



МИНИСТЕРСТВО ПРОСВЕЩЕНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ
«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ
ПЕДАГОГИЧЕСКИЙ
УНИВЕРСИТЕТ ИМ. А. И. ГЕРЦЕНА»

**ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ
И ТЕХНОЛОГИЧЕСКОГО ОБРАЗОВАНИЯ**
Кафедра информационных технологий и электронного обучения

Основная профессиональная образовательная программа
Направление подготовки 09.03.01 Информатика и вычислительная техника
Направленность (профиль) «Технологии разработки программного
обеспечения»
форма обучения – очная

ТЕХНИЧЕСКОЕ ЗАДАНИЕ
к выпускной квалификационной работе на тему
«Применение машинного обучения для рекомендаций по питанию с
учётом физиологических потребностей человека»

Заказчик
Лотуга Данила Сергеевич

Разработчик:
Лотуга Данила Сергеевич

Санкт-Петербург
2024

СОДЕРЖАНИЕ

1	ОБЩИЕ ПОЛОЖЕНИЯ	4
1.1	Цель разработки	4
1.2	Основания для разработки (Ссылка на тему дипломной работы, то есть её титульный лист, а также список нормативных актов и ГОСТов, на основе которых регулируется работа)	4
1.3	Характеристика объекта разработки	5
2	НАЗНАЧЕНИЕ РАЗРАБОТКИ	6
3	ТРЕБОВАНИЯ К ФУНКЦИОНАЛЬНЫМ ХАРАКТЕРИСТИКАМ	7
3.1	Функции системы	7
3.2	Взаимодействие компонентов системы	7
4	ТРЕБОВАНИЯ К НАДЕЖНОСТИ	9
5	УСЛОВИЯ ЭКСПЛУАТАЦИИ	11
5.1	Описание предполагаемой среды	11
5.2	Условия для пользователей	12
6	ТРЕБОВАНИЯ К СОСТАВУ И ПАРАМЕТРАМ ТЕХНИЧЕСКИХ СРЕДСТВ	13
6.1	Серверная часть	13
6.2	Клиентская часть	13
7	ТРЕБОВАНИЯ К ИНТЕРФЕЙСУ	15
7.1	Пользовательский интерфейс по видам пользователей	15
7.2	Административная панель	15
8	ТРЕБОВАНИЯ К ПРОГРАММНОМУ ОБЕСПЕЧЕНИЮ	17
9	ТРЕБОВАНИЯ К ИНФОРМАЦИОННОМУ ОБЕСПЕЧЕНИЮ	19
9.1	Описание структуры базы данных	19
9.2	Виды и форматы входных и выходных данных	19
10	ПОРЯДОК КОНТРОЛЯ И ПРИЕМКИ	21

10.1	Описание тестовых сценариев	21
10.2	Условия приемки	21
11	СТАДИИ И ЭТАПЫ РАЗРАБОТКИ	23
12	ТРЕБОВАНИЯ К ДОКУМЕНТАЦИИ	25
12.1	Описание структуры и содержания руководства пользователя	25
12.2	Документация по API для взаимодействия компонентов	26
13	ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ	28
13.1	Ожидаемая производительность системы	28
13.2	Объем данных и требования к серверу	29

1 ОБЩИЕ ПОЛОЖЕНИЯ

1.1 Цель разработки

Целью настоящего исследования является создание интеллектуальной программной системы, специализирующейся на автоматизации процессов обработки визуальных данных и связанных с ними метаданных для обеспечения высокоточного анализа и последующего формирования рекомендаций в области диетологии. Данное программное обеспечение предполагает внедрение алгоритмов машинного обучения с целью углублённого анализа изображений и метаинформации, включая вычисление энергетической ценности, идентификацию состава ингредиентов и оценку распределения макронутриентов. Конечная цель состоит в предоставлении адаптивных решений для формирования персонализированных диетических стратегий и оптимизации рациона питания пользователей. Система акцентирует внимание на минимизации временных затрат и повышении аналитической точности за счёт применения передовых методов машинного зрения и структурного подхода к организации баз данных.

1.2 Основания для разработки (Ссылка на тему дипломной работы, то есть её титульный лист, а также список нормативных актов и ГОСТов, на основе которых регулируется работа)

Разработка системы базируется на утверждённой теме дипломной работы, формально закреплённой в документации учебного заведения. В основу подхода положены регламенты нормативно-правовых актов и государственные стандарты, касающиеся проектирования и разработки информационных систем. основополагающими стандартами выступают ГОСТы, регулирующие аспекты структурирования данных, архитектурные принципы построения баз данных и нормативы применения нейросетевых технологий в контексте аналитики данных.

- ГОСТ 34.601-90: «Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания». Этот стандарт описывает этапы создания автоматизи-

рованных систем, включая проектирование баз данных и структур данных.

- ГОСТ Р 70462.1-2022: «Информационные технологии. Искусственный интеллект. Нейронные сети. Часть 1. Общие положения». Стандарт описывает общие положения по использованию нейронных сетей, включая их применение в системах классификации и аналитики данных.
- ГОСТ Р 71657-2024: «Технологии искусственного интеллекта в образовании. Функциональная подсистема создания научных публикаций. Общие положения». Хотя этот стандарт ориентирован на образовательную сферу, он содержит рекомендации по использованию технологий искусственного интеллекта, включая нейросетевые, для обработки и анализа данных.

