

СОДЕРЖАНИЕ

Обозначения.....	8
Термины и определения	9
Введение.....	11
1. История и эволюция оценки эргономики в авиации	12
1.1. Основные этапы развития эргономики в авиации.	12
1.2. Уменьшение численности экипажей: от многоместных экипажей до двух пилотов.....	12
1.3. Особенности современной эргономики в кабине пилота.	13
2. Современная нагрузка на пилотов.....	14
2.1. Факторы когнитивной и физической нагрузки на пилотов.	14
2.2. Анализ инцидентов, связанных с человеческим фактором.	14
2.3. Методы измерения нагрузки: применение PVT и NASA-TLX.	14
3. Теоретические основы эргономической оценки.....	16
3.1. Основы эргономики в авиации	16
3.2. Методы оценки эргономических характеристик	16
3.3. Теоретические основы тестов PVT и NASA-TLX	17
3.3.1. Тест психомоторной бдительности (PVT)	17
3.3.1.1. Принципы работы PVT	17
3.3.1.2. Алгоритм проведения теста PVT	17
3.3.1.3. Применение PVT.....	18
3.3.2. NASA Task Load Index (NASA-TLX).....	18
3.3.2.1. Структура NASA-TLX.....	18
3.3.2.2. Алгоритм проведения NASA-TLX.....	19
3.3.2.3. Применение NASA-TLX	19
3.3.3. Сравнение и сочетание методов PVT и NASA-TLX.....	19
4. Оценка эргономики в различных отраслях	21
4.1. Эргономика в медицинской сфере.....	21
4.1.1. Ключевые задачи медицинской эргономики	21
4.1.2. Задачи медицинской эргономики.....	22
4.1.3. Оценка когнитивной нагрузки.....	22
4.1.4. Примеры внедрения.....	22
4.2. IT и офисная среда.....	23
4.2.1. Физическое пространство	23

4.2.2.	Программное обеспечение	23
4.2.3.	Примеры внедрения.....	23
4.3.	Автомобильная промышленность	24
4.3.1.	Водительское место	24
4.3.2.	Оценка внимания и усталости	24
4.3.3.	Примеры внедрения.....	24
4.4.	Военная сфера.....	24
4.4.1.	Кабины летчиков.....	25
4.4.2.	Когнитивная нагрузка.....	25
4.4.3.	Примеры внедрения.....	25
5.	Формулировка задачи и цель разработки	26
5.1.	Обоснование разработки.....	26
5.2.	Цель и задачи разработки.....	26
5.2.1.	Цель разработки.....	26
5.2.2.	Задачи разработки	27
6.	Требования	28
7.	Выбор стека технологий	30
7.2	Анализ существующих решений.....	30
7.1.1.	Инструменты для проведения теста PVT:	30
7.1.2.	Инструменты для проведения NASA-TLX:.....	31
7.1.3.	Результаты анализа существующих решений:	34
7.2	Анализ подходящих технологий	34
8.	Определение технического задания	40
9.	Разработка программного обеспечения	42
9.2.	Архитектура приложения.....	42
9.3.1.	Слой пользовательского интерфейса (UI)	43
9.3.2.	Сервисный слой	44
9.3.3.	Слой данных.....	46
9.3.4.	Преимущества архитектуры.....	50
9.3.5.	Диаграмма архитектуры	51
10.	Структура проекта	52
11.	Потоки данных в системе.....	54
12.	Реализация ключевых функций.....	55
15.1.	Авторизация пользователя.....	55

12.4.1. Внешний вид кадра авторизация	56
15.2. Регистрация пользователя.....	57
12.4.1. Внешний вид кадра регистрации	60
12.4. Проведение PVT.....	64
12.3.1. Внешний вид кадра PVT тестирования.....	68
12.4. Расчёт NASA-TLX	71
12.4.1. Внешний вид кадра тестирования NASA-TLX	77
13. Тестирование программы.....	81
15.1. Сценарии тестирования.....	81
14. Организационно-экономическая часть.....	91
15.1. Введение	91
15.2. Расчет времени на проведение работ.....	91
15.3. Расчет заработной платы.....	96
15.4. Расходные материалы.....	97
15.5. Покупные комплектующие изделия	97
15.6. Электроэнергия на технологические цели	98
15.7. Отчисления во внебюджетные фонды.....	99
15.8. Отчисления на накладные расходы	99
15.9. Амортизационные отчисления	100
15.10. Смета затрат	100
15.11. Вывод.....	101
15. Охрана труда и окружающей среды	103
15.1. Описание выполняемых работ	103
15.2. Описание рабочего помещения.....	103
15.3. Описание рабочего места.....	104
15.4. Специальная оценка условий труда.....	106
15.4.1. Шум.....	106
15.4.2. Освещенность	107
15.4.3. Электробезопасность	110
15.4.4. Пожарная опасность.....	111
15.5. Итоговая оценка условий труда	114
Заключение	115
Список использованных источников и литературы	116

Обозначения

NASA-TLX — NASA Task Load Index.

PVT — Psychomotor Vigilance Task.

ПО — Программное обеспечение.

ORM — Object-Relational Mapping.

MVP — Minimum Viable Product.

SQLite — Лёгкая реляционная база данных.

SQLAlchemy — Библиотека для работы с базами данных.

PyInstaller — Инструмент для создания исполняемых файлов.

Pydantic — Библиотека для валидации данных в Python.

Pattern Matching — Механизм сопоставления с образцом

Термины и определения

NASA-TLX (NASA Task Load Index) - это широко используемый субъективный многомерный инструмент оценки рабочей нагрузки, разработанный в 1980-х годах в Исследовательском центре Эймса НАСА.

PVT - тестирование (Psychomotor Vigilance Task) — это метод оценки психомоторной бдительности и скорости реакции человека на визуальные стимулы.

Летный экипаж — персонал, управляющий летательным аппаратом в полете.

Стека технологий — это совокупность программных инструментов, библиотек, языков программирования и фреймворков, используемых для разработки программного обеспечения.

ПО (Программное обеспечение) — это набор программ и сопутствующей документации, предназначенный для выполнения определённых задач на компьютерах и других вычислительных устройствах.

ORM (Object-Relational Mapping) — метод программирования, позволяющий преобразовывать данные между несовместимыми типами систем с использованием объектно-ориентированного программирования. ORM упрощает взаимодействие с базами данных, превращая запросы в объекты языка программирования.

MVP (Minimum Viable Product) — минимально жизнеспособный продукт, представляющий собой первую рабочую версию приложения, содержащую только базовую функциональность. Основная цель MVP — быстрое тестирование идеи с минимальными затратами.

SQLite — лёгкая реляционная база данных, которая хранит все данные в одном файле. Её особенности включают простоту интеграции, автономность и кроссплатформенность.

SQLAlchemy — популярная библиотека для Python, предоставляющая инструменты для работы с базами данных через ORM и прямые SQL-запросы.

PyInstaller — инструмент для создания исполняемых файлов из Python-программ. Он поддерживает кроссплатформенность и упрощает распространение приложений.

Pydantic — библиотека Python для валидации данных и работы с типами. Использует аннотации типов Python для проверки и сериализации данных.

Pattern Matching — новая возможность Python 3.10, позволяющая использовать синтаксис, аналогичный конструкции switch/case, для обработки сложных данных и упрощения логики программирования.

Введение

Современная авиация предъявляет высокие требования к эргономике и эффективности человеко-машинного взаимодействия. Кабина пилота, как ключевой элемент авиационной техники, должна обеспечивать оптимальные условия для выполнения задач в различных режимах эксплуатации. От эргономических характеристик кабины зависят не только комфорт и здоровье пилотов, но и безопасность полетов в целом.

Эргономическая оценка кабины самолета требует комплексного подхода, включающего анализ когнитивной нагрузки, психофизиологических особенностей пилота, а также использование передовых технологий тестирования.

Методы тестирования психомоторной бдительности (PVT) и методика NASA-TLX доказали свою эффективность в оценке функционального состояния операторов. PVT позволяет объективно измерять когнитивные способности и реакцию оператора в реальном времени, а NASA-TLX — выявлять субъективное восприятие нагрузки по различным параметрам, таким как психическое напряжение, усилия и временные ограничения. Интеграция этих методов в единую систему оценки кабины самолета открывает новые возможности для повышения точности анализа и улучшения эргономических характеристик авиационной техники.

Целью данной работы является разработка системы эргономической оценки кабины самолета, объединяющей тесты PVT и методику NASA-TLX. Для достижения этой цели было создано программное обеспечение, обеспечивающее автоматизацию тестирования, сбор и обработку данных, а также визуализацию результатов. В процессе работы проведен обзор существующих методов эргономической оценки человеко-машинных интерфейсов, разработана методика анализа состояния оператора и представлено графическое описание созданного приложения.

1. История и эволюция оценки эргономики в авиации

1.1. Основные этапы развития эргономики в авиации.

Эргономика в авиации начала развиваться с первых лет авиационной индустрии, когда стало понятно, что комфорт и удобство пилота влияют на безопасность полетов. В 1930-х годах начались первые исследования по проектированию кабины пилота с учетом физиологических и когнитивных особенностей человека. Основной акцент делался на улучшение видимости приборов и эргономичное расположение органов управления.

После Второй мировой войны эргономика получила новый импульс благодаря развитию реактивной авиации. Быстрые самолеты требовали большей концентрации и реакции пилотов, что привело к созданию более интуитивных приборных панелей и удобных кресел. В 1960-х годах стали активно применяться принципы человеко-машинного взаимодействия, включая первые методы оценки когнитивной нагрузки.

В 1980–1990-х годах началось активное использование компьютерного моделирования для проектирования кабины пилота. Это позволило тестировать эргономические решения без необходимости строить физические макеты. С этого момента эргономика стала неотъемлемой частью проектирования всех современных самолетов.

1.2. Уменьшение численности экипажей: от многоместных экипажей до двух пилотов.

Первоначально управление самолетами требовало большой команды, включающей пилота, штурмана, радиста и инженера. Каждая из этих ролей была необходима для обеспечения надежности полета. Однако с развитием технологий численность экипажа постепенно уменьшалась.

В 1970-х годах внедрение автоматических систем навигации и управления двигателями позволило сократить экипаж до трех человек, исключив роль штурмана. К 1990-м годам с появлением цифровых систем управления и усовершенствованных автопилотов экипаж уменьшился до двух пилотов. Эти изменения повысили требования к когнитивным и физическим

возможностям пилотов, что сделало эргономику еще более важной частью проектирования кабины.

1.3. Особенности современной эргономики в кабине пилота.

Современные кабины самолетов проектируются с учетом следующих факторов:

- Интуитивность управления: Все органы управления расположены так, чтобы минимизировать время на их использование.
- Интеграция систем: Информация с приборов объединяется на мультифункциональных дисплеях, что облегчает восприятие данных.
- Когнитивная поддержка: Используются системы предупреждений и подсказок, снижающие нагрузку на пилотов в стрессовых ситуациях.
- Эргономичные кресла: Удобные сиденья с поддержкой поясницы и возможностью регулировки положения минимизируют физическую усталость во время длительных рейсов.

Эти элементы направлены на обеспечение безопасности и эффективности работы пилотов, особенно в условиях длительных перелетов и высоких нагрузок.

2. Современная нагрузка на пилотов

2.1. Факторы когнитивной и физической нагрузки на пилотов.

Работа пилотов сопряжена с высокой когнитивной и физической нагрузкой, особенно во время взлета, посадки и работы в сложных погодных условиях. Основные факторы включают:

- Когнитивная нагрузка: Обработка большого объема информации, необходимость быстрого принятия решений в условиях ограниченного времени.
- Физическая нагрузка: Усталость от длительного нахождения в одном положении, особенно на длительных рейсах.
- Психоэмоциональный стресс: Ответственность за жизнь пассажиров и экипажа.

Исследования показывают, что высокая когнитивная нагрузка может приводить к ошибкам, связанным с человеческим фактором, что делает использование эргономичных решений жизненно необходимым.

2.2. Анализ инцидентов, связанных с человеческим фактором.

Согласно отчету Международной организации гражданской авиации (ИКАО), до 70% авиационных происшествий вызваны ошибками пилотов. [1]. Наиболее распространенные причины включают:

- Усталость, связанная с длительным временем полета.
- Неправильное восприятие информации с приборов.
- Ошибки, вызванные перегрузкой информации.

Примером является инцидент 2009 года с рейсом Air France 447, где ошибки пилотов в условиях сложной погодной ситуации привели к катастрофе. Этот случай подчеркивает важность разработки инструментов для оценки и снижения когнитивной нагрузки.

2.3. Методы измерения нагрузки: применение PVT и NASA-TLX.

PVT (Psychomotor Vigilance Test) и NASA-TLX (Task Load Index) являются одними из ключевых инструментов для оценки нагрузки на пилотов,

но не единственными в этой области. Хотя эти методы играют важную роль в исследовании когнитивной и физической нагрузки, они чаще используются как часть более обширных систем анализа. Кроме того, их совокупное применение для оценки эргономики в авиации пока не является стандартной практикой и не всегда охватывает все аспекты взаимодействия человека и системы.

PVT: используется для измерения уровня бодрствования и времени реакции. Этот тест помогает оценить усталость пилотов и их способность к выполнению задач в сложных условиях.

NASA-TLX: позволяет определить субъективную рабочую нагрузку, включая умственную, физическую и временную нагрузки. Это помогает адаптировать рабочие условия и интерфейсы кабины к возможностям пилотов.

Эти методы применяются как в авиации, так и в других отраслях для анализа и оптимизации рабочих процессов. Их использование способствует снижению числа ошибок и повышению уровня безопасности полетов.

3. Теоретические основы эргономической оценки

3.1. Основы эргономики в авиации

Эргономика — это научная дисциплина, изучающая взаимодействие человека с элементами системы с целью оптимизации производительности и обеспечения комфорта, безопасности и эффективности труда. В авиации эргономика направлена на адаптацию рабочих мест, оборудования и процедур к возможностям и ограничениям человека, учитывая физические, когнитивные и организационные аспекты деятельности пилотов и экипажа.

Кабина пилота является сложной человеко-машинной системой, где эргономические аспекты напрямую влияют на безопасность и эффективность полетов. Неправильное расположение приборов, неудобные органы управления или избыточная когнитивная нагрузка могут привести к ошибкам пилотирования. Исследования показывают, что эргономические недостатки кабин самолетов являются факторами риска для безопасности полетов, подчеркивая необходимость тщательной эргономической проработки кабин разрабатываемых самолетов.

3.2. Методы оценки эргономических характеристик

Существует множество методов оценки эргономических характеристик в авиации, включая аналитические, экспериментальные и экспертные подходы. Аналитические методы основываются на математическом моделировании и прогнозировании взаимодействия человека с системой. Экспериментальные методы включают лабораторные и полевые исследования с участием операторов. Экспертные методы предполагают использование опросников, которые участники заполняют самостоятельно, фиксируя свои ощущения и данные о нагрузке. Это позволяет собрать первичную информацию, отражающую субъективное восприятие параметров, таких как усталость, стресс и когнитивная нагрузка. Затем эти данные передаются экспертам, которые проводят их детальный анализ и оценку. Такой подход сочетает в себе преимущества самофиксации, обеспечивающей непосредственный доступ к индивидуальному восприятию, и экспертной интерпретации, позволяющей

учитывать объективные аспекты и делать выводы о состоянии оператора и эргономических характеристиках системы. Комбинация этих подходов позволяет получить всестороннюю оценку эргономики авиационных систем.

Человеко-машинное взаимодействие в авиации характеризуется высокой сложностью и динамичностью. Пилоты должны быстро и точно обрабатывать большие объемы информации, принимая решения в условиях временного давления и стресса. Эргономические исследования человеческого фактора в современных технических системах подчеркивают необходимость согласования конструктивных особенностей машин с характеристиками деятельности человека-оператора для повышения надежности и безопасности эксплуатации.

3.3. Теоретические основы тестов PVT и NASA-TLX

3.3.1. Тест психомоторной бдительности (PVT)

Тест психомоторной бдительности (PVT) представляет собой объективный инструмент для измерения уровня бодрствования и когнитивной функции оператора. Этот тест разработан для оценки скорости реакции на визуальные стимулы, что делает его идеальным для определения снижения бдительности, вызванного усталостью, стрессом или другими факторами.

3.3.1.1. Принципы работы PVT

PVT основывается на простой задаче: пользователь должен как можно быстрее нажимать кнопку в ответ на появление визуального сигнала на экране. Записываются время реакции и количество пропущенных стимулов, что позволяет определить уровень усталости или снижение когнитивной функции. Основные показатели включают:

- Среднее время реакции.
- Частота ошибок (пропущенные сигналы).
- Вариабельность времени реакции.

