = 1

print(f"Добавлено в историю покупок: {product}")

def most\_frequent\_products(self):

if not self.purchase\_history:

print("Вы пока не добавляли продукты в историю покупок.")

return

sorted\_products = sorted(self.purchase\_history.items(), key=lambda x: x[1], reverse=True)

print("Ваши наиболее часто покупаемые продукты:")

for product, count in sorted\_products[:5]:

print(f"- {product}: {count} раз")

def main():

bot = ShopAssistantBot()

bot.greet()

bot.ask\_purchase\_frequency()

bot.suggest\_questions()

bot.handle\_user\_input()

if \_\_name\_\_ == "\_\_main\_\_":

main()

Заключение

В ходе практики была успешно разработана концепция и прототип мобильного приложения, направленного на помощь покупателям в магазине, с акцентом на удобство и доступность, в частности для людей с ограничениями по зрению. Выполнена комплексная работа, включающая анализ рынка и потребностей целевой аудитории, изучение современных технологий сканирования и преобразования текста в речь, а также определение функциональных требований к приложению.

В проектной части создана структура сайта на HTML, разработан визуальный прототип интерфейса в Figma, отражающий удобство и интуитивность использования, а также реализован демонстрационный чат-бот на языке Python, имитирующий взаимодействие пользователя с приложением. Все эти компоненты демонстрируют основные возможности будущего мобильного приложения — сканирование товаров, получение аудиоописания и выбор нужных позиций.

Результаты работы подтверждают актуальность и востребованность выбранной темы, а также показывают перспективы дальнейшей разработки полноценного приложения, способного значительно улучшить опыт покупок, сделать его более комфортным и инклюзивным. Полученный опыт и наработки создают прочную основу для дальнейшего развития проекта и внедрения инновационных решений в сфере помощи покупателям.

Список использованной литературы

1. Буянов М. П., Чернышёва А. С., Рудяк Ю. В./ «Использование искусственного интеллекта в мобильных приложениях для людей с ограниченными возможностями» / 2024 г.

2. Шрайнер Б. А., Гончаров И. О., Гордиенко П. Р./ «Технологии компьютерного зрения для помощи людям с ограниченными возможностями здоровья по зрению» / 2021 г.

3. Михаил Рубанов / «Про доступность iOS» / 2020 г.

4. А. М. Крапивняк / «Разработка социального мобильного приложения для людей с ограниченными возможностями» / 2021 г.

5. Н. Климов / «Доступная среда в современных приложениях. Методика тестирования программ с элементами доступной среды» / 2025 г.

Интернет источники:

1. Диаграмма из рисунка 1: <https://www.gazeta.ru/social/news/2024/12/04/24537338.shtml?updated>

2. Диаграмма из рисунка 2 - <https://igb.ru/docs/Доклад%20Нероев%20В.В.%20Белые%20ночи%202024%20Заболеваемость.pdf>