1.3 Характеристика объекта разработки

Характеристика объекта разработки отражает сложность и многокомпонентность разрабатываемой системы. Программное обеспечение включает в себя модульный архитектурный подход, который предполагает наличие функциональных блоков для ввода данных, их предобработки и очистки, а также глубокого анализа и интерпретации результатов посредством моделей, обученных на специализированных датасетах. Важным компонентом является реализация интуитивно понятного пользовательского интерфейса, предоставляющего доступ к ключевым функциям системы. Параллельно внедряется прикладной программный интерфейс (API), обеспечивающий интеграцию системы с внешними приложениями и сервисами. В системе реализуется надёжное централизованное хранилище данных, предназначенное для долговременного и масштабируемого хранения изображений блюд и их метаинформации, включая ингредиентный состав, калорийность, показатели энергетического баланса и дополнительные характеристики. Такая модульная структура обеспечивает гибкость и лёгкую адаптацию системы под разнообразные аналитические задачи и пользовательские сценарии.

2 НАЗНАЧЕНИЕ РАЗРАБОТКИ

Назначение разработки. Разрабатываемая система предназначена для широкого круга пользователей, включая специалистов в области диетологии, научных сотрудников, изучающих питание и пищевые продукты, а также для конечных пользователей, заинтересованных в получении персонализированных рекомендаций по рациону. Система также может быть полезна компаниям, занимающимся разработкой продуктов питания и диетических решений, для анализа их ассортимента с точки зрения питательной ценности и соответствия современным стандартам здорового питания.

Основное предназначение системы заключается в предоставлении инструментов для всестороннего анализа визуальной и текстовой информации о пищевых продуктах, их энергетической ценности, составе и возможном влиянии на здоровье. Это достигается за счёт автоматизации вычислительных процессов и использования моделей машинного обучения, что позволяет повысить скорость обработки больших массивов данных и точность результатов. Система способна минимизировать вероятность ошибок, обусловленных человеческим фактором, и обеспечить стандартизацию подхода к оценке продуктов питания.

Преимуществами данной разработки являются её высокая адаптивность, масштабируемость и способность интеграции в существующие экосистемы. Модульная архитектура позволяет эффективно использовать систему как автономно, так и в составе более широких информационных решений, что делает её незаменимым инструментом для профессиональной работы с данными о питании. Кроме того, интуитивно понятный интерфейс снижает порог входа для новых пользователей, обеспечивая удобный доступ ко всем функциональным возможностям системы.

3 ТРЕБОВАНИЯ К ФУНКЦИОНАЛЬНЫМ ХАРАКТЕРИСТИКАМ

3.1 Функции системы

Система должна обеспечить комплексный подход к обработке изображений и метаданных о пищевых продуктах, с использованием алгоритмов машинного обучения для проведения точной аналитики и формирования рекомендаций. Основными функциями системы являются сбор и предобработка данных, тренировка и валидация моделей машинного обучения, анализ полученных результатов, а также предоставление API для интеграции с внешними приложениями и сервисами. Система автоматически обрабатывает изображения, извлекая визуальные признаки для дальнейшей классификации и оценки энергетической ценности, состава ингредиентов и макронутриентов. Модели машинного обучения, обучаемые на большом объёме данных, позволяют более точно предсказать состав пищи и её питательные свойства. Обработка данных включает фильтрацию, очистку изображений и метаданных для обеспечения их качества и пригодности для дальнейшего анализа.

Кроме того, система должна интегрировать механизм формирования персонализированных диетических рекомендаций, исходя из анализа данных, полученных о калорийности продуктов и их влиянии на здоровье. Адаптивность системы обеспечивается за счёт её способности масштабироваться в зависимости от объёма обрабатываемых данных, включая возможность добавления новых категорий продуктов и специфических требований в систему рекомендаций.

3.2 Взаимодействие компонентов системы

Интерфейс пользователя предоставляет возможность взаимодействия с системой через простой и интуитивно понятный веб-интерфейс, в то время как API служит для автоматизированного доступа к функционалу системы, позволяя пользователям и сторонним приложениям получать данные о продуктах, их питательной ценности и рекомендованных решениях. Важной частью функционала является централизованное хранилище данных, обеспечивающее долго-

временное и масштабируемое хранение изображений пищи и метаинформации о ней.

Система должна обеспечивать полноценное взаимодействие с внешними сервисами, гарантируя корректную работу на всех этапах — от загрузки данных и обучения модели до предоставления результатов через API для дальнейшей аналитики или пользовательского интерфейса.

Каждый компонент системы выполняет свою уникальную роль. Модуль ввода данных отвечает за загрузку изображений и метаданных, реализуя процедуры их предварительной обработки, включая конвертацию форматов изображений и очистку текста в метаданных. Модуль анализа данных и машинного обучения анализирует введённые данные, выполняет обучение и валидацию моделей, используя современные нейросетевые технологии для достижения высокого качества аналитических прогнозов. Модуль представления и визуализации данных предоставляет интерфейс для пользователей, отображая результаты работы модели в удобной и понятной форме. Интеграция между всеми компонентами осуществляется через API, который управляет процессом взаимодействия и обеспечивает стабильность работы системы.