3.3.1.2. Алгоритм проведения теста PVT

Подготовка: Устройство для тестирования (компьютер, планшет или специализированное оборудование) должно быть настроено на генерацию визуальных стимулов с нерегулярными интервалами.

Тестирование: Пользователь проходит серию испытаний, реагируя на визуальные стимулы. Время реакции фиксируется автоматически.

Анализ: Полученные данные обрабатываются для вычисления средних значений и выявления отклонений.

Интерпретация: Результаты анализируются с учетом индивидуальных характеристик пользователя и условий тестирования.

3.3.1.3. Применение PVT

PVT широко используется в авиации для оценки готовности пилотов к выполнению сложных задач, особенно в условиях длительных рейсов. Например, тестирование пилотов перед сменой позволяет определить уровень их усталости и принять соответствующие меры для повышения безопасности полетов. Также PVT активно применяется в медицине, транспорте и военной сфере.

3.3.2. NASA Task Load Index (NASA-TLX)

NASA-TLX (NASA Task Load Index) — это широко используемый субъективный многомерный инструмент оценки рабочей нагрузки, разработанный в 1980-х годах в Исследовательском центре Эймса НАСА. Он предназначен для оценки воспринимаемой оператором рабочей нагрузки при выполнении различных задач и используется в таких областях, как авиация, здравоохранение и других сложных социотехнических системах.

3.3.2.1. Структура NASA-TLX

NASA-TLX включает шесть шкал, каждая из которых оценивается пользователем по семибалльной шкале:

Ментальная нагрузка: степень умственной и перцептивной активности, необходимой для выполнения задачи (например, мышление, принятие решений, запоминание).

Физическая нагрузка: объем физической активности, требуемой для выполнения задачи (например, нажатие кнопок, управление устройствами).

Временная нагрузка: ощущаемое давление времени, связанное с выполнением задачи.

Усилие: количество усилий, затраченных для достижения уровня производительности в задаче.

Производительность: субъективная оценка успешности выполнения задачи и удовлетворенности результатом.

Уровень фрустрации: степень раздражения, стресса и неудовлетворенности, испытываемых во время выполнения задачи.

3.3.2.2. Алгоритм проведения NASA-TLX

Подготовка: Участнику объясняются шкалы и процедура оценки.

Оценка: После выполнения задачи участник оценивает каждый из шести параметров.

Взвешивание: Участник распределяет вес значимости между параметрами, что позволяет учитывать индивидуальные особенности задачи.

Анализ: Рассчитывается общий индекс нагрузки путем усреднения взвешенных значений.

3.3.2.3. Применение NASA-TLX

PVT широко используется в авиации для оценки готовности пилотов к выполнению сложных задач, особенно в условиях длительных рейсов. Например, тестирование пилотов перед сменой позволяет определить уровень их усталости и принять соответствующие меры для повышения безопасности полетов. Также PVT активно применяется в медицине, транспорте и военной сфере.

3.3.3. Сравнение и сочетание методов PVT и NASA-TLX

Оба метода, PVT и NASA-TLX, часто используются в сочетании для получения полного анализа состояния оператора:

- **PVT:** Обеспечивает объективную оценку когнитивной функции, измеряя скорость реакции.
- **NASA-TLX:** Позволяет понять субъективное восприятие нагрузки пользователем.

Их совместное использование особенно полезно в авиации, где необходимо учитывать как объективные показатели, так и субъективные ощущения пилотов. Например, пилот может демонстрировать нормальное время реакции по PVT, но сообщать о высоком уровне фрустрации и умственной нагрузки по NASA-TLX, что указывает на необходимость адаптации условий работы или интерфейса.

4. Оценка эргономики в различных отраслях

Эргономика давно вышла за рамки традиционных производственных процессов и сегодня охватывает множество отраслей, где человек взаимодействует с техникой, информацией или физической средой. Ее принципы применяются для повышения безопасности, производительности и удовлетворенности пользователей, адаптируя системы и процессы к человеческим возможностям и ограничениям.

В каждой отрасли эргономика сталкивается с уникальными задачами. Несмотря на различия, объединяющим элементом является стремление обеспечить гармоничное взаимодействие между человеком и системой, минимизируя риски и оптимизируя результаты.

В этом разделе мы подробно рассмотрим, как принципы эргономики реализуются в нескольких ключевых сферах, таких как медицина, IT, транспорт и военная индустрия. Для каждой из отраслей будет представлено влияние эргономики, решаемые задачи и достигнутые результаты.

4.1. Эргономика в медицинской сфере

Эргономика в медицине играет ключевую роль в обеспечении безопасности труда медперсонала и улучшении качества лечения пациентов. В последние годы этот аспект стал особенно актуальным, поскольку медицинская сфера сталкивается с новыми вызовами, такими как повышенная нагрузка на врачей и медсестер. Интересно, что многие решения в этой области разрабатываются на основе данных, полученных из реальных медицинских учреждений и научных исследований.

4.1.1. Ключевые задачи медицинской эргономики

Медицинская эргономика направлена на создание условий труда, которые минимизируют физическую и психическую нагрузку на медицинский персонал. В условиях высокой рабочей нагрузки современные медицинские учреждения активно применяют научные подходы, включая использование специализированного программного обеспечения для анализа эргономических параметров. Эти методы позволяют минимизировать физические и

когнитивные нагрузки, а также предотвратить профессиональные заболевания.

Особое внимание уделяется профилактике профессиональных заболеваний, таких как остеохондроз позвоночника, который часто встречается у медсестёр из-за физических перегрузок при перемещении пациентов и оборудования.

4.1.2. Задачи медицинской эргономики

Основные задачи медицинской эргономики включают:

- **Оптимизацию операционных пространств:** Применение эргономичных операционных столов с автоматической регулировкой высоты и угла наклона способствует снижению утомляемости хирургов.
- **Разработку удобных инструментов:** Антропометрическое проектирование хирургического оборудования снижает физическую нагрузку и повышает точность манипуляций.
- **Реорганизацию рабочих мест медсестер:** Удобное размещение оборудования и материалов уменьшает риск травм и ошибок.

4.1.3. Оценка когнитивной нагрузки

Методика NASA-TLX применяется для оценки когнитивной нагрузки медицинского персонала, особенно в ситуациях, требующих высокой концентрации внимания. Исследования показали, что внедрение эргономичных решений позволяет снизить уровень стресса на 20%, что ведет к повышению скорости принятия решений на 15% в условиях высокой интенсивности работы.

4.1.4. Примеры внедрения

Примером успешного применения является проектирование операционных в клиниках Германии. Использование эргономичных столов и оптимизация расположения оборудования позволили сократить длительность операций на 10–15%. Такие решения снижают вероятность профессионального выгорания у хирургов и повышают общее качество медицинской помощи.

4.2. IT и офисная среда

Эргономика в IT и офисной среде направлена на улучшение физического и когнитивного комфорта сотрудников. Современные решения включают как физические улучшения рабочего пространства, так и использование цифровых инструментов для оценки и оптимизации эргономических характеристик.

4.2.1. Физическое пространство

Регулируемые по высоте столы, эргономичные кресла и специализированное освещение стали стандартом в крупных IT-компаниях. Такие улучшения снижают риск профессиональных заболеваний, связанных с длительным пребыванием в одной позе. Например, сотрудники, использующие регулируемые столы, демонстрируют увеличение производительности труда на 12%.

4.2.2. Программное обеспечение

Эргономика программного обеспечения фокусируется на разработке удобных интерфейсов, минимизирующих когнитивную нагрузку. Системы анализа, такие как NASA-TLX, позволяют разработчикам оптимизировать цифровые продукты, обеспечивая удобство пользователей. Результаты исследований показывают, что интуитивные интерфейсы сокращают время выполнения задач на 30%.

Примером использования таких методов является применение программного обеспечения для отслеживания движений глаз, такого как iTracker. Этот инструмент позволяет анализировать поведение пользователя, выявлять сложные участки интерфейса и оптимизировать их для улучшения пользовательского опыта. Анализ с использованием iTracker может дополнить данные, полученные от опросников, таких как NASA-TLX, предоставляя разработчикам объективную информацию о нагрузке на пользователей при взаимодействии с системой.

4.2.3. Примеры внедрения

Компании Google и Microsoft внедряют эргономические рабочие пространства и инструменты для анализа когнитивной нагрузки, что позволяет

сократить количество ошибок в работе сотрудников и снизить уровень профессионального выгорания. Эти меры привели к сокращению числа больничных на 20%.

4.3. Автомобильная промышленность

Эргономика в автомобильной промышленности направлена на повышение комфорта и безопасности как для водителей, так и для пассажиров. Особое внимание уделяется проектированию водительского места, а также системам оценки и управления когнитивной нагрузкой водителей.

4.3.1. Водительское место

Внедрение эргономичных кресел с поддержкой поясницы и автоматической регулировкой положения позволяет снизить утомляемость водителей на длительных маршрутах. Современные интерфейсы управления помогают водителю быстрее адаптироваться к транспортному средству, что повышает уровень безопасности.

4.3.2. Оценка внимания и усталости

Системы предупреждения усталости водителя разрабатываются на основе тестов PVT. Эти системы мониторят внимание и состояние водителя в режиме реального времени, отправляя сигналы о необходимости отдыха. Применение таких решений, значительно, снижает риск ДТП.

4.3.3. Примеры внедрения

Автомобили Tesla и Volvo оснащены адаптивными креслами и интеллектуальными системами мониторинга усталости водителей. Это способствует снижению аварийности и увеличению общей удовлетворенности клиентов. Эргономические улучшения также положительно влияют на продажи и имидж производителей.

4.4. Военная сфера

Эргономика в военной сфере охватывает проектирование рабочих мест операторов, кабин пилотов и систем управления, обеспечивая эффективность действий в экстремальных условиях. Разработка военной техники включает

как физические, так и когнитивные аспекты взаимодействия человека с оборудованием.

4.4.1. Кабины летчиков

Современные кабины истребителей разрабатываются с учетом эргономических требований. Интуитивно понятное расположение органов управления, минимизация вибрации и улучшенная обзорность повышают точность и скорость действий пилотов в сложных условиях.

4.4.2. Когнитивная нагрузка

Применение NASA-TLX позволяет разрабатывать интерфейсы и оборудование, которые минимизируют стресс и когнитивные перегрузки операторов. Это особенно важно для обеспечения безопасности при выполнении миссий в условиях высокой напряженности.

4.4.3. Примеры внедрения

Примером является внедрение модернизированных кабин для военных истребителей, что позволило сократить время реакции пилотов на 15%. Такие улучшения способствуют успешному выполнению миссий и минимизации рисков для жизни экипажа.

5. Формулировка задачи и цель разработки

5.1. Обоснование разработки

Эргономика играет важнейшую роль в обеспечении безопасности и эффективности работы пилотов. Анализ аварийных ситуаций в авиации показывает, что до 70% инцидентов связаны с человеческим фактором, включая усталость и перегрузку. [1]. Это делает создание системы для эргономической оценки кабины самолета актуальной задачей.

Современные инструменты для оценки состояния операторов, такие как тесты PVT и NASA-TLX, применяются в различных отраслях, включая авиацию, медицину и военную сферу. Однако их интеграция в специализированное программное обеспечение для авиации остается ограниченной. Существующие решения, как правило, либо не адаптированы для условий авиационной среды, либо недостаточно удобны для использования в реальном времени. Это создает необходимость разработки новой системы, которая:

- Учитывает специфику работы пилотов.
- Интегрирует проверенные методики PVT и NASA-TLX.
- Позволяет объективно оценивать эргономические характеристики кабины самолета.

Разработка такой системы способствует повышению уровня безопасности полетов и снижению риска ошибок, связанных с человеческим фактором.

5.2. Цель и задачи разработки

5.2.1. Цель разработки

Создание минимально жизнеспособного продукта (MVP) программного обеспечения для эргономической оценки кабины самолета, использующего методики PVT и NASA-TLX для анализа когнитивной и физической нагрузки на пилотов.

5.2.2. Задачи разработки

1. Сформировать требования к системе:

- Сбор информации о существующих решениях и методиках.
- Определение функциональных и нефункциональных требований.

2. Выбор стека технологий для реализации ПО:

- Анализ доступных библиотек и инструментов.
- Обоснование выбора наиболее подходящей стека технологий.

3. Разработать ключевые модули системы:

- Модуль для проведения теста PVT с реализацией визуализации стимулов и регистрации времени реакции.
- Модуль NASA-TLX для ввода данных, расчета индекса нагрузки и визуализации.

4. Создать графический пользовательский интерфейс:

- Проектирование интуитивно понятного GUI для тестирования и анализа данных.
- Интеграция визуализации результатов с основными функциями системы.

5. Обеспечить сохранение и управление данными:

- Реализация механизма хранения результатов тестов.
- Добавление функций экспорта данных в удобный формат.

6. Провести тестирование и валидацию системы:

- Проверка работоспособности всех модулей.
- Проверка корректности отчетов

6. Требования

Система предназначена для проведения автономного анализа эргономических характеристик кабин самолета с использованием тестов PVT (Psychomotor Vigilance Test) и NASA-TLX (Task Load Index). Она должна работать в условиях ограниченного доступа к сети Интернет, чтобы обеспечить безопасность данных, так как летные данные представляют собой ценную информацию. Использование системы предполагает её установку на локальные машины, что исключает необходимость облачных сервисов и минимизирует риски утечки информации.

Для взаимодействия с системой предполагается использование стандартных устройств ввода информации (клавиатура и мышь), а также возможность просмотра и сохранения данных через интуитивно понятный графический интерфейс. Автономность системы является ключевым требованием, чтобы избежать зависимостей от внешних сервисов и инфраструктуры.

01_VC	Система должна функционировать в полном автономном режиме без необходимости подключения к сети Интернет.
02_VC	Система должна начинать процесс сбора данных и проведения тестирования в момент запуска через пользовательский интерфейс.
03_VC	Система должна работать с такими устройствами ввода информации, как клавиатура и мышь
04_VC	Система должна проводить анализ данных в соответствии с методиками тестов PVT и NASA-TLX, реализуя predetermined алгоритмы обработки.
05_VC	Система должна блокировать возможность начала тестирования, если пользователь не прошел процесс

	регистрации или авторизации через предоставленный интерфейс.
06_VC	Система должна предоставлять пользователю визуальное представление результатов тестов в графическом интерфейсе.
07_VC	Система должна сохранять результаты каждого теста для последующего анализа и экспорта.
08_VC	Система должна использовать компоненты с открытым исходным кодом.
09_VC	Система должна минимизировать требования к ресурсам, чтобы работать на устройствах с ограниченными вычислительными возможностями.

7. Выбор стека технологий

7.2 Анализ существующих решений

На текущий момент не существует интегрированных программных продуктов, специально предназначенных для оценки эргономики, которые одновременно включают тесты PVT (Psychomotor Vigilance Task) и NASA-TLX (NASA Task Load Index). Однако, отдельные инструменты для проведения каждого из этих тестов доступны и широко используются в различных исследованиях и практиках.

7.1.1. Инструменты для проведения теста PVT:

PVT-192 — портативное устройство, предназначенное для измерения времени реакции в лабораторных и полевых условиях. Оно используется для оценки уровня бдительности и выявления когнитивных нарушений, связанных с усталостью. Принцип действия PVT-192 основан на измерении времени реакции пользователя на визуальные стимулы, что позволяет объективно оценить психомоторную скорость и внимание. Устройство оснащено внутренним аккумулятором и может работать автономно, что делает его удобным для длительных исследований в различных условиях.



Рисунок 1 – Устройство PVT-192

PVT Self-Test — мобильное приложение, доступное для различных платформ, которое предоставляет возможность самостоятельно проводить тест PVT. Оно используется в исследованиях сна и бдительности, а также для индивидуального мониторинга уровня усталости. Принцип действия приложения аналогичен устройству PVT-192: пользователь реагирует на визуальные стимулы, а приложение фиксирует время реакции, позволяя оценить когнитивное состояние.

Оба инструмента широко применяются в научных исследованиях, связанных с изучением влияния усталости на когнитивные функции, а также в практических областях, требующих мониторинга уровня бдительности операторов, таких как авиация, транспорт и медицина

7.1.2. Инструменты для проведения NASA-TLX:

Первоначальная версия инструмента, известная как NASA-TLX Paper-and-Pencil Version, представляет собой бумажный опросник. Участник после выполнения задания оценивает шесть аспектов нагрузки по шкале от 0 до 100. Затем происходит взвешивание значимости этих аспектов путём парных сравнений, что позволяет учесть особенности восприятия конкретной задачи. Итоговый индекс нагрузки рассчитывается с учётом весовых коэффициентов, предоставляя точную количественную оценку рабочей нагрузки. Эта версия проста в использовании и до сих пор применяется в ситуациях, где цифровые инструменты недоступны или нецелесообразны.

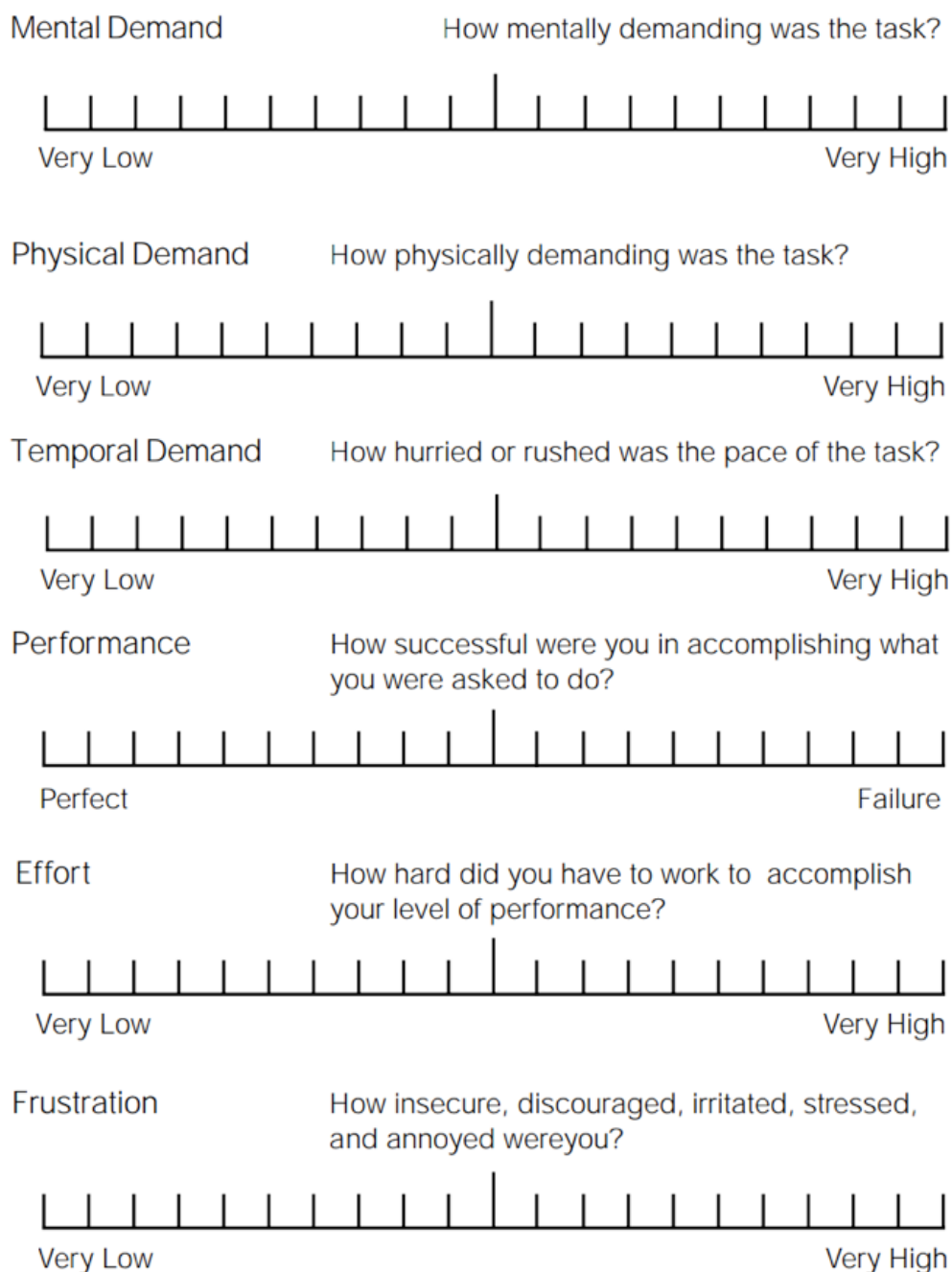


Рисунок 2 – Бумажной версии NASA-TLX

С развитием технологий NASA представила цифровую версию инструмента — NASA-TLX App. Это приложение предоставляет все возможности бумажной версии, но дополнительно упрощает процесс сбора и анализа данных. Удобный интерфейс позволяет участникам быстро заполнять оценки, а исследователи получают мгновенный доступ к обработанным результатам. Возможности приложения делают его незаменимым инструментом для современных исследований, где требуется оперативный

анализ данных. NASA-TLX App активно используется в тестировании интерфейсов, авиационных систем и медицинских приборов.

The image shows two side-by-side screenshots of the NASA-TLX App interface. The left screen is titled 'Rate task experience' and contains fields for 'NAME' (with a placeholder 'Please enter your name'), 'TASK' (with instructions to read the user guide), 'DATE' (showing '2018-09-12 20:56'), and two horizontal scales for 'Mental Demand' and 'Physical Demand'. The 'Mental Demand' scale has a red marker at the 8th position (out of 10), and the 'Physical Demand' scale has a red marker at the 3rd position. The right screen is titled 'Rate workload sources' and shows 'Question 15/15'. It has two options: 'PHYSICAL DEMAND' and 'FRUSTRATION', each with a descriptive text and a scale. Below these is a checkered flag icon and a 'FINISH' button.

Рисунок 3 – Электронная версия NASA-TLX (приложение от компании NASA)

Применение NASA-TLX выходит далеко за пределы лабораторий. В авиации инструмент позволяет анализировать нагрузку на пилотов при выполнении сложных манёвров или в стрессовых условиях, таких как посадка при плохой видимости. В медицинской сфере он помогает оценивать рабочую нагрузку врачей и медсестёр в интенсивных условиях, например, в операционных или отделениях интенсивной терапии. В области разработки пользовательских интерфейсов NASA-TLX используется для оптимизации дизайна, чтобы минимизировать когнитивную нагрузку пользователей.

Интересно, что оба формата инструмента, бумажный и цифровой, имеют свои преимущества. Бумажная версия остаётся востребованной благодаря своей доступности и независимости от технической инфраструктуры, что особенно важно в полевых условиях или при ограниченных ресурсах.

Цифровая версия, напротив, идеально подходит для масштабных исследований, где требуется автоматизация сбора данных и их анализ в реальном времени.

7.1.3. Результаты анализа существующих решений:

Несмотря на наличие отдельных инструментов для проведения тестов PVT и NASA-TLX, а также комплексных систем для общей эргономической оценки, на рынке отсутствует интегрированное решение, объединяющее обе методики в едином программном продукте. Это подчеркивает актуальность разработки подобного приложения, которое могло бы предоставить возможность проведения оценки эргономики с использованием нескольких методик в едином интерфейсе.

7.2 Анализ подходящих технологий

При разработке программного обеспечения важным этапом является выбор библиотек и инструментов, которые обеспечат необходимую функциональность, стабильность и удобство использования. Для проекта, ориентированного на оценку эргономики кабины самолета, особое внимание уделено таким критериям, как открытый исходный код, кроссплатформенность, поддержка автономного режима работы и переносимость данных между устройствами.

Для реализации проекта выбран язык программирования Python версии 3.10. Python предоставляет мощный и гибкий инструмент для разработки благодаря своей читаемости, богатой экосистеме библиотек и поддержке кроссплатформенности.

Версия 3.10 была выбрана из-за её стабильности и набора возможностей, таких как улучшенная поддержка типов и сопоставление с образцом (pattern matching), которые облегчают разработку сложных приложений.

Python также широко используется в научных исследованиях благодаря своей простоте и обширному набору библиотек, таких как NumPy, SciPy и Matplotlib. Эти библиотеки предоставляют мощные инструменты для

математического и статистического анализа, что делает Python подходящим выбором для создания приложения, ориентированного на эргономический анализ. Его кроссплатформенность гарантирует, что разработанное приложение будет работать одинаково эффективно на Windows, macOS и Linux.

Графический интерфейс приложения построен с использованием библиотеки Tkinter. Это стандартная библиотека Python, которая имеет открытый исходный код и является частью стандартной поставки языка. Tkinter обеспечивает простоту разработки и интеграции, что делает её подходящей для минимально жизнеспособного продукта (MVP). В отличие от альтернатив, таких как PyQt или Kivy, она не требует сложной настройки и лицензирования. Например, PyQt обладает расширенным функционалом, но его использование ограничивается лицензией GPL, что делает его менее подходящим для опенсорсных проектов. Kivy, хотя и является кроссплатформенной библиотекой, больше ориентирован на мобильные приложения, что ограничивает её применение для настольных решений.

Таблица 7.2.1 Характеристики библиотек для пользовательских интерфейсов

Библиотека	Открытый исходный код	Кроссплатформенность	Простота интеграции	Лицензия
Tkinter	Да	Да	Высокая	MIT
PyQt	Да	Да	Средняя	GPL
Kivy	Да	Да	Средняя	MIT

Для хранения данных была выбрана база данных SQLite. Её основные преимущества — лёгкость, кроссплатформенность и переносимость. В отличие от серверных решений, таких как PostgreSQL или MySQL, SQLite не требует настройки серверного окружения. Все данные хранятся в одном файле, что упрощает перенос приложения между устройствами. Этот выбор особенно актуален для автономной работы приложения, когда необходимо обеспечить доступ к данным без подключения к серверу.

Таблица 7.2.2 Характеристики баз данных

База данных	Открытый исходный код	Кроссплатформенность	Переносимость данных	Простота использования
SQLite	Да	Да	Высокая	Высокая
PostgreSQL	Да	Да	Низкая	Средняя
MySQL	Да	Да	Низкая	Средняя

Для создания исполняемого файла выбрана библиотека PyInstaller. Она позволяет упаковать Python-приложение в один файл, что значительно упрощает его использование и распространение. PyInstaller поддерживает работу на Windows, macOS и Linux, что делает её универсальным инструментом для сборки. В отличие от cx_Freeze и py2exe, PyInstaller требует меньше усилий для настройки и предоставляет более широкий спектр возможностей.

Таблица 7.2.3 Характеристики библиотек для создания исполняемого файла

Инструмент	Открытый исходный код	Кроссплатформенность	Простота использования
PyInstaller	Да	Да	Высокая
cx_Freeze	Да	Да	Средняя
py2exe	Да	Нет	Средняя

Для взаимодействия с базой данных используется SQLAlchemy. Эта библиотека сочетает в себе гибкость и мощность, позволяя работать как на уровне ORM, так и с чистыми SQL-запросами. Её конкурентами являются Django ORM и Peewee, однако Django ORM требует использование всего фреймворка Django, что избыточно для данного проекта. Peewee, хотя и является легковесным, уступает SQLAlchemy в функциональности и поддержке.

Таблица 7.2.4 Характеристики библиотек для взаимодействия с базой данных

ORM	Открытый исходный код	Гибкость	Интеграция с SQLite
SQLAlchemy	Да	Высокая	Да
Django ORM	Да	Средняя	Да
Peewee	Да	Низкая	Да

Для работы с данными внутри приложения используется Pydantic. Эта библиотека обеспечивает валидность данных и их строгое соответствие типам Python. Она выбрана благодаря простоте использования, скорости и поддержке современных подходов к работе с данными. Альтернативы, такие как Cerberus и Marshmallow, предоставляют схожие функции, но уступают Pydantic в производительности и удобстве.

Таблица 7.2.5 Характеристики библиотек для взаимодействия с базой данных

Библиотека	Открытый исходный код	Простота использования	Производительность
Pydantic	Да	Высокая	Высокая
Cerberus	Да	Средняя	Средняя
Marshmallow	Да	Средняя	Средняя

Для обработки данных, их анализа и визуализации в проекте используются библиотеки NumPy, Pandas и Matplotlib. Эти библиотеки обеспечивают удобство работы с большими объёмами данных, высокую производительность вычислений и широкие возможности для представления информации. Для выбора именно этих библиотек проведено сравнение с их аналогами.

NumPy — ключевой инструмент для работы с многомерными массивами и матрицами. Его производительность значительно превосходит стандартные возможности Python за счёт использования низкоуровневых оптимизаций. Конкурентами **NumPy** являются **SciPy** и **TensorFlow**, которые предоставляют схожий функционал, но ориентированы на более узкие области

применения. SciPy используется для научных расчётов, а TensorFlow — для машинного обучения, что делает NumPy наиболее универсальным выбором.

Таблица 7.2.6 Характеристики библиотек для вычислений и визуализации

Библиотека	Открытый исходный код	Простота использования	Производительность	Специализация
NumPy	Да	Высокая	Высокая	Общие вычисления
SciPy	Да	Средняя	Высокая	Научные расчёты
TensorFlow	Да	Низкая	Высокая	Машинное обучение

Pandas — лидер в обработке табличных данных и временных рядов. Конкурентами являются **Dask** и **Vaex**, которые обеспечивают параллельную обработку больших данных. Однако Pandas остаётся лучшим выбором для небольших и средних проектов из-за простоты использования и доступности инструментов для работы с данными.

Таблица 7.2.7 Характеристики библиотек для обработки табличных данных

Библиотека	Открытый исходный код	Простота использования	Поддержка больших данных
Pandas	Да	Высокая	Средняя
Dask	Да	Средняя	Высокая
Vaex	Да	Средняя	Высокая

Matplotlib — стандарт для создания графиков и визуализации данных. Конкуренты, такие как **Seaborn** и **Plotly**, предлагают более современные интерфейсы и дополнительные функции. Однако Matplotlib обеспечивает высокий уровень контроля над элементами графиков и полностью поддерживает работу в автономном режиме, что делает её оптимальной для проектов без подключения к сети.

Таблица 7.2.8 Характеристики библиотек для обработки табличных данных

Библиотека	Открытый исходный код	Простота использования	Гибкость настройки	Автономность
Matplotlib	Да	Высокая	Высокая	Да
Seaborn	Да	Высокая	Средняя	Да
Plotly	Да	Средняя	Высокая	Нет

Выбор NumPy, Pandas и Matplotlib продиктован их универсальностью, производительностью и открытым исходным кодом. Эти библиотеки идеально подходят для обработки и анализа данных в рамках автономного приложения, обеспечивая необходимые инструменты для реализации поставленных задач.

8. Определение технического задания

Разработка программного обеспечения для оценки эргономики кабины самолета начинается с анализа исходных данных и условий, в которых оно будет использоваться. Главной целью является создание инструмента, который способен проводить тестирование операторов с применением методик PVT и NASA-TLX для оценки когнитивной нагрузки и уровня бдительности. Важно, чтобы это программное обеспечение соответствовало специфическим требованиям, включая автономность, простоту использования и способность работать в кроссплатформенном режиме.

Программное обеспечение должно быть полностью автономным, исключая необходимость подключения к сети во время работы. Это обусловлено возможностью эксплуатации в условиях ограниченного доступа к сети. Также важно, чтобы система поддерживала корректное функционирование на наиболее популярных операционных системах, таких как Windows, Linux, что делает её более универсальной. Среда эксплуатации предполагает использование персональных компьютеров или ноутбуков с минимальными техническими характеристиками, такими как процессор с тактовой частотой 2 GHz, оперативная память объёмом 4 GB и свободное место на диске в 200 MB.

Основными пользователями программного обеспечения станут пилоты, которые проходят тестирование для определения их когнитивной нагрузки, а также исследователи и инженеры, занимающиеся анализом полученных данных. Для этих категорий важно, чтобы приложение было интуитивно понятным, легко настраиваемым и обеспечивало визуализацию результатов в удобной форме. Кроме того, оно должно предоставлять возможность сохранения результатов в локальной базе данных для последующего анализа и экспортировать их в формате CSV или JSON, что позволит интегрировать данные с другими системами или включать их в отчёты.

Программное обеспечение должно учитывать ряд технических ограничений. Например, оно должно работать в среде с ограниченным доступом к внешним ресурсам и быть оптимизированным для минимальных

системных требований. Функциональные возможности включают проведение тестов PVT и NASA-TLX, расчёт ключевых метрик, таких как средняя скорость реакции и уровень рабочей нагрузки, а также визуализацию данных в виде графиков и таблиц.

Простота установки и настройки является ключевым нефункциональным требованием. Пользовательский интерфейс должен быть разработан с использованием библиотеки Tkinter, что обеспечит интуитивно понятное взаимодействие даже для пользователей с минимальными техническими знаниями. Производительность приложения должна быть на высоком уровне, чтобы минимизировать нагрузку на систему и обеспечить стабильную работу даже на компьютерах с базовыми характеристиками.

На основании вышеуказанных условий сформировано техническое задание. Программное обеспечение должно быть создано на языке Python версии 3.10 с использованием библиотек с открытым исходным кодом, таких как Tkinter для интерфейса, SQLite для работы с базой данных, а также SQLAlchemy, NumPy, Pandas и Matplotlib для обработки и визуализации данных. Основной функционал включает реализацию тестов PVT и NASA-TLX, сохранение и экспорт данных. Программа должна быть полностью автономной и оптимизированной для работы на системах с минимальными характеристиками.