Поддержка интеграции системы с другими платформами и сервисами предполагает разработку гибкой архитектуры API, которая позволит подключать новые источники данных, такие как дополнительные датасеты для обучения или внешние системы для дальнейшей персонализации диетических рекомендаций.

4 ТРЕБОВАНИЯ К НАДЕЖНОСТИ

Для обеспечения устойчивости системы к сбоям и исключительных ситуаций требуется разработка механизма сохранения данных, который будет минимизировать риск потери информации при аварийных отключениях и других непредвиденных обстоятельствах. Система должна иметь встроенные функции резервного копирования данных, что позволяет восстанавливать сохранённую информацию в случае нештатных ситуаций, таких как сбой оборудования или ошибки в процессе обработки данных. Для этого предусмотрена реализация регулярных автоматических бекапов данных, включая изображения и метаданные, с возможностью их хранения в разных точках облачной инфраструктуры. При этом резервные копии должны быть доступными для быстрой и безопасной загрузки, что позволит оперативно восстанавливать целостность и актуальность данных.

Кроме того, механизмы обеспечения надежности системы должны предусматривать адаптивную нагрузочную балансировку, с целью предотвращения сбоев при пиковой нагрузке, и эффективное распределение задач между сервисами для оптимизации производительности в процессе обработки и анализа данных. Решения, связанные с многозадачностью, должны быть строго регулируемы с применением стратегии очередей и ограничений времени работы, чтобы предотвратить перегрузку системы при одновременном выполнении множества вычислительных операций.

Одним из важнейших элементов надежности является обеспечение корректной обработки ошибок. Система должна включать в себя продвинутую систему логирования, которая будет фиксировать информацию о любом сбое или аномалии в работе, с детализированным описанием характера ошибки, её потенциальных причин и возможных путей устранения. Каждое исключение должно быть правильно перехвачено, что позволит предотвратить полную остановку процессов или искажение данных. Логирование должно быть централизованным, чтобы предоставить возможность мониторинга работы системы и упрощения процесса диагностики неисправностей.

Особое внимание должно быть уделено аспектам безопасного взаимодействия между компонентами системы. Параллельно с обработкой ошибок необходимо предусмотреть механизмы для изоляции и блокировки операций, ко-

которые могут привести к нарушению целостности данных, несанкционированному доступу или их повреждению. В системе должны быть реализованы многоуровневые системы авторизации и проверки подлинности пользователей, что обеспечит высокий уровень безопасности при обработке персонализированных данных.

Дополнительно, системы оповещений и алертинга должны обеспечивать мгновенную реакцию на критические ошибки, автоматически информируя техническую команду о возникших проблемах. Это позволит значительно ускорить процесс восстановления функциональности системы и предотвратить длительные простои.

5 УСЛОВИЯ ЭКСПЛУАТАЦИИ

5.1 Описание предполагаемой среды

Предполагаемая среда разработки и эксплуатации системы основывается на использовании облачной инфраструктуры для обеспечения масштабируемости и высокой доступности. Серверная архитектура системы ориентирована на использование виртуализированных вычислительных ресурсов, что позволяет эффективно распределять нагрузку, гарантируя бесперебойную работу системы при различных объемах данных и интенсивности запросов. Ключевыми компонентами архитектуры являются модуль базы данных, обеспечивающий хранение изображений, метаданных и параметров питания, а также вычислительные узлы, на которых выполняется обучение моделей машинного обучения, аналитика и обработка данных. Для эффективной обработки изображений и повышения производительности системы предполагается использование платформы, поддерживающей технологии ускорения вычислений на GPU, таких как CUDA.

В качестве серверной базы выбран контейнеризованный подход, реализованный посредством Docker, что обеспечивает лёгкую интеграцию всех компонентов системы и их последующую переносимость между различными окружениями, а также упрощает деплоймент на облачных платформах. Для коммуникации между микросервисами используется RESTful API, что позволяет предоставлять стандартный интерфейс для взаимодействия внешних приложений с системой. API включает в себя эндпоинты для загрузки и обработки изображений, получения рекомендаций, а также выполнения аналитических операций.

Интерфейс пользователя ориентирован на простоту и удобство использования, минимизируя необходимость в технических знаниях. Для обеспечения интерактивности и удобного взаимодействия с пользователем используется веб-интерфейс, поддерживающий два основных режима — режим загрузки данных и режим получения аналитики. Модуль API взаимодействует с интерфейсом через асинхронные запросы, что позволяет обновлять состояние приложения в реальном времени, не требуя перезагрузки страницы.

5.2 Условия для пользователей

Система предназначена для удалённого доступа через веб-интерфейс, что позволяет пользователю взаимодействовать с ней из любой точки, имеющей интернет-соединение. Для работы с системой пользователю достаточно современного веб-браузера, поддерживающего стандарты HTTP/2, что обуславливает высокую производительность передачи данных и снижение задержек при взаимодействии с сервером. Ожидается, что браузеры последней версии (например, Chrome, Firefox, Edge) будут использоваться для работы с системой, что обеспечит стабильную работу интерфейса.

Для корректной работы пользовательского интерфейса и API необходима минимальная скорость интернет-соединения не менее 10 Мбит/с. При этом для эффективной обработки изображений и аналитики системы может потребоваться стабильное соединение, способное поддерживать постоянный поток данных, что важно для предотвращения потерь информации при интенсивной передаче.