9. Разработка программного обеспечения

Для реализации логики программы выбрана интегрированная среда разработки (IDE) **PyCharm Community Edition**, которая идеально подходит для разработки на языке Python благодаря своей удобной функциональности, подсветке синтаксиса, инструментам отладки и интеграции с системами управления версиями, такими как Git.

Разработка велась на платформе операционной системы Windows, которая обеспечивала стабильность работы и совместимость с инструментами, необходимыми для создания приложения.

В рамках проекта создано приложение с рабочим названием **"Ergonomics_Assessment"**, которое было реализовано с использованием таких библиотек, как Tkinter для разработки пользовательского интерфейса, SQLAlchemy для управления базой данных SQLite, а также Pydantic и NumPy для обработки данных.

При сборке проекта используется PyInstaller, который позволяет упаковать приложение в единый исполняемый файл. Это обеспечивает удобство распространения и возможность использования приложения на различных системах без необходимости установки дополнительных зависимостей.

Итогом сборки является файл **"Ergonomics_Assessment.exe"**, который включает весь необходимый функционал для выполнения тестов, обработки данных и предоставления результатов.

9.2. Архитектура приложения

Архитектура приложения была спроектирована таким образом, чтобы обеспечить её гибкость, модульность и удобство в поддержке.

При разработке был выбран модульный подход, который позволяет чётко разделить ответственность между компонентами, упрощая их тестирование, модификацию и масштабирование.

Основой для взаимодействия модулей стало использование современных паттернов проектирования, таких как "Фасад" и "Репозиторий", которые

помогают упорядочить структуру приложения и сделать её интуитивно понятной для разработчиков.

Архитектура включает три основных слоя: пользовательский интерфейс, сервисный слой и слой данных. Каждый из них выполняет свою чётко определённую задачу, обеспечивая надёжную и стабильную работу системы.

9.3.1. Слой пользовательского интерфейса (UI)

Пользовательский интерфейс отвечает за взаимодействие с конечным пользователем. Для его реализации была выбрана библиотека Tkinter, которая предоставляет инструменты для создания простых, но функциональных графических интерфейсов.

Пользовательский интерфейс включает несколько экранов (фреймов), каждый из которых отвечает за выполнение своей задачи: авторизация, регистрация, проведение тестов и отображение результатов.

Управление экранами осуществляется через специальный компонент — менеджер окон (WindowManager). Этот менеджер обеспечивает переключение между экранами, передачу данных и их обновление.

Например, если пользователь завершает тест PVT, результаты этого теста автоматически передаются на экран завершения, где пользователь может ознакомиться с итогами.

Менеджер окон был реализован с применением паттерна "Синглтон", что гарантирует существование единственного экземпляра класса на протяжении всего времени работы приложения. Это исключает вероятность создания дублирующих менеджеров и обеспечивает централизованное управление окнами. Пример реализации класса:

```
# ui/window_manager.py

class WindowManager:
    _instance = None

    def __new__(cls, *args, **kwargs):
        if cls._instance is None:
            cls._instance = super(WindowManager, cls).__new__(cls)
        return cls._instance

    def __init__(self, root):
```

```

self.root = root
self.frames = {}
self.current_user_id = None # <-- здесь храним ID пользователя
    после логина

self.root.grid_rowconfigure(0, weight=1)
self.root.grid_columnconfigure(0, weight=1)

def create_and_register(self, name, frame_class, *args, **kwargs):
    frame = frame_class(self.root, *args, **kwargs)
    frame.grid(row=0, column=0, sticky="nsew")
    self.frames[name] = frame

def show_frame(self, name, **kwargs):
    frame = self.frames.get(name)
    if frame:
        if hasattr(frame, "update_data"):
            frame.update_data(**kwargs)
            frame.tkraise()
        else:
            raise ValueError(f"Frame '{name}' не найден!")

```

Использование паттерна "Синглтон" для менеджера окон позволяет сократить количество создаваемых объектов и улучшить управление ресурсами приложения.

9.3.2. Сервисный слой

Сервисный слой реализует бизнес-логику приложения. Он обрабатывает действия пользователя, взаимодействует с базой данных через слой данных и предоставляет результаты для отображения в интерфейсе.

Для упрощения работы и структурирования кода в сервисном слое используются паттерны "Фасад" и "Репозиторий".

Паттерн "Фасад" позволяет организовать доступ к функционалу через единый интерфейс, скрывая сложную внутреннюю реализацию. Например, сервис работы с пользователями (UserService) включает методы для проверки логина, создания нового пользователя, обновления данных и удаления записей.

Это делает работу с пользователями более упорядоченной и минимизирует количество кода, необходимого для выполнения операций.

Пример кода из сервиса работы с пользователями:

```

class UserService:

    def is_login_exists(self, login: str) -> bool:
        with UserRepository() as repo:

```



```

        return repo.is_login_exists(login)

    def create_user(self, user_data: dict) -> int:
        with UserRepository() as repo:
            return repo.create_user(user_data)

    def get_user_by_id(self, user_id: int) -> Optional[User]:
        with UserRepository() as repo:
            return repo.get_user_by_id(user_id)

    def check_user(self, login: str, password: str) -> tuple[str, int] |
None:
        """
        Возвращает (FIO, user_id), если логин и пароль правильные,
        иначе None.
        """
        with UserRepository() as repo:
            result = repo.check_user_with_id(login, password)
            if result:
                return result
            return None

    def update_user(self, user_id: int, update_data: dict) -> User:
        with UserRepository() as repo:
            return repo.update_user(user_id, update_data)

    def delete_user(self, user_id: int) -> None:
        with UserRepository() as repo:
            repo.delete_user(user_id)

    def get_all_users(self) -> list[Type[User]]:
        with UserRepository() as repo:
            return repo.get_all_users()

```

Пример кода из сервиса работы с тестами:

```

class TestService:

    def save_pvt_round(
        self,
        user_id: int,
        exercise_name: str,
        task_number: str,
        type_test: str,
        round_index: int,
        reaction_time: float
    ) -> PVTResult:
        with TestRepository() as repo:
            return repo.save_pvt_round(
                user_id=user_id,
                exercise_name=exercise_name,
                task_number=task_number,
                type_test=type_test,
                round_index=round_index,
                reaction_time=reaction_time
            )

    def get_pvt_results_for_user(self, user_id: int) -> List[PVTResult]:
        with TestRepository() as repo:
            return repo.get_pvt_results_for_user(user_id)

    def save_nasa_tlx_result(
        self,
        user_id: int,

```

```

        exercise_name: str,
        task_number: str,
        mental_demand: int,
        physical_demand: int,
        temporal_demand: int,
        performance: int,
        effort: int,
        frustration: int,
        weight_mental: int,
        weight_physical: int,
        weight_temporal: int,
        weight_performance: int,
        weight_effort: int,
        weight_frustration: int,
        weighted_tlx: float
    ) -> NASATLXResult:
        with TestRepository() as repo:
            return repo.save_nasa_tlx_result(
                user_id=user_id,
                exercise_name=exercise_name,
                task_number=task_number,
                mental_demand=mental_demand,
                physical_demand=physical_demand,
                temporal_demand=temporal_demand,
                performance=performance,
                effort=effort,
                frustration=frustration,
                weight_mental=weight_mental,
                weight_physical=weight_physical,
                weight_temporal=weight_temporal,
                weight_performance=weight_performance,
                weight_effort=weight_effort,
                weight_frustration=weight_frustration,
                weighted_tlx=weighted_tlx
            )

    def get_nasa_tlx_results_for_user(self, user_id: int) ->
List[NASATLXResult]:
        with TestRepository() as repo:
            return repo.get_nasa_tlx_results_for_user(user_id)

```

9.3.3. Слой данных

Слой данных отвечает за взаимодействие с базой данных, которая хранит всю необходимую информацию о пользователях и результатах тестов. База данных построена на основе SQLite, что делает приложение лёгким и кроссплатформенным.

Для работы с базой данных используется SQLAlchemy, предоставляющая ORM для управления данными через объекты Python.

Важной особенностью является использование паттерна "Репозиторий" для всех операций с базой данных. Это гарантирует строгое разделение между бизнес-логикой и данными, что повышает безопасность и снижает вероятность ошибок. Репозиторий инкапсулирует все операции, включая

создание, чтение, обновление и удаление записей, а также выполняет валидацию данных через Pydantic.

Пример реализации моделей для валидации данных пользователя:

```
class UserBase(BaseModel):
    name: str
    login: Optional[str] = None
    password: Optional[str] = None

# Модель для создания
class UserCreate(UserBase):
    pass

# Модель для обновления
class UserUpdate(UserBase):
    pass
```

Пример реализации моделей для валидации данных теста PVT:

```
# ===== PVT RESULT =====

class PVTResultBase(BaseModel):
    user_id: int
    exercise_name: str
    task_number: str

    type_test: Optional[str] = None # "before"/"after"
    round_index: Optional[int] = None
    reaction_time: Optional[float] = None

    timestamp: datetime = Field(default_factory=datetime.utcnow)
```

Пример реализации моделей для валидации данных теста NASA-TLX:

```
# ===== NASA TLX RESULT =====

class NASATLXResultBase(BaseModel):
    user_id: int
    exercise_name: str
    task_number: str

    mental_demand: Optional[int] = None
    physical_demand: Optional[int] = None
    temporal_demand: Optional[int] = None
    performance: Optional[int] = None
    effort: Optional[int] = None
    frustration: Optional[int] = None

    weight_mental: Optional[int] = None
    weight_physical: Optional[int] = None
    weight_temporal: Optional[int] = None
    weight_performance: Optional[int] = None
    weight_effort: Optional[int] = None
    weight_frustration: Optional[int] = None

    weighted_tlx: Optional[float] = None

    timestamp: datetime = Field(default_factory=datetime.utcnow)
```

Пример реализации репозитория для работы с пользователями:

```

class UserRepository:
    def __init__(self):
        self.db: Session = None

    def __enter__(self):
        self.db = SessionLocal()
        return self

    def __exit__(self, exc_type, exc_value, traceback):
        if self.db:
            self.db.close()

    def is_login_exists(self, login: str) -> bool:
        return self.db.query(User).filter(User.login == login).first() is
not None

    def create_user(self, user_data: dict) -> int:
        try:
            validated_data = UserSchema(**user_data)
        except ValidationError as e:
            error_details = [
                {"field": err["loc"][0], "message": err["msg"]}
                for err in e.errors()
            ]
            raise ValueError(f"Ошибка валидации: {error_details}")

        if self.is_login_exists(validated_data.login):
            raise ValueError("Пользователь с таким логином уже существует!")

        user = User(**validated_data.dict())
        self.db.add(user)
        self.db.commit()
        self.db.refresh(user)
        return user.id

    def get_user_by_id(self, user_id: int) -> Optional[User]:
        return self.db.query(User).filter(User.id == user_id).first()

    def check_user(self, login: str, password: str) -> str | None:
        return self.db.query(User.name).filter(User.login == login,
User.password == password).scalar()

```

Пример реализации репозитория для работы с тестами:

```

class TestRepository:
    def __init__(self):
        self.db: Session = None

    def __enter__(self):
        self.db = SessionLocal()
        return self

    def __exit__(self, exc_type, exc_val, exc_tb):
        if self.db:
            self.db.close()

    # ----- PVT -----

    def save_pvt_round(
        self,
        user_id: int,
        exercise_name: str,
        task_number: str,

```

```

        type_test: str,
        round_index: int,
        reaction_time: float
    ) -> PVTResult:
        """
        Сохраняет один раунд PVT-теста.
        Возвращает объект PVTResult.
        """
        result = PVTResult(
            user_id=user_id,
            exercise_name=exercise_name,
            task_number=task_number,
            type_test=type_test,
            round_index=round_index,
            reaction_time=reaction_time
        )
        self.db.add(result)
        self.db.commit()
        self.db.refresh(result)
        return result

def get_pvt_results_for_user(self, user_id: int) -> List[PVTResult]:
    return (
        self.db.query(PVTResult)
        .filter(PVTResult.user_id == user_id)
        .order_by(PVTResult.timestamp.desc())
        .all()
    )

# ----- NASA-TLX -----

def save_nasa_tlx_result(
    self,
    user_id: int,
    exercise_name: str,
    task_number: str,
    mental_demand: int,
    physical_demand: int,
    temporal_demand: int,
    performance: int,
    effort: int,
    frustration: int,
    weight_mental: int,
    weight_physical: int,
    weight_temporal: int,
    weight_performance: int,
    weight_effort: int,
    weight_frustration: int,
    weighted_tlx: float
) -> NASATLXResult:
    """
    Сохраняет итог NASA-TLX вместе с сырыми баллами и весами.
    """
    result = NASATLXResult(
        user_id=user_id,
        exercise_name=exercise_name,
        task_number=task_number,
        mental_demand=mental_demand,
        physical_demand=physical_demand,
        temporal_demand=temporal_demand,
        performance=performance,
        effort=effort,
        frustration=frustration,

```

```
weight_mental=weight_mental,  
weight_physical=weight_physical,  
weight_temporal=weight_temporal,  
weight_performance=weight_performance,  
weight_effort=weight_effort,  
weight_frustration=weight_frustration,  
weighted_tlx=weighted_tlx  
)  
self.db.add(result)  
self.db.commit()  
self.db.refresh(result)  
return result
```

9.3.4. Преимущества архитектуры

Основная цель архитектуры — обеспечить простоту в разработке, удобство для пользователя и стабильность системы. Среди основных преимуществ:

- **Модульность:** Каждый компонент отвечает за свою часть функционала, что облегчает тестирование и добавление новых возможностей.
- **Гибкость:** Паттерны "Фасад" и "Репозиторий" позволяют легко изменять внутреннюю реализацию без влияния на интерфейсы.
- **Читаемость:** Чёткое разделение слоёв делает структуру кода интуитивно понятной.
- **Кроссплатформенность:** Использование Tkinter и SQLite позволяет запускать приложение на Windows и Linux.

Эта архитектура не только отвечает всем требованиям к функциональности, но и создаёт основу для дальнейшего развития приложения.