Для пользователей, работающих с системой через API, потребуется наличие токенов авторизации, с целью обеспечения безопасности данных и возможности разграничения прав доступа. Авторизация будет реализована через систему, использующую JWT (JSON Web Tokens), что позволит эффективно управлять сессиями пользователей и ограничивать доступ к различным функциям системы в зависимости от уровня их прав.

Таким образом, минимальные требования к использованию системы ограничиваются доступом к интернету и поддержкой браузеров, что позволяет создать гибкую среду для работы с системой как для специалистов, так и для конечных пользователей, заинтересованных в персонализированных диетических рекомендациях.

6 ТРЕБОВАНИЯ К СОСТАВУ И ПАРАМЕТРАМ ТЕХНИЧЕСКИХ СРЕДСТВ

6.1 Серверная часть

Серверная часть системы должна быть спроектирована с учётом требований по масштабируемости и высокой доступности. Для обеспечения бесперебойной работы и возможности расширения нагрузки при увеличении объёмов обрабатываемых данных и запросов, сервер должен быть оснащён минимум четырьмя ядрами CPU и 16 ГБ оперативной памяти. Данные требования обеспечивают достаточную вычислительную мощность для обработки запросов от пользователей и выполнения аналитики, а также позволяют эффективно работать с изображениями и метаданными.

Одним из критически важных аспектов является поддержка ускорения вычислений с использованием технологий GPU. Наличие поддержки CUDA позволит значительно ускорить процессы обучения моделей машинного обучения и анализа изображений, что особенно важно при работе с большими объёмами данных. Помимо стандартных вычислительных мощностей, сервер должен обеспечивать безопасное и масштабируемое хранилище данных, предлагая возможность централизованного хранения и эффективного обращения к данным, включая изображения, текстовую информацию о продуктах и результаты анализа. Это требование будет реализовано через использование распределённых хранилищ, таких как PostgreSQL, с возможностью интеграции облачных решений для хранения изображений и выполнения вычислений.

Система также должна поддерживать резервное копирование данных с целью защиты от потери информации в случае сбоев. Наличие механизмов мониторинга, алертинга и автоматической перезапуска контейнеров (Docker) будет способствовать высокой надёжности серверной части.

6.2 Клиентская часть

Клиентская часть системы, предоставляющая пользовательский интерфейс, должна быть совместима с основными современными браузерами, такими как Google Chrome, Mozilla Firefox, а также Microsoft Edge, с обязательной

поддержкой последних версий этих браузеров. Интерфейс должен быть полностью адаптивным, что гарантирует корректную работу системы как на стационарных устройствах, так и на мобильных.

Минимальные технические требования для клиентской части заключаются в наличии устройства с современным браузером и стабильным интернет-соединением с минимальной пропускной способностью 10 Мбит/с. Важным аспектом является поддержка клиента для работы с веб-интерфейсом, включающим функции загрузки данных, получения аналитики и рекомендаций, а также общения с системой через API.

Для обеспечения эффективной и удобной работы с API необходимо наличие механизма авторизации, который будет реализован через систему токенов JWT. Входные данные и параметры, передаваемые между клиентом и сервером, должны соответствовать строгим стандартам безопасности для предотвращения несанкционированного доступа.

Таким образом, клиентская часть системы должна быть легко доступна пользователям через обычные браузеры, без необходимости установки дополнительных приложений, а её работа должна обеспечивать комфортное взаимодействие с сервером для получения результатов и аналитики в реальном времени.

7 ТРЕБОВАНИЯ К ИНТЕРФЕЙСУ

7.1 Пользовательский интерфейс по видам пользователей

Пользовательский интерфейс системы предназначен для обеспечения удобного и интуитивно понятного взаимодействия с функционалом приложения для различных категорий пользователей. Для конечных пользователей, как людей, так и специалистов в области диетологии, интерфейс будет оптимизирован для простоты ввода данных и получения рекомендаций. Главным экраном станет информационная панель, на которой будет отображаться основная информация о блюде: калорийность, состав макронутриентов, список ингредиентов и возможность загрузки изображений. Пользователь сможет получить рекомендации по улучшению рациона питания на основе поступивших данных о блюде. При этом интерфейс будет содержать элементы визуализации, такие как диаграммы, таблицы и графики, которые позволят быстро оценить питательную ценность блюд и отклонения от рекомендованных норм. Для удобства пользователей предусмотрены функциональные элементы, такие как форма для ввода данных (включая изображения) и поиск по базе данных, что позволит быстро находить и анализировать блюда, соответствующие заданным критериям.

Система будет поддерживать фильтрацию и сортировку блюд по ключевым характеристикам, таким как калорийность, состав, категория блюда или тип диеты. Кроме того, на пользовательском интерфейсе будут размещены дополнительные инструменты, направленные на обучение пользователя, такие как подсказки и FAQ, что снизит порог вхождения и ускорит освоение функционала системы. Визуальная составляющая интерфейса будет адаптивной и обеспечит корректную работу на различных устройствах, включая мобильные телефоны и планшеты, что увеличит доступность системы для широкой аудитории.

7.2 Административная панель

Административная панель системы будет предназначена для управления данными, мониторинга состояния модели и настройки параметров системы.