9.3.5. Диаграмма архитектуры

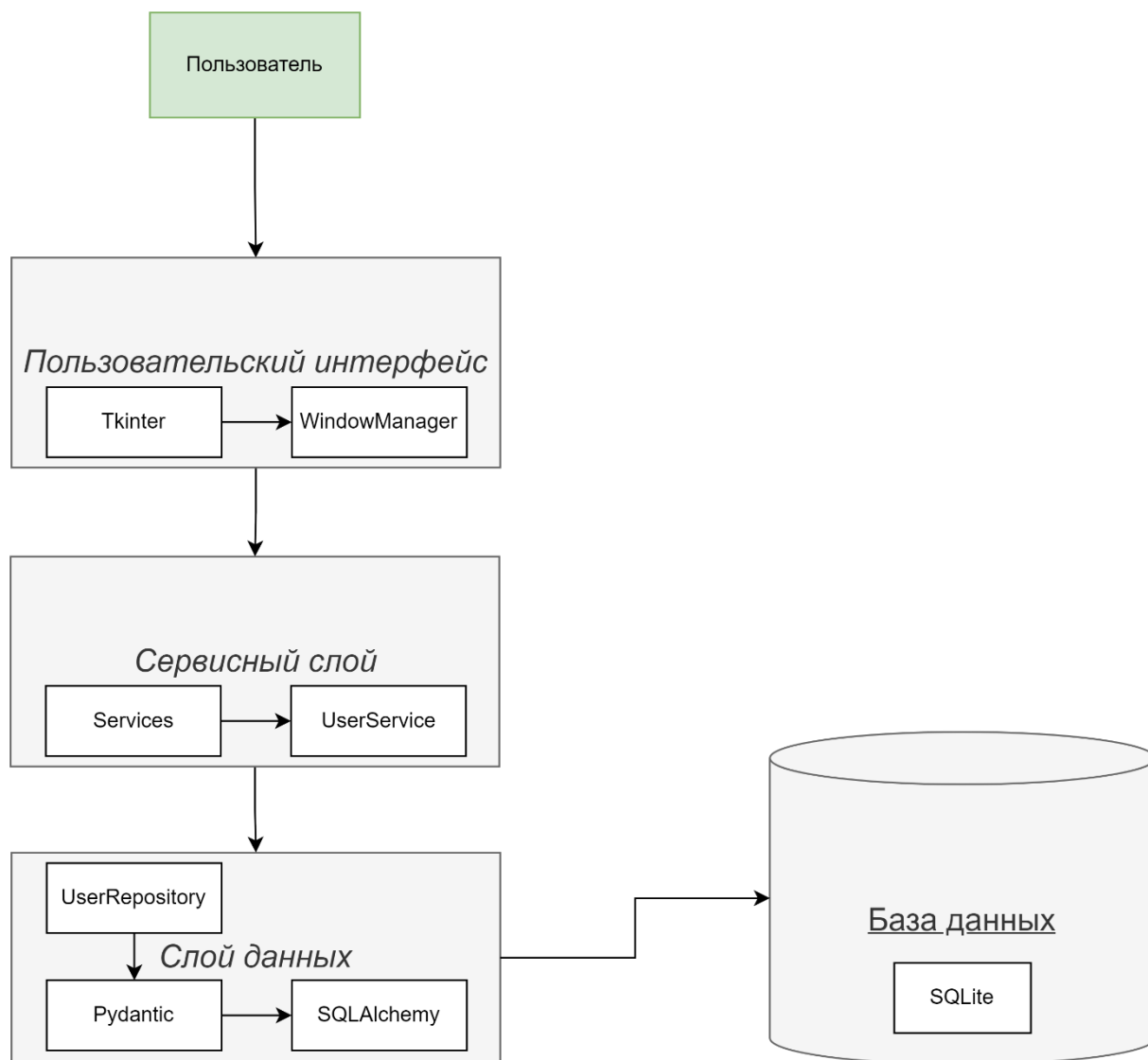


Рисунок 4 – Диаграмма архитектуры

10. Структура проекта

Структура проекта организована в виде нескольких ключевых модулей и папок, каждая из которых выполняет определённые функции. Проект состоит из следующих директорий и файлов:

```
ErgonomicsApplication
|   app.py
|   config.py
|   README.md
|   requirements.txt
|
+---app
|   |   __init__.py
|   |
|   +---database
|   |   |       database.py
|   |   |       models.py
|   |   |       __init__.py
|   |   |
|   +---repository
|   |   |       TestRepository.py
|   |   |       UserRepository.py
|   |   |       __init__.py
|   |   |
|   +---schemas
|   |   |       TestResultsSchema.py
|   |   |       UserSchema.py
|   |   |       __init__.py
|   |   |
|   \---services
|   |       TestService.py
|   |       UserService.py
|   |       __init__.py
|   |
+---tests
|   |       test_nasa_tlx_logic.py
|   |       test_pvt_logic.py
|   |       test_user_service.py
|   |       __init__.py
|   |
\---ui
    |       window_manager.py
    |       __init__.py
    |
    +---assets
    |   |       logo.png
    |   |
    |   \---icons
    |       |       test.png
    |       |       user.png
    |
    +---components
    |   |       CenteredFrame.py
    |   |       ErrorLabel.py
    |   |       StyledButton.py
    |   |       __init__.py
    |   |
    \---screens
        |       AuthWindow.py
        |       LoginWindow.py
```



```
MainWindow.py
NASA_TLXWindow.py
PVTWindow.py
RegisterWindow.py
ReportWindow.py
ResultsWindow.py
TestInstructionWindow.py
TestSelectionWindow.py
init.py
```

app — включает в себя основную бизнес-логику, модели данных и взаимодействие с базой данных:

- **schemas:** Содержит схемы для валидации данных с использованием библиотеки Pydantic.
 - `UserSchema.py`: Определяет структуру данных для пользователя.
 - `TestResultsSchema.py`: Определяет структуру результатов тестирования.
- **services:** Реализует слой бизнес-логики. Например, `UserService.py` отвечает за управление данными пользователей.
- **repository:** Содержит репозитории для работы с базой данных. Репозитории инкапсулируют операции CRUD.
- **database:** Включает модели базы данных и функции для инициализации подключения.

ui — отвечает за графический интерфейс приложения:

- **screens:** Содержит файлы, представляющие основные экраны приложения, такие как авторизация (`AuthWindow.py`), выбор теста (`TestSelectionWindow.py`), тесты (`PVTWindow.py`, `NASA_TLXWindow.py`) и результаты (`ResultsWindow.py`).
- **components:** Содержит переиспользуемые элементы интерфейса, например кнопки с определённым стилем (`StyledButton.py`) и центрированные фреймы (`CenteredFrame.py`).
- **assets:** Хранит статические ресурсы, такие как изображения и иконки.

tests — модуль для автоматического тестирования. Включает тесты для проверки работы бизнес-логики, таких как логика проведения тестов PVT и NASA-TLX.

11. Потоки данных в системе

Работа программы организована таким образом, чтобы обеспечить последовательное выполнение всех этапов от авторизации до генерации отчётов, гарантируя сохранность данных и их корректную обработку.

Пользователь начинает взаимодействие с приложением на этапе авторизации. Для входа в систему требуется ввести логин и пароль. Эти данные проверяются в базе данных, чтобы убедиться в их корректности. Если логин и пароль совпадают с сохранёнными значениями, программа сохраняет уникальный идентификатор пользователя (`user_id`) для последующего использования. Уникальный идентификатор пользователя становится доступным в каждом кадре программы, что позволяет всегда иметь доступ к пользователю, персонализировать дальнейшую работу и связать результаты тестов с конкретным пользователем. В случае неудачной авторизации пользователю выводится соответствующее сообщение об ошибке.

После успешной авторизации пользователь попадает на экран выбора упражнения и задачи. Здесь он выбирает, какое упражнение и задачу он будет выполнять, например, тестирование до или после выполнения тренировочного упражнения. Выбранные параметры передаются в слой логики приложения, где они используются для настройки параметров тестов PVT и NASA-TLX.

На этапе проведения тестов, если выбран PVT, система отображает визуальные стимулы, на которые пользователь должен реагировать как можно быстрее. Каждое нажатие фиксируется с точностью до миллисекунд, и программа сохраняет время реакции пользователя для каждого раунда. Результаты записываются в базу данных, что позволяет проанализировать их позже, сравнить с предыдущими тестами или использовать для построения отчётов.

В случае проведения NASA-TLX пользователь оценивает уровень умственной, физической и временной нагрузки, а также усилия, производительность и уровень стресса по 100-балльной шкале. Эти оценки обрабатываются алгоритмами, которые рассчитывают взвешенный показатель

нагрузки (Weighted TLX). Вес каждого параметра определяется индивидуально для каждого теста путём парного сравнения, чтобы учесть субъективное восприятие важности каждого аспекта нагрузки.

После завершения всех тестов начинается этап генерации отчётов. Программа автоматически формирует графики и статистические сводки, которые позволяют визуализировать изменения когнитивных и физических характеристик пользователя. Отчёты могут включать сравнение времён реакции до и после выполнения упражнений, визуализацию шкал NASA-TLX в виде диаграмм и обобщённые результаты. Все отчёты сохраняются в системе, что позволяет пользователю или администратору обратиться к ним в любой момент.

12. Реализация ключевых функций

15.1. Авторизация пользователя

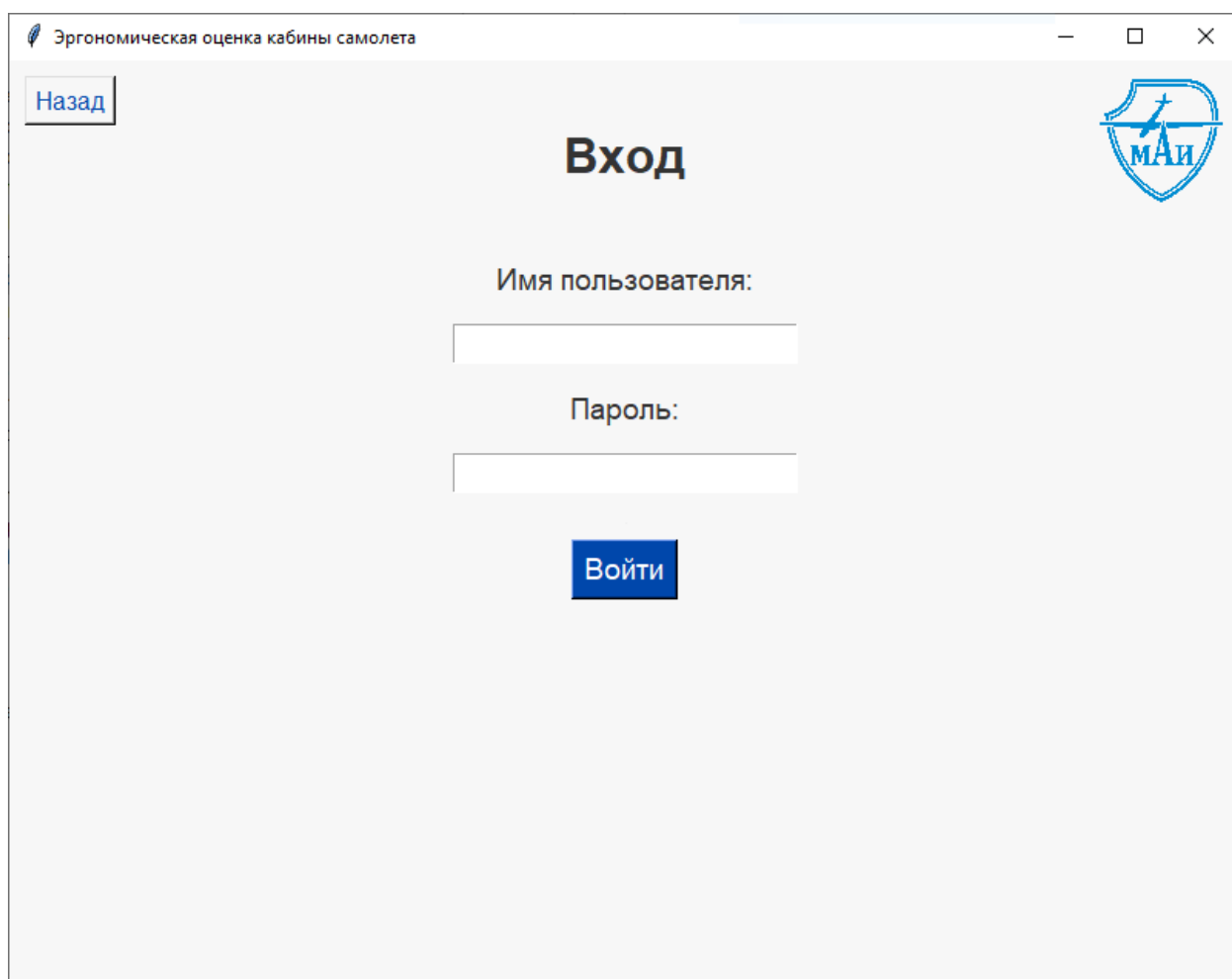
Процесс авторизации пользователя включает в себя проверку введённых данных (логина и пароля) на соответствие информации, хранящейся в базе данных. Это позволяет обеспечить доступ только авторизованным пользователям и связать их действия с уникальным идентификатором (user_id), используемым в других частях программы. Логика авторизации выглядит следующим образом:

```
class LoginWindow(CenteredFrame):  
  
    ...  
  
    def login(self):  
        username = self.username_entry.get()  
        password = self.password_entry.get()  
  
        result = services.user.check_user(login=username, password=password)  
  
        if result is None:  
            self.error_label.config(text="Неверный логин или пароль!")  
            return  
  
        FIO, user_id = result  
  
        self.master.window_manager.current_user_id = user_id  
  
        self.master.window_manager.show_frame("MainWindow", FIO=FIO)
```

При успешной авторизации идентификатор пользователя сохраняется в системе, что позволяет отслеживать, какие тесты были выполнены и кем. В

случае ввода неверных данных программа выводит сообщение об ошибке, и пользователю предлагается повторить попытку.

12.4.1. Внешний вид кадра авторизация



The screenshot shows a web application window titled "Эргономическая оценка кабины самолета". In the top-left corner, there is a "Назад" (Back) button. In the top-right corner, there is a logo of the MAI (Moscow Aviation Institute) featuring a shield with a stylized aircraft and the letters "МАИ". The main heading in the center is "Вход" (Login). Below the heading, there are two input fields: the first is labeled "Имя пользователя:" (Username:) and the second is labeled "Пароль:" (Password:). Below these fields is a blue button labeled "Войти" (Login).

Рисунок 5 – Внешний вид кадра авторизация

На кадре представлены поля для ввода данных для авторизации.

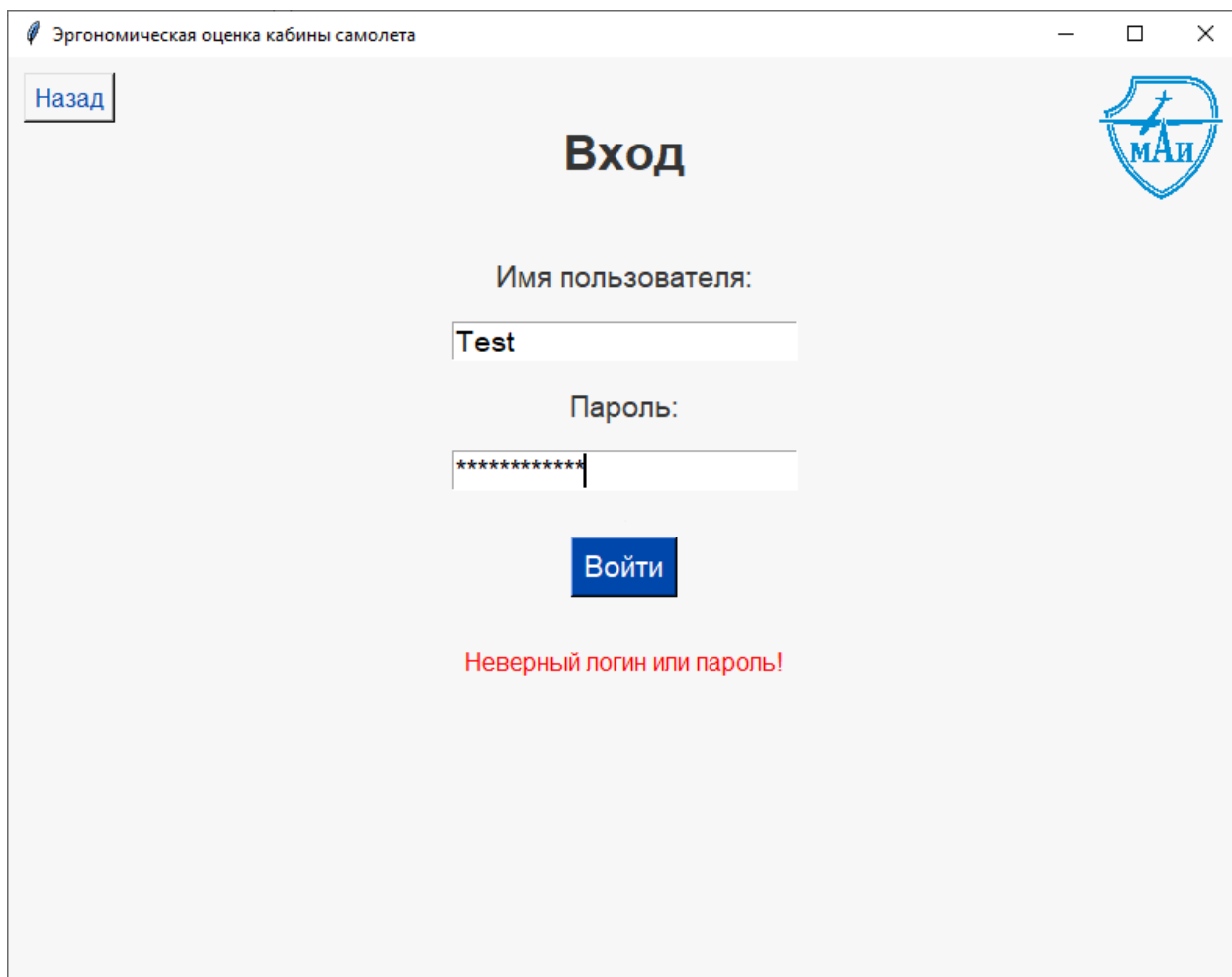


Рисунок 6 – Внешний вид кадра авторизация при неверно введенных данных

15.2. Регистрация пользователя

Процесс регистрации пользователя включает ввод пользователем персональных данных, таких как ФИО, логин и пароль, и их последующую проверку на корректность. Это обеспечивает создание уникальной учётной записи для доступа к системе. Регистрация включает несколько ключевых этапов: проверка уникальности логина, валидацию и соответствия паролей, создание пользователя в базе данных:

```
class RegisterWindow(CenteredFrame):
    def __init__(self, master):
        super().__init__(master)
        self.services = services
        self.configure(bg="#f7f7f7")

        tk.Label(self, text="Регистрация", font=("Arial", 24, "bold"),
bg="#f7f7f7", fg="#333").pack(pady=20)

        tk.Label(self, text="ФИО", font=("Arial", 14), bg="#f7f7f7",
fg="#333").pack()
        self.FIO_entry = tk.Entry(self, font=("Arial", 14))
        self.FIO_entry.pack(pady=5)
```

```

        tk.Label(self, text="Имя пользователя:", font=("Arial", 14),
bg="#f7f7f7", fg="#333").pack()
        self.login_entry = tk.Entry(self, font=("Arial", 14))
        self.login_entry.pack(pady=5)
        self.login_entry.bind("<KeyRelease>", self.validate_login)

        tk.Label(self, text="Пароль", font=("Arial", 14), bg="#f7f7f7",
fg="#333").pack()
        self.password_entry = tk.Entry(self, show="*", font=("Arial", 14))
        self.password_entry.pack(pady=5)
        self.password_entry.bind("<KeyRelease>", self.validate_passwords)

        tk.Label(self, text="Подтвердите пароль", font=("Arial", 14),
bg="#f7f7f7", fg="#333").pack()
        self.confirm_password_entry = tk.Entry(self, show="*",
font=("Arial", 14))
        self.confirm_password_entry.pack(pady=5)
        self.confirm_password_entry.bind("<KeyRelease>",
self.validate_passwords)

        self.error_label = tk.Label(self, text="", fg="red", bg="#f7f7f7",
font=("Arial", 12))
        self.error_label.pack(pady=10)

        self.register_button = tk.Button(
            self,
            text="Регистрация",
            font=("Arial", 14),
            bg="#0047ab",
            fg="white",
            activebackground="#0051c7",
            activeforeground="white",
            command=self.register
        )
        self.register_button.pack(pady=10)

        self.back_button = tk.Button(
            self,
            text="Назад",
            font=("Arial", 14),
            bg="#0047ab",
            fg="white",
            activebackground="#0051c7",
            activeforeground="white",
            command=lambda: master.window_manager.show_frame("AuthWindow")
        )
        self.back_button.place(x=10, y=10)

    def validate_login(self, event=None):
        login = self.login_entry.get()
        if len(login) < 3 or len(login) > 50:
            self.error_label.config(text="Логин должен быть от 3 до 50
СИМВОЛОВ")
        else:
            try:
                if self.services.user.is_login_exists(login):
                    self.error_label.config(text="Логин уже используется!")
            except Exception as e:
                self.error_label.config(text=f"Ошибка проверки логина: {e}")

    def validate_passwords(self, event=None):
        password = self.password_entry.get()