В этой панели администратор получит доступ к функционалу для управления данными о блюдах, ингредиентах, а также для мониторинга процесса обучения и обновления нейронных сетей. Панель будет включать разделы для добавления, изменения и удаления рецептов, анализа результатов обучения и просмотра статистики по использованию системы. Важно, что данный интерфейс обеспечит детальную настройку параметров алгоритмов, таких как типы предсказаний или уровни фильтрации данных, что позволит персонализировать работу системы для различных целей.

Административная панель также будет включать раздел для управления пользователями и правами доступа, что позволит назначать различные роли для конечных и экспертных пользователей. Эти возможности обеспечат систематизацию и контроль над операциями, выполняемыми в рамках работы с данными и обучением моделей. Для повышения уровня безопасности панель будет защищена системой аутентификации, используя сложные механизмы парольной защиты, а также двухфакторную аутентификацию для особых действий.

8 ТРЕБОВАНИЯ К ПРОГРАММНОМУ ОБЕСПЕЧЕНИЮ

Для разработки программного обеспечения будут использованы современные и эффективные технологии, соответствующие требованиям к функциональности и производительности системы. Серверная часть будет реализована на языке Python, который является основным языком для обработки данных, построения и обучения моделей машинного обучения, а также для интеграции с веб-фреймворками и базами данных. В качестве основного веб-фреймворка используется Django, который обеспечит надёжную основу для разработки, поддерживая такие важные функции, как работа с базами данных, создание RESTful API и безопасность приложения.

Для работы с данными изображений и выполнения задач машинного обучения будет использован фреймворк PyTorch. PyTorch предоставляет широкие возможности для работы с нейронными сетями и является одним из лидеров среди инструментов для реализации алгоритмов машинного обучения, обеспечивая гибкость и высокую производительность в процессе тренировки и предсказания моделей.

Для разработки интерфейса будет использован фреймворк React, который позволяет создавать динамичные и интерактивные пользовательские интерфейсы с использованием компонентов, что удобно для управления состоянием приложения и взаимодействия с сервером. Также для удобства управления состоянием клиента будет использована библиотека Redux Toolkit, которая позволит эффективно работать с состоянием данных и их синхронизацией с сервером.

База данных для хранения информации о блюдах, их составе и мета-информации будет реализована на PostgreSQL, что обеспечит её надёжность, масштабируемость и способность эффективно работать с большими объёмами структурированных данных. Взаимодействие с базой данных будет реализовано с использованием Django ORM, которая предоставляет абстракцию для работы с базой данных и позволяет значительно упростить написание запросов и манипуляций с данными.

Для контейнеризации системы будет использоваться Docker, который обеспечит независимость от конкретной операционной системы и поможет гарантировать консистентность среды разработки, тестирования и эксплуата-

ции приложения. Контейнеризация также позволяет упрощать развертывание и масштабирование системы.

Для интеграции различных компонентов и упрощения взаимодействия с внешними сервисами и клиентами, будет реализован API, использующий стандарты JSON для обмена данными. Интеграция с внешними приложениями и сервисами будет осуществляться через строго определённые интерфейсы, что обеспечит удобство расширяемости и модификации системы в будущем.

Таким образом, выбор программного обеспечения и технологий для реализации данного проекта ориентирован на достижение высокой производительности, надёжности, а также на возможность масштабирования и интеграции с внешними сервисами.

9 ТРЕБОВАНИЯ К ИНФОРМАЦИОННОМУ ОБЕСПЕЧЕНИЮ

9.1 Описание структуры базы данных

Структура базы данных разрабатываемой системы должна обеспечивать эффективное хранение и обработку данных о блюдах, их составе, питательной ценности и изображениях. Для этого будет использована реляционная структура данных на основе системы управления базами данных PostgreSQL. Система предполагает несколько взаимосвязанных таблиц, каждая из которых отвечает за хранение различных видов информации. Основной таблицей будет являться таблица, содержащая данные о блюде, в которой будут храниться такие атрибуты, как уникальный идентификатор блюда, название, описание, калорийность, содержание макронутриентов и указания на связанные с блюдом изображения. Таблица, содержащая метаинформацию о блюде, будет связана с основной таблицей через внешний ключ, обеспечивая целостность данных. Включение изображений будет осуществляться через ссылочные поля, которые будут указывать на места хранения файлов в системе или облачных сервисах, что снизит нагрузку на основную базу данных.

Для управления составом блюд предусмотрены отдельные таблицы, в которых будут храниться данные о каждом ингредиенте, его количестве и калорийности. Эта информация также будет связана с таблицей рецептов для поддержания консистентности данных. Кроме того, для улучшения анализа данных в системе будет поддерживаться индексация по ключевым параметрам, таким как калорийность, содержание белков, жиров и углеводов, а также создание пользовательских тегов для поиска по категориям блюд. Эта структура будет гибкой и расширяемой, что обеспечит возможность интеграции с дополнительными источниками данных в будущем.

9.2 Виды и форматы входных и выходных данных

Виды и форматы входных и выходных данных будут строго стандартизированы для всех компонентов системы. Для работы с API и взаимодействия с внешними клиентами будут использоваться форматы JSON, так как этот формат широко поддерживается различными языками программирования и явля-

ется удобным для передачи структурированных данных. Входные данные будут включать JSON-объекты с метаданной о блюде, например, название, описание, список ингредиентов, количество каждого компонента и ссылки на изображения. Изображения в качестве входных данных будут поступать в популярных форматах, таких как JPEG или PNG, что обеспечит совместимость с большинством источников данных.