```

```

confirm_password = self.confirm_password_entry.get()

if len(password) < 4:
    self.error_label.config(text="Пароль слишком короткий!")
elif password != confirm_password:
    self.error_label.config(text="Пароли не совпадают!")
else:
    self.error_label.config(text="")

def register(self):
    FIO = self.FIO_entry.get()
    login = self.login_entry.get()
    password = self.password_entry.get()
    confirm_password = self.confirm_password_entry.get()

    self.error_label.config(text="")

    if password != confirm_password:
        self.error_label.config(text="Пароли не совпадают!")
        return

    user_data = {"name": FIO, "login": login, "password": password}

    try:
        user_id = self.services.user.create_user(user_data=user_data)
        self.master.window_manager.current_user_id = user_id
        print(f"[DEBUG] Регистрация прошла успешно. user_id={user_id}")

        tk.Label(self, text="Успешно зарегистрирован!", fg="green",
bg="#f7f7f7", font=("Arial", 14)).pack()
        self.master.window_manager.show_frame("MainWindow", FIO=FIO)
    except ValidationError as e:
        first_error = e.errors()[0]
        field = first_error["loc"][0]
        message = first_error["msg"]
        self.error_label.config(text=f"Ошибка в поле '{field}':
{message}")
    except ValueError as e:
        self.error_label.config(text=str(e))
    except Exception as e:
        self.error_label.config(text=f"Возникла ошибка: {e}")

```

Если данные успешно проверены и сохранены, пользователю выводится сообщение об успешной регистрации, после чего он перенаправляется на главный экран приложения. Этот процесс минимизирует возможность дублирования аккаунтов и обеспечивает безопасность учётных данных.

12.4.1. Внешний вид кадра регистрации

Эргономическая оценка кабины самолета

[Назад](#)

Регистрация

ФИО

Имя пользователя:

Пароль

Подтвердите пароль

[Регистрация](#)

Рисунок 7 – Внешний вид кадра регистрации

На кадре представлены поля для ввода данных для регистрации.

Эргономическая оценка кабины самолета

Назад

Регистрация

ФИО

Курнаев Данила Владими

Имя пользователя:

dkurnaev_error

Пароль

Подтвердите пароль

Логин уже используется!

Регистрация

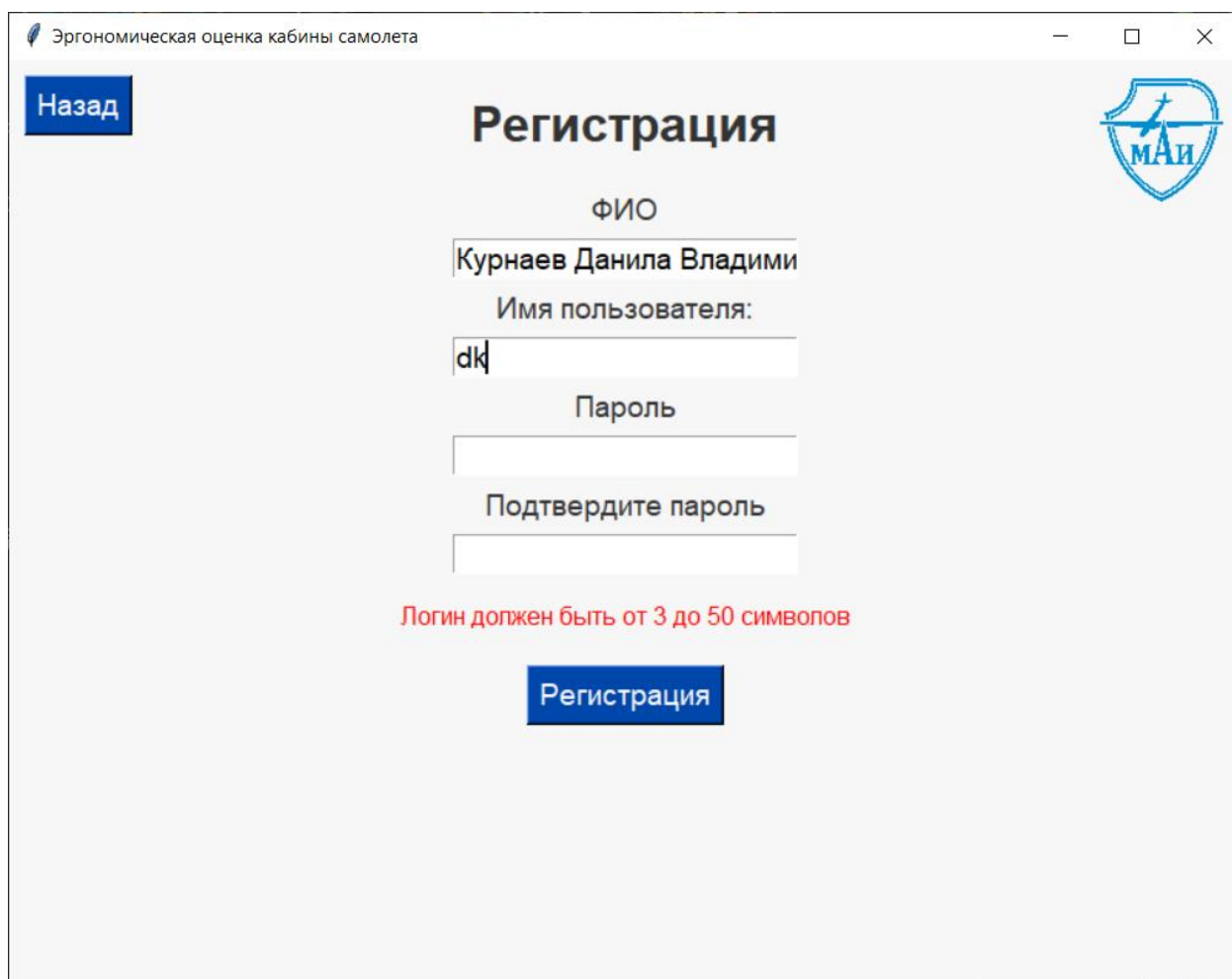
Рисунок 8 – Внешний вид кадра регистрации при неуникальном имени пользователя

На данном кадре представлен интерфейс окна регистрации, в котором пользователь попытался создать учётную запись с логином, уже существующим в базе данных. Программа проверяет уникальность логина при каждом изменении соответствующего поля ввода.

Если введённое имя пользователя не уникально, под полем логина появляется сообщение об ошибке: «Логин уже используется!». Это позволяет пользователю сразу понять, что необходимо выбрать другое имя для завершения процесса регистрации.

Такая реализация предотвращает дублирование записей в базе данных и обеспечивает уникальность каждого пользователя системы, что важно для корректной работы приложения, особенно в части сохранения и анализа

данных. Пользователь может продолжить ввод данных или изменить логин, чтобы выполнить регистрацию.



Эргономическая оценка кабины самолета

Назад

Регистрация

ФИО

Курнаев Данила Владими

Имя пользователя:

dk

Пароль

Подтвердите пароль

Логин должен быть от 3 до 50 символов

Регистрация

МАИ

Рисунок 9 – Внешний вид кадра регистрации при некорректной длине имени пользователя

На данном кадре представлен интерфейс окна регистрации, в котором пользователь попытался создать учётную запись с логином, длина которого меньше допустимой величины. Программа проверяет длину логина при каждом изменении соответствующего поля ввода.

Эргономическая оценка кабины самолета

Назад

Регистрация

ФИО

Курнаев Данила Владими

Имя пользователя:

dkurnaev

Пароль

Подтвердите пароль

Пароль слишком короткий!

Регистрация

Рисунок 10 – Внешний вид кадра регистрации при некорректной длине пароля

На данном кадре представлен интерфейс окна регистрации, в котором пользователь попытался создать учётную запись с паролем, длина которого меньше допустимой величины. Программа проверяет длину пароля при каждом изменении соответствующего поля ввода.

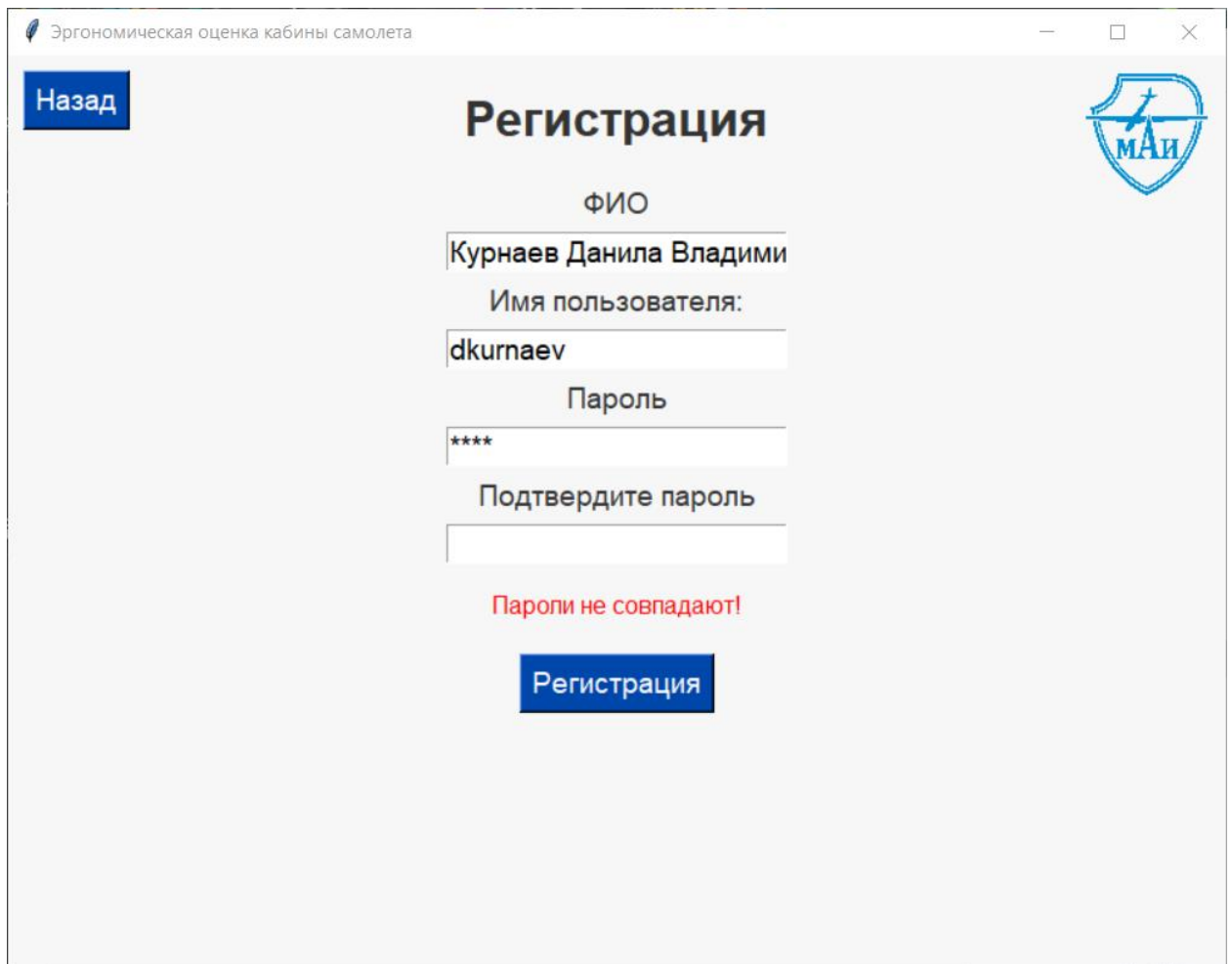


Рисунок 11 – Внешний вид кадра регистрации при несовпадении пароля

На данном кадре представлен интерфейс окна регистрации, в котором пользователь попытался создать учётную запись с паролем, который не совпадает с подтверждённым паролем. Программа проверяет соответствие введённых паролей при каждом изменении поля ввода, чтобы гарантировать их идентичность.

12.4. Проведение PVT

PVT (Psychomotor Vigilance Test) представляет собой тестирование на время реакции, где пользователь должен как можно быстрее реагировать на визуальные стимулы. Логика работы этого теста включает отображение сигнала, измерение времени реакции и сохранение данных в базу. После каждого раунда информация о времени реакции записывается, что позволяет проводить анализ эффективности выполнения задания.

```
class PVTWindow(CenteredFrame):
    """
```

```

Окно для проведения PVT (Psychomotor Vigilance Test).
Цикл:
- Несколько раундов (max_rounds).
- В каждом раунде случайная задержка 2-5 секунд, потом «ЖМИ!».
- Засекаем время реакции (мс).
- В конце выводим статистику (среднее, мин, макс).
- Если type_test='before', это «перед упражнением».
- Если type_test='after', это «после упражнения».
"""
def __init__(self, master):
    super().__init__(master)

    self.user_id = None
    self.task_number = None
    self.exercise_name = None
    self.type_test = None          # "before" или "after"
    self.max_rounds = 5           # количество раундов
    self.current_round = 0
    self.results = []             # список времени реакции за каждый раунд
(в сек)

    self.start_time = None
    self.services = Services()

    self.label = tk.Label(
        self,
        text="PVT: Нажмите кнопку, как только увидите сигнал",
        font=("Arial", 14),
        bg="#f7f7f7",
        fg="#333"
    )
    self.label.pack(pady=40)

    self.button = tk.Button(
        self,
        text="Начать тест",
        command=self.start_round,
        font=("Arial", 14),
        bg="#0047ab",
        fg="white",
        activebackground="#0051c7",
        activeforeground="white"
    )
    self.button.pack(pady=20)

    # Кнопка «Назад»: возвращается на экран выбора теста
    self.back_button = tk.Button(
        self,
        text="Назад",
        command=lambda:
master.window_manager.show_frame("TestSelectionWindow"),
        font=("Arial", 12),
        bg="#f7f7f7",
        fg="#333"
    )
    self.back_button.place(x=10, y=10)

    def update_data(self, type_test: str = "before", exercise_name='Взлет',
task_number='1', **kwargs):
        """
        Вызывается при показе экрана.
        Сбрасываем результаты, раунды, выставаем type_test = 'before' или
'after'.

```

```

        """
        self.type_test = type_test
        self.exercise_name = exercise_name
        self.task_number = task_number

        self.results.clear()
        self.current_round = 0

        self.label.config(text="PVT: Нажмите кнопку, как только увидите
сигнал")
        self.button.config(text="Начать тест", state=tk.NORMAL,
command=self.start_round)

    def start_round(self):
        """
        Запускаем очередной раунд (если не достигли max_rounds).
        """
        if self.current_round >= self.max_rounds:
            # Все раунды пройдены - завершаем
            self.finish_test()
            return

        self.label.config(
            text=(
                f"PVT: Раунд {self.current_round + 1} из {self.max_rounds}."
                "Ожидайте сигнал..."
            )
        )
        self.button.config(state=tk.DISABLED)

        # Случайная задержка 2-5 секунд до появления «ЖМИ!»
        delay_ms = random.randint(2000, 5000)
        self.master.after(delay_ms, self.show_signal)

    def show_signal(self):
        """
        Показываем сигнал "ЖМИ!" и включаем засекание времени.
        """
        self.start_time = time.time()
        self.label.config(text="ЖМИ!")
        self.button.config(text="Нажать сейчас!", state=tk.NORMAL,
command=self.record_response)

    def record_response(self):
        """
        Засекаем время реакции, сохраняем и переходим к следующему раунду.
        """
        end_time = time.time()
        reaction_time_s = end_time - self.start_time
        self.results.append(reaction_time_s)

        self.user_id = self.master.window_manager.current_user_id

        self.services.test.save_pvt_round(
            user_id=self.user_id,
            exercise_name=self.exercise_name,
            task_number=self.task_number,
            type_test=self.type_test,
            round_index=self.current_round,
            reaction_time=reaction_time_s
        )

```

```

        self.label.config(
            text=f"Время реакции: {reaction_time_s * 1000:.0f} мс"
        )
        self.current_round += 1

        if self.current_round < self.max_rounds:
            self.button.config(text="Следующий раунд",
                                command=self.start_round, state=tk.NORMAL)
        else:
            self.button.config(text="Завершить тест",
                                command=self.finish_test, state=tk.NORMAL)

    def finish_test(self):
        """
        Расчёт итогов: среднее, мин, макс. Переход на следующий экран.
        """
        if len(self.results) == 0:
            avg_ms = 0
            min_ms = 0
            max_ms = 0
        else:
            avg_ms = sum(self.results) / len(self.results) * 1000
            min_ms = min(self.results) * 1000
            max_ms = max(self.results) * 1000

        self.label.config(
            text=(
                f"Тест завершён!\n"
                f"Среднее время: {avg_ms:.0f} мс\n"
                f"Мин. время: {min_ms:.0f} мс\n"
                f"Макс. время: {max_ms:.0f} мс"
            )
        )

        # Кнопка для перехода на «ExerciseWaitWindow» или, если это after,
        сразу на NASA-TLX
        if self.type_test == "before":
            self.button.config(
                text="Перейти к упражнению",
                command=lambda:
                    self.master.window_manager.show_frame("ExerciseWaitWindow"),
                state=tk.NORMAL
            )
        else:
            self.button.config(
                text="Перейти к NASA-TLX",
                command=lambda:
                    self.master.window_manager.show_frame("TestInstructionWindow",
                                                            current_test="NASA-TLX Test"),
                state=tk.NORMAL
            )

```

Таким образом, система собирает все данные о реакции пользователя в каждом раунде. Эти данные могут быть использованы для построения статистики, вычисления среднего времени реакции и оценки когнитивных способностей до и после выполнения упражнений.

12.3.1. Внешний вид кадра PVT тестирования

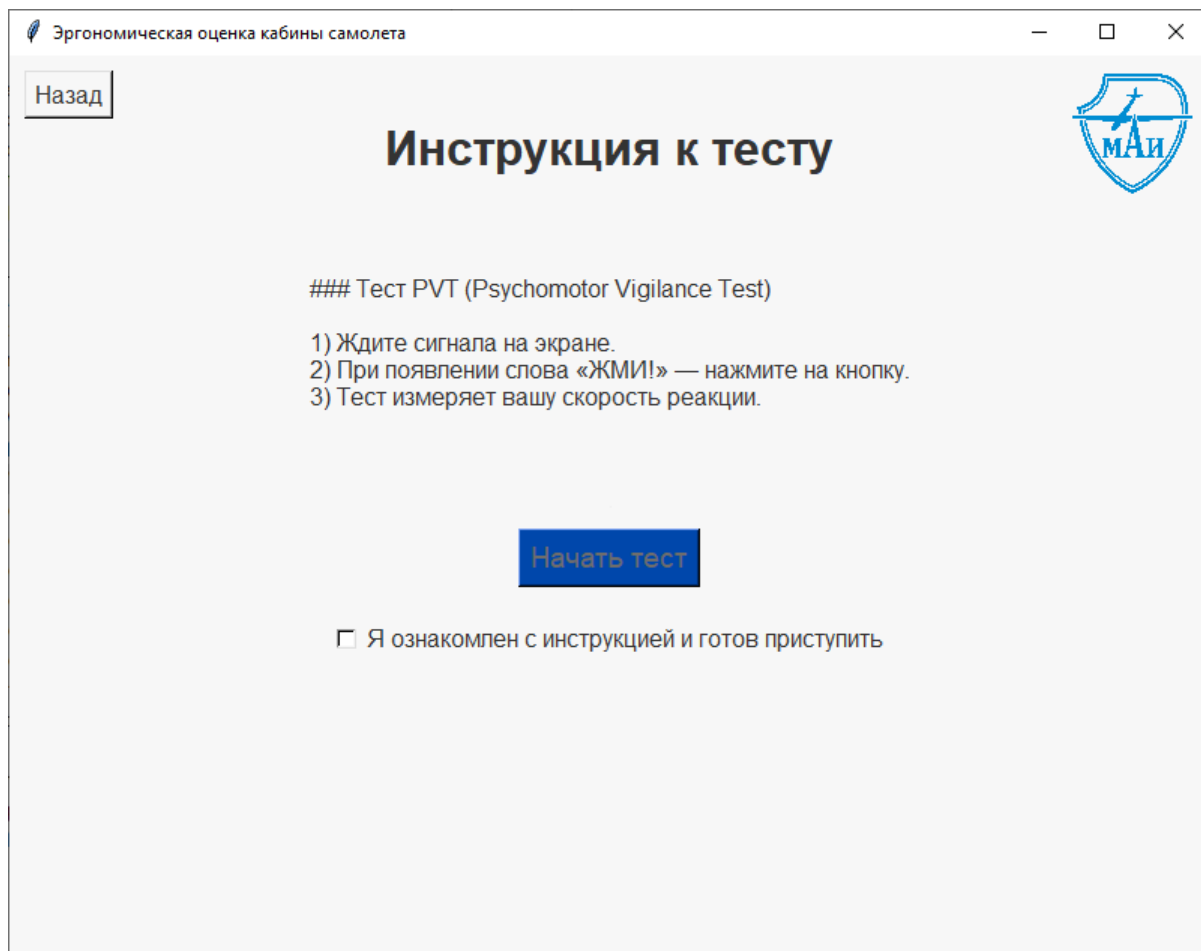


Рисунок 12 – Внешний вид кадра инструкции выполнения PVT теста

На данном кадре представлена краткая инструкция, поясняющая основные этапы и правила прохождения теста.

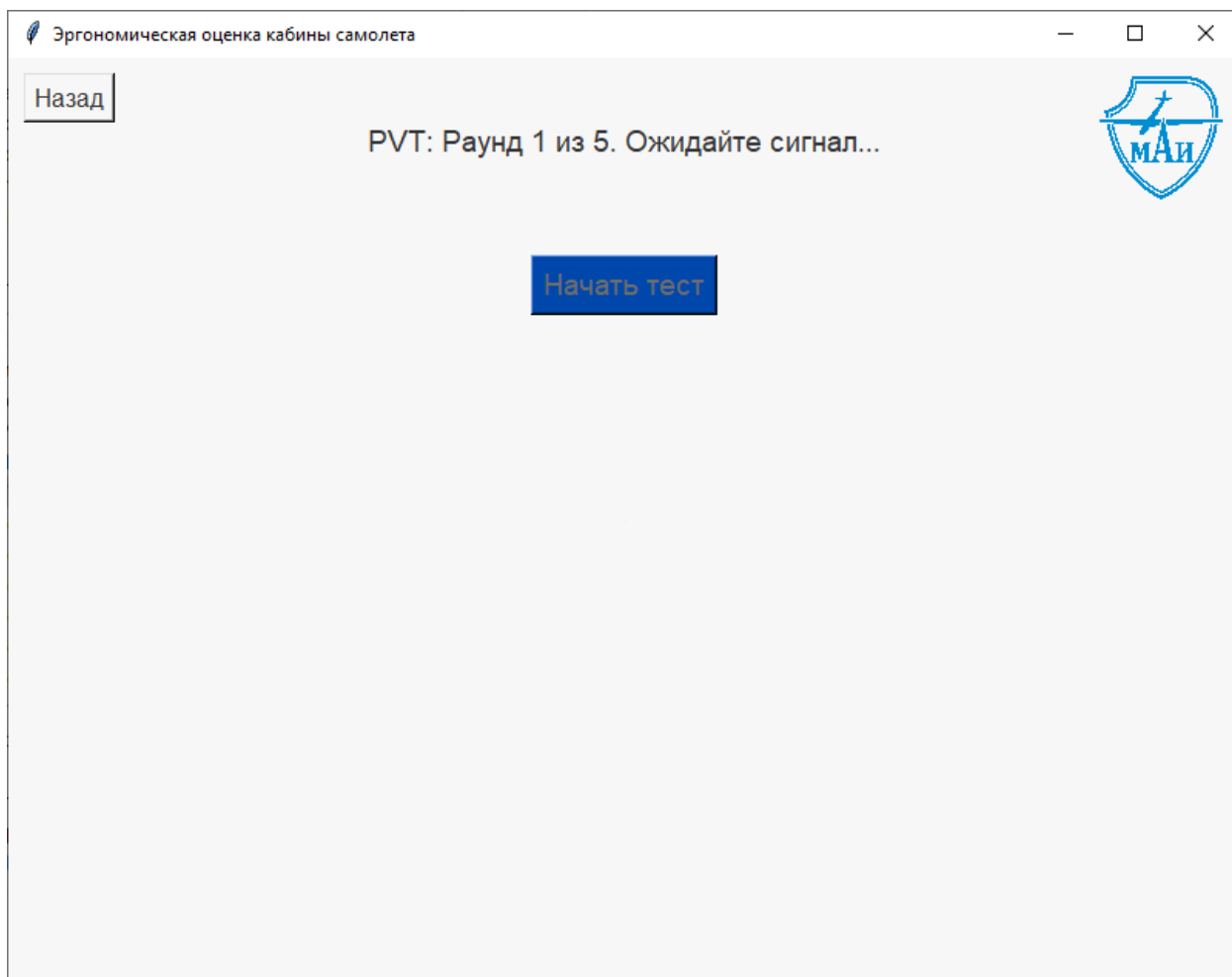


Рисунок 13 – Внешний вид кадра ожидания сигнала

На данном кадре мы видим момент, когда испытуемый (или участник эксперимента) находится в состоянии ожидания сигнала, описанного в инструкции ранее.

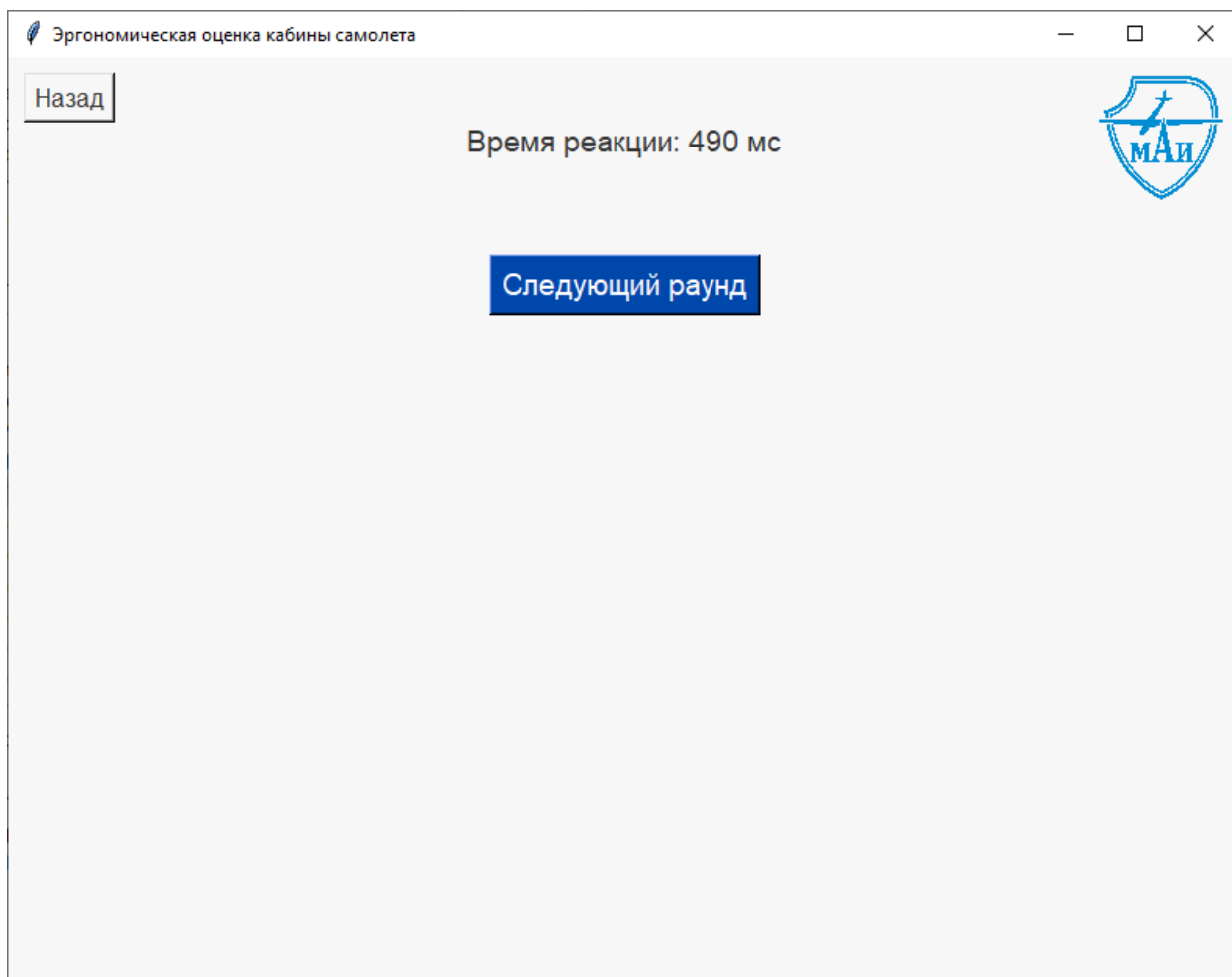


Рисунок 14 – Внешний вид кадра вывода результата прошедшего раунда

Внешний вид кадра вывода результата прошедшего раунда. Оформление сделано таким образом, чтобы участник мог быстро и наглядно понять, как он справился с поставленной задачей.