Выходные данные, генерируемые системой после обработки запросов API и выполнения алгоритмов машинного обучения, будут также представлять собой JSON-объекты. Ответы на запросы о калорийности, составе блюда и рекомендациях будут содержать структурированные данные, включая общую информацию о питательной ценности, советы по улучшению рациона и дополнительные характеристики продуктов. В процессе работы с нейронными сетями выходные данные могут включать результаты классификации изображений, такие как идентифицированные ингредиенты и их калорийность, а также вероятность принадлежности к определённой категории (например, тип рациона). Эти данные будут передаваться клиенту для дальнейшего использования в приложении или аналитике. В случае использования предварительно обученных моделей нейронных сетей формат входных данных будет соответствовать предсказаниям модели, состоящим из массивов числовых значений, которые будут декодироваться в понятный пользователю вид.

10 ПОРЯДОК КОНТРОЛЯ И ПРИЕМКИ

10.1 Описание тестовых сценариев

Тестирование системы будет осуществляться по заранее разработанным тестовым сценариям, которые будут направлены на оценку качества работы системы и соответствия её функциональных возможностей заявленным требованиям. Тестирование будет охватывать все ключевые процессы, включая загрузку и обработку данных, работу нейронных сетей, а также корректность работы API. Протестированы будут все возможные пути взаимодействия пользователя с системой, начиная от ввода данных до получения рекомендаций по рациону питания и расчету калорийности блюд. Одним из важнейших аспектов тестирования будет проверка устойчивости системы к возможным ошибкам, включая некорректный формат данных, ошибочную загрузку изображений или сбои при запросах к серверу.

Каждый компонент системы будет проверяться по отдельности, начиная от интерфейса загрузки изображений до интеграции с нейросетевой моделью, что позволит выявить слабые места в обработке запросов и диагностике ошибок. Все данные, вводимые через пользовательский интерфейс, будут подвергаться валидации с целью определения их правильности и соответствия заданным форматам. Взаимодействие между компонентами системы будет проверяться на каждом этапе работы для обнаружения возможных расхождений или утечек данных. Будет уделено внимание проверке безопасности данных и защите пользовательской информации. Ожидается, что результаты анализа будут фиксироваться в отчётах о тестировании, с подробным описанием каждого теста, его целей и результатов.

10.2 Условия приемки

Условия приемки системы будут определяться на основе выполнения всех функциональных требований, отражённых в техническом задании. Основным критерием приемки будет являться способность системы полноценно выполнять все заявленные задачи без ошибок в обработке данных, с достаточной скоростью выполнения операций, соответствующей указанным временным

ограничениям. Также важным параметром станет стабильная работа интерфейса на разных типах устройств, обеспечивающая корректную работу всех элементов управления и визуализации данных. Принятие системы в эксплуатацию будет зависеть от выполнения требуемых показателей по безопасности и надёжности, включая наличие защищённых каналов передачи данных и механизмов аутентификации.

Период тестирования будет завершаться проведением конечных испытаний, на которых будут определяться возможные причины нестабильной работы системы или неточности в предсказаниях моделей. В случае обнаружения ошибок, система будет доработана и повторно протестирована с учётом исправлений. Принятие системы будет произведено только после окончательного выполнения всех функциональных тестов и успешного разрешения всех обнаруженных недостатков.

11 СТАДИИ И ЭТАПЫ РАЗРАБОТКИ

Стадии разработки данного программного обеспечения будут построены с учётом требуемой глубины исследования, аналитических задач и интеграции различных компонентов системы. Общая структура разработки будет разделена на несколько последовательных стадий, каждая из которых предполагает выполнение ряда специфичных этапов, обеспечивающих достижение целевых показателей и успешную реализацию функционала.

Первая стадия разработки будет связана с анализом требований и проектированием. На этом этапе будет проводиться сбор и анализ исходных данных, а также выявление ключевых функциональных требований к системе. Важным моментом будет являться детальная проработка архитектуры программного обеспечения, с выделением основных компонентов, таких как модуль загрузки данных, модель машинного обучения, интерфейсы пользователя и администратора, API для внешней интеграции. Будет детализирован процесс взаимодействия между системой и внешними сервисами, а также проработан функционал для будущей аналитики и рекомендаций. Данный этап потребует значительных усилий для проектирования структур баз данных и разработки схемы информационного обмена между компонентами.

На второй стадии, когда архитектурные решения будут утверждены, приступают к реализации самой системы. Этот этап будет включать в себя несколько ключевых этапов. Сначала будет разработана серверная часть системы, реализован функционал для сбора и обработки входных данных, обеспечена интеграция с нейросетевыми моделями и выполнены настройки для вычислений с учётом обучения и тестирования моделей. Следующим шагом станет создание клиентской части, где будет реализован интерфейс для загрузки изображений, визуализации результатов анализа и предоставления рекомендаций пользователю. В ходе этой стадии также будет производиться интеграция системы с внешними сервисами и базами данных, что требует настройки правильного взаимодействия через API.

Третья стадия будет посвящена тестированию и отладке разработанного ПО. Это критически важная стадия, которая включает несколько этапов: модульное тестирование отдельных компонентов системы, интеграционное тестирование для проверки работы всех частей вместе, а также функциональное те-

стирование в рамках работы с пользователями. Протестированы будут различные сценарии работы с системой, включая проверку стабильности обработки данных, корректность работы нейросетевой модели при разных входных данных, а также устойчивость API. По результатам тестирования будут выявляться ошибки или недостатки, которые должны быть устранены в процессе отладки.

Четвёртая стадия будет заключаться в развертывании и внедрении системы на целевую платформу. Этот процесс потребует подготовки серверной инфраструктуры, настройки резервного копирования и мониторинга, а также интеграции с другими существующими программными решениями. После настройки инфраструктуры будут проведены контрольные испытания в условиях, максимально приближенных к реальной эксплуатации, с целью удостовериться в том, что система функционирует как на локальных, так и на удалённых платформах. Система будет оптимизирована на основе результатов тестирования, учитывая количество одновременных запросов, скорость отклика и безопасность данных.

На пятой стадии, после успешного внедрения, будет проведён этап обучения пользователей и мониторинга эксплуатационной деятельности системы. Пользователи получают доступ к интерактивным и функциональным возможностям ПО через подготовленные инструктажи и курсы обучения. В этот момент специалисты будут собирать отзывы пользователей для возможных доработок и оптимизации работы системы. Модуль обучения также поможет понять, насколько интуитивно понятен интерфейс и какие улучшения необходимы для повышения качества работы с системой. Система будет находиться под регулярным мониторингом, для оценки работы нейросетевых моделей, анализа их точности и корректности, а также для сбора данных о возникновении потенциальных проблем.

Шестая стадия разработки будет связана с финишной доработкой и окончательной сдачей проекта. Этот этап будет включать завершение внедрения всех запросов пользователей и оптимизацию системы с учётом опыта её использования в реальной эксплуатационной среде. Завершающий этап работы будет заключаться в подготовке финальной документации и отчётов, которые будут включать полное описание функционала, архитектуры программного обеспечения и этапов разработки.

12 ТРЕБОВАНИЯ К ДОКУМЕНТАЦИИ

12.1 Описание структуры и содержания руководства пользователя

Руководство пользователя предназначено для всех категорий пользователей системы и должно предоставлять подробные инструкции, обеспечивающие максимально простое и интуитивно понятное использование всех функций системы. Оно будет разделено на несколько ключевых разделов, каждый из которых подробно опишет одну из основных функций, доступных в приложении. Структура руководства будет включать описание установки и настройки системы, интерфейса пользователя, а также рабочих сценариев и функциональных возможностей системы.

Первоначально руководство будет содержать раздел, посвященный установке системы, где будут детализированы требования к серверу и клиентскому оборудованию, а также пошаговая инструкция по развертыванию приложения в рабочем окружении. Следующим разделом будет описание процесса настройки и подключения системы, где будут даны инструкции по конфигурированию интерфейса, а также установке необходимых зависимостей, если таковые имеются.

Ключевая часть руководства будет посвящена описанию пользовательского интерфейса. В этом разделе будет предоставлено подробное описание всех доступных экранов и функциональных блоков, таких как страницы загрузки данных, анализа изображений, получения и интерпретации рекомендаций, а также работы с результатами анализов. Каждому элементу пользовательского интерфейса будет уделено внимание: кнопки, поля ввода данных, уведомления и системные сообщения. Пояснения будут сопровождаться скриншотами, что поможет пользователям наглядно ознакомиться с интерфейсными элементами.

Неотъемлемой частью руководства будет раздел с примерами работы системы. Это позволит пользователю лучше понять, как и для каких целей использовать те или иные функции. Пошаговые инструкции с примерами (например, по загрузке изображения и его анализу) помогут избежать ошибок на этапе работы. В случае возникновения проблем пользователи смогут найти возможные пути их решения в разделе частых вопросов и советов, включающих информа-

цию о возможных сбоях, ошибках, а также рекомендации по устранению неполадок.

В дополнение, в руководство будут включены практические рекомендации по оптимальному использованию системы, позволяющие повысить эффективность работы и добиться наилучших результатов. Руководство будет доступно как в текстовом, так и в виде мультимедийного контента, что обеспечит доступность для различных пользователей.

12.2 Документация по API для взаимодействия компонентов

Документация по API будет важным элементом для разработчиков, обеспечивая чёткое и полное описание всех доступных программных интерфейсов, необходимых для интеграции разрабатываемой системы с внешними сервисами, а также для внутреннего взаимодействия между компонентами системы. Она будет содержать не только описание всех ключевых конечных точек (endpoints) API, но также детализированные схемы данных, примеры запросов и ответов, а также инструкции по обработке ошибок и специфических состояний.

В документации будет указан список всех методов API, а также их предназначение и описание параметров запросов, типов данных, поддерживаемых входных и выходных форматов, требований к формату изображений и метаданных, а также описание стандартных и дополнительных фильтров или параметров для настройки запросов. Ожидается, что описание будет исчерпывающим и покрывать все варианты запросов, которые может выполнить внешнее приложение, а также указывать на возможные ошибки, которые могут быть возвращены сервером в ответ на запрос. Для каждой конечной точки API будет представлено подробное описание ожидаемых параметров, примеры входных и выходных данных, а также возможные ошибки и способы их исправления.

В документации также будет объяснено, как настроить аутентификацию и авторизацию для обращения к API. Это будет включать в себя описание способов получения токенов доступа, а также как внедрить защиту для запросов, чтобы исключить несанкционированный доступ к данным и сервисам. Разделы, посвящённые авторизации, включают описание форматов ключей API и способы работы с ними. Важной частью будет также инструкция по интеграции API в

мобильные или веб-приложения, что позволит разработчикам быстро подключить систему и начать обмениваться данными.

Документация будет снабжена множеством примеров использования различных конечных точек API, включая как базовые, так и более сложные сценарии использования. Каждый пример будет подробно объяснен с выделением важнейших аспектов каждого запроса, что поможет разработчикам минимизировать ошибки и ускорить процесс интеграции.

Для повышения качества использования, документация по API будет содержать раздел с типами возможных ошибок и способов их устранения. Это также будет включать решения для самых распространенных проблем, таких как некорректный формат изображения, ошибки в запросах, проблемы с подключением к серверу и другие аспекты, которые могут возникнуть при взаимодействии с системой.

13 ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

13.1 Ожидаемая производительность системы

Ожидаемая производительность системы определяется несколькими основными параметрами, связанными с обработкой данных, масштабируемостью и откликом пользовательского интерфейса. В первую очередь, производительность зависит от способности системы эффективно обрабатывать входные изображения с минимальными задержками при анализе. Ожидается, что система будет поддерживать обработку изображений с высокой разрешающей способностью в пределах от 256x256 до 1024x1024 пикселей, при этом время обработки одного изображения для получения первичной информации о составе и энергетической ценности не будет превышать нескольких секунд. Важно отметить, что данная производительность напрямую зависит от использованных моделей машинного обучения, которые будут постоянно оптимизироваться, что повысит скорость и точность работы системы.

Кроме того, система должна обеспечивать бесперебойную работу при большом объеме одновременных запросов. При расчете нагрузочного тестирования и возможных пиковых нагрузок было определено, что система должна быть способна обрабатывать запросы с минимальными задержками даже при одновременной работе 1000 и более пользователей. Для достижения данного уровня производительности система будет использовать методы параллельной обработки, эффективные алгоритмы сжатия данных, а также поддержку многоядерных процессоров в серверной архитектуре для распределения нагрузки.

Кроме того, в сфере обработки метаданных и сохранения получаемых данных потребуются интеграция с высокоскоростными системами базы данных, что обеспечит стабильность работы даже при больших объемах записей и обращений, сделанных пользователями. Ожидается, что в ходе эксплуатации системы при высоких темпах роста объема обрабатываемых данных с минимальными требованиями к серверному оборудованию производительность будет поддерживаться благодаря гибким алгоритмам кэширования и стратегии упрощения запросов.

Отдельное внимание будет уделено работе с внешними API и интеграциями, что требует быстрое взаимодействие между компонентами системы без

задержек, особенно для предоставления данных для рекомендаций по питанию, в том числе при комплексных расчетах, зависящих от множества факторов.

13.2 Объем данных и требования к серверу

Объем данных, обрабатываемых системой, напрямую связан с числом пользователей, а также с размерами входных изображений и метаданных. Вначале предполагается, что система будет работать с около 10 тыс. изображений и сопутствующих данных, при этом каждое изображение может занимать от 100 до 500 килобайт, в зависимости от уровня детализации и формата. При таких объемах потребуется решение для долговременного хранения данных, чтобы эффективно справляться с информацией, доступной пользователям и метаданными.

Ожидаемый объем данных на начальном этапе разработки будет около 10-15 гигабайт для всех изображений и метаданных, однако предполагается, что с увеличением пользовательской базы и расширением системы объем данных будет расти, что приведет к необходимости увеличения ресурсов хранения. При этом для эффективного хранения и обработки таких объемов данных необходимы серверы с многоуровневыми решениями для хранения, включая SSD-накопители для быстрой работы с изображениями и резервные системы на базе жестких дисков для долговременного хранения метаданных.

Серверная инфраструктура должна поддерживать расширяемость для дальнейшего увеличения как в плане пользователей, так и в плане обрабатываемых данных. Для этого серверы должны быть оснащены процессорами с несколькими ядрами для параллельной обработки данных и большими объемами оперативной памяти, что позволит ускорить обработку изображений и хранение временных данных. Для базы данных также потребуется оптимизация запросов и индексации, что обеспечит быстрое извлечение данных в реальном времени.

Дополнительно, серверная архитектура должна поддерживать гибкость в плане нагрузки, используя возможности виртуализации и кластеризации для масштабирования и управления трафиком в условиях пиковых нагрузок. Время отклика при массовых запросах не должно превышать нескольких миллисекунд, что потребует использования прокси-серверов для распределения трафика между различными вычислительными узлами.

Таким образом, требования к серверу включают наличие мощных процессоров для обработки больших объемов данных в реальном времени, наличие масштабируемых систем хранения данных и защиты от сбоев, а также обеспечение гибкости инфраструктуры для работы при высокой нагрузке и роста объема данных.