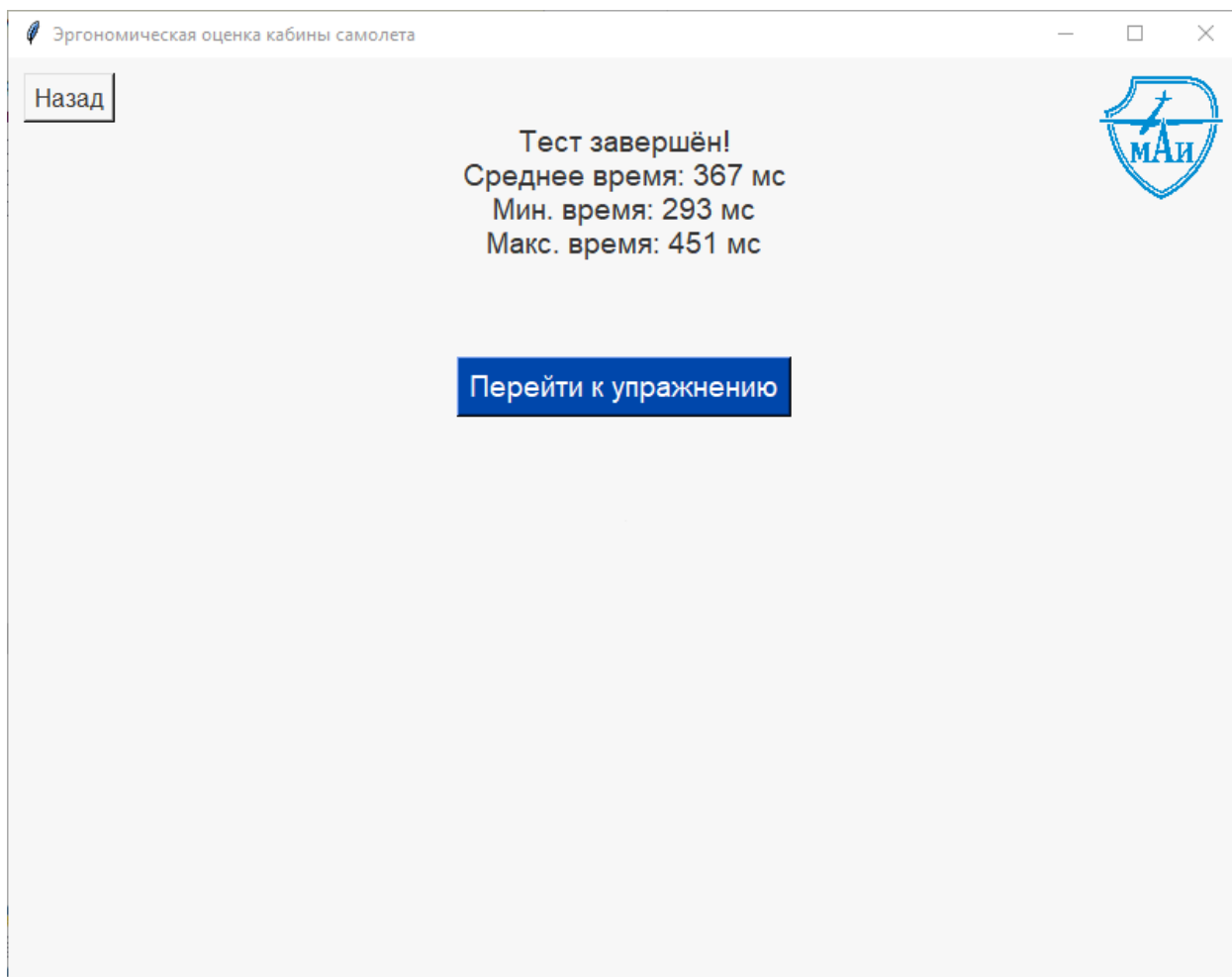


Рисунок 15 – Внешний вид кадра общих результатов по раундам

На данном кадре представлен сводный экран с общими результатами по завершенным раундам.

Отдельно отображаются три ключевые метрики времени реакции: среднее, максимальное и минимальное значение.

По завершении просмотра результаты автоматически записываются в базу данных для дальнейшего анализа и хранения.

12.4. Расчёт NASA-TLX

NASA-TLX (Task Load Index) используется для оценки уровня рабочей нагрузки. Пользователь вводит субъективные оценки по шести шкалам, включая умственную и физическую нагрузку, временные ограничения, усилия, уровень стресса и производительность. После этого программа рассчитывает взвешенный показатель нагрузки (Weighted TLX), используя данные парных сравнений.

Каждая шкала оценивается на основе приоритетов, определённых пользователем в ходе парных сравнений. Это позволяет учитывать индивидуальное восприятие нагрузок и получить более точный итоговый показатель. Рассчитанные результаты сохраняются в базе данных для дальнейшего анализ.

```
# ui/screens/NASA_TLXWindow.py
import tkinter as tk

from app.services import Services
from ui.components.CenteredFrame import CenteredFrame

class NASA_TLXWindow(CenteredFrame):
    """
    Окно NASA-TLX с 2 шагами:
    1) Оценка по 6 шкалам (0-100).
    2) 15 парных сравнений для определения весов.
    3) Итоговый расчёт Weighted TLX.

    Шкалы:
    - mental_demand
    - physical_demand
    - temporal_demand
    - performance
    - effort
    - frustration
    """
    def __init__(self, master):
        super().__init__(master)

        self.services = Services()

        self.exercise_name = None
        self.task_number = None
        self.user_id = None

        self.step = 0 # 0=шкалы, 1=парные сравнения, 2=конец
        self.questions = [
            "Mental Demand (Умственная нагрузка)",
            "Physical Demand (Физическая нагрузка)",
            "Temporal Demand (Временная нагрузка)",
            "Performance (Удовлетворённость выполнением)",
            "Effort (Общие усилия)",
            "Frustration (Уровень фрустрации)"
        ]
        # Храним оценки (0..100)
        self.scores = [0, 0, 0, 0, 0, 0]
        self.current_question = 0

        # Для шкал
        self.label_question = tk.Label(self, text="", font=("Arial", 14),
            wraplength=400, bg="#f7f7f7", fg="#333")
        self.label_question.pack(pady=20)

        self.scale = tk.Scale(self, from_=0, to=100, orient=tk.HORIZONTAL,
            length=300, font=("Arial", 12))
        self.scale.pack(pady=10)

        self.next_button = tk.Button(
```

```

        self,
        text="Далее",
        command=self.next_action,
        font=("Arial", 14),
        bg="#0047ab",
        fg="white",
        activebackground="#0051c7",
        activeforeground="white"
    )
    self.next_button.pack(pady=20)

    self.back_button = tk.Button(
        self,
        text="Отменить тест",
        command=lambda: master.window_manager.show_frame("MainWindow"),
        font=("Arial", 12),
        bg="#f7f7f7",
        fg="#333"
    )
    self.back_button.place(x=10, y=10)

    # Структуры для «весов»
    self.all_pairs = [
        ("Mental", "Physical"), ("Mental", "Temporal"), ("Mental",
"Performance"), ("Mental", "Effort"), ("Mental", "Frustration"),
        ("Physical", "Temporal"), ("Physical", "Performance"),
        ("Physical", "Effort"), ("Physical", "Frustration"),
        ("Temporal", "Performance"), ("Temporal", "Effort"),
        ("Temporal", "Frustration"),
        ("Performance", "Effort"), ("Performance", "Frustration"),
        ("Effort", "Frustration")
    ]
    # Привязка названий к индексам
    self.name_to_index = {
        "Mental": 0,
        "Physical": 1,
        "Temporal": 2,
        "Performance": 3,
        "Effort": 4,
        "Frustration": 5
    }
    # Сколько раз каждая шкала выиграла
    self.weights = [0, 0, 0, 0, 0, 0] # соответствует 6 шкалам

    # Для парных сравнений
    self.pair_index = 0
    self.label_pair = tk.Label(self, text="", font=("Arial", 14),
bg="#f7f7f7", fg="#333")
    # Кнопки сравнения
    self.button_left = tk.Button(self, text="", font=("Arial", 12),
command=self.choose_left)
    self.button_right = tk.Button(self, text="", font=("Arial", 12),
command=self.choose_right)

    def update_data(self, exercise_var=None, task_number=None, user_id=None,
**kwargs):
        """
        Сбрасываем всё при входе на экран.
        """
        self.exercise_name = exercise_var
        self.task_number = task_number
        self.user_id = user_id

```

```

self.step = 0
self.current_question = 0
self.pair_index = 0
self.weights = [0, 0, 0, 0, 0, 0]
self.scores = [0, 0, 0, 0, 0, 0]

self.show_question_step()

def show_question_step(self):
    """
    Шаг 1: Показываем вопрос из списка
    self.questions[self.current_question].
    """
    if self.current_question < len(self.questions):
self.label_question.config(text=f"{self.questions[self.current_question]}\n(
0=минимум, 100=максимум)")
        self.scale.set(50) # Среднее значение
    else:
        # Переходим к парным сравнениям
        self.step = 1
        self.show_pairs_step()

def next_action(self):
    """
    Универсальная кнопка "Далее".
    Если step=0, мы собираем шкалы.
    Если step=1, парные сравнения.
    """
    if self.step == 0:
        self.save_current_score()
    elif self.step == 1:
        # Если мы на шаге парных сравнений - ничего не делаем в
next_button (т.к. у нас кнопки Left/Right)
        pass

def save_current_score(self):
    response = self.scale.get()
    self.scores[self.current_question] = response
    self.current_question += 1
    if self.current_question < len(self.questions):
        self.show_question_step()
    else:
        # Закончили шкалы
        self.show_pairs_step()

def show_pairs_step(self):
    """
    Переходим ко 2-му шагу (парные сравнения).
    Прячем виджеты для шкал, показываем виджеты для сравнения.
    """
    self.step = 1
    self.label_question.pack_forget()
    self.scale.pack_forget()
    self.next_button.pack_forget()

    self.label_pair.pack(pady=20)
    self.button_left.pack(side="left", padx=50)
    self.button_right.pack(side="right", padx=50)

    self.show_next_pair()

def show_next_pair(self):

```

```

"""
Показываем пару. Например, ("Mental", "Physical").
"""
if self.pair_index >= len(self.all_pairs):
    # Все пары сравнили
    self.finish_weighting()
    return

left_name, right_name = self.all_pairs[self.pair_index]
self.label_pair.config(text=f"Что было важнее?\n{left_name} vs
{right_name}")
self.button_left.config(text=left_name)
self.button_right.config(text=right_name)

def choose_left(self):
    """
    Если пользователь выбрал левую шкалу, увеличиваем её вес.
    """
    left_name, _ = self.all_pairs[self.pair_index]
    idx = self.name_to_index[left_name]
    self.weights[idx] += 1

    self.pair_index += 1
    self.show_next_pair()

def choose_right(self):
    """
    Если пользователь выбрал правую шкалу, увеличиваем её вес.
    """
    _, right_name = self.all_pairs[self.pair_index]
    idx = self.name_to_index[right_name]
    self.weights[idx] += 1

    self.pair_index += 1
    self.show_next_pair()

def finish_weighting(self):
    """
    Завершаем парные сравнения.
    Переходим к подсчёту Weighted NASA-TLX.
    """
    self.step = 2
    self.label_pair.config(text="Идёт подсчёт результатов...")

    # Прячем кнопки
    self.button_left.pack_forget()
    self.button_right.pack_forget()

    self.calculate_tlx()

def calculate_tlx(self):
    """
    Считаем Weighted TLX:
    Weighted_TLX = sum(Score_i * Weight_i) / sum(Weight_i)
    Если все Weight_i=0 (например, пользователь что-то не выбрал?),
    делим на 1.
    """
    total_weight = sum(self.weights)
    if total_weight == 0:
        total_weight = 1 # чтобы не было деления на ноль

    weighted_sum = 0
    for i, score_value in enumerate(self.scores):

```

```

        weighted_sum += score_value * self.weights[i]

    weighted_tlx = weighted_sum / total_weight

    self.user_id = self.master.window_manager.current_user_id

    self.services.test.save_nasa_tlx_result(
        user_id=self.user_id,
        exercise_name=self.exercise_name or "Unknown",
        task_number=self.task_number or "0",
        mental_demand=self.scores[0],
        physical_demand=self.scores[1],
        temporal_demand=self.scores[2],
        performance=self.scores[3],
        effort=self.scores[4],
        frustration=self.scores[5],
        weight_mental=self.weights[0],
        weight_physical=self.weights[1],
        weight_temporal=self.weights[2],
        weight_performance=self.weights[3],
        weight_effort=self.weights[4],
        weight_frustration=self.weights[5],
        weighted_tlx=weighted_tlx
    )

    text_result = (
        "Результаты NASA-TLX:\n\n"
        f"Ваши оценки (0-100) по шкалам:\n"
        f"Mental: {self.scores[0]}, Physical: {self.scores[1]},
Temporal: {self.scores[2]},\n"
        f"Performance: {self.scores[3]}, Effort: {self.scores[4]},
Frustration: {self.scores[5]}\n\n"
        f"Вес каждой шкалы (0..5): {self.weights}\n\n"
        f"Итоговый Weighted TLX = {weighted_tlx:.1f} (из 100)\n"
    )

    self.label_pair.config(text=text_result)

    # Кнопка завершения
    finish_btn = tk.Button(
        self,
        text="Завершить",
        command=lambda:
self.master.window_manager.show_frame("TestCompletionWindow"),
        font=("Arial", 14),
        bg="#0047ab",
        fg="white",
        activebackground="#0051c7",
        activeforeground="white"
    )
    finish_btn.pack(pady=30)

```


12.4.1. Внешний вид кадра тестирования NASA-TLX

Эргономическая оценка кабины самолета

Назад

МАИ

Инструкция к тесту

Тест NASA-TLX (Task Load Index)

Оцените субъективную нагрузку по 6 шкалам...

Начать тест

☐ Я ознакомлен с инструкцией и готов приступить

Рисунок 16 – Внешний вид кадра инструкции выполнения NASA-TLX теста

На данном кадре представлена краткая инструкция, поясняющая основные этапы и правила прохождения теста.

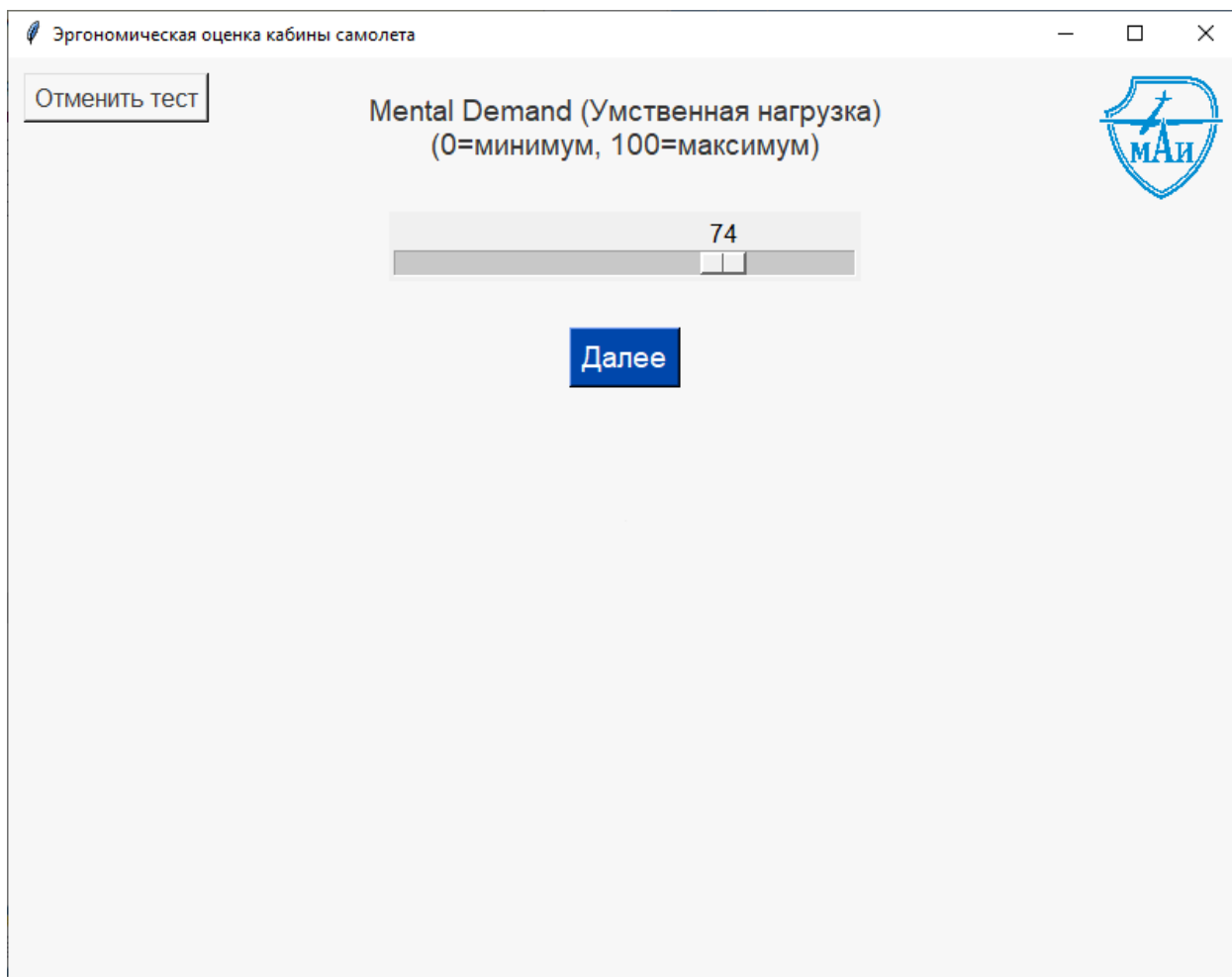


Рисунок 17 – Внешний вид кадра сбора метрик для NASA-TLX

На представленном кадре пользователь видит форму для сбора субъективных оценок рабочей нагрузки NASA-TLX. Каждая шкала представлена горизонтальным ползунком (скролл-баром), позволяющим устанавливать значение от 0 (минимальная нагрузка) до 100 (максимальная). Участнику предстоит оценить различные аспекты нагрузки (ментальные, физические, временные и др.), перемещая ползунки по соответствующим шкалам.

Собранные данные в дальнейшем анализируются исследователями, чтобы определить, какие факторы наиболее существенно влияют на общую рабочую нагрузку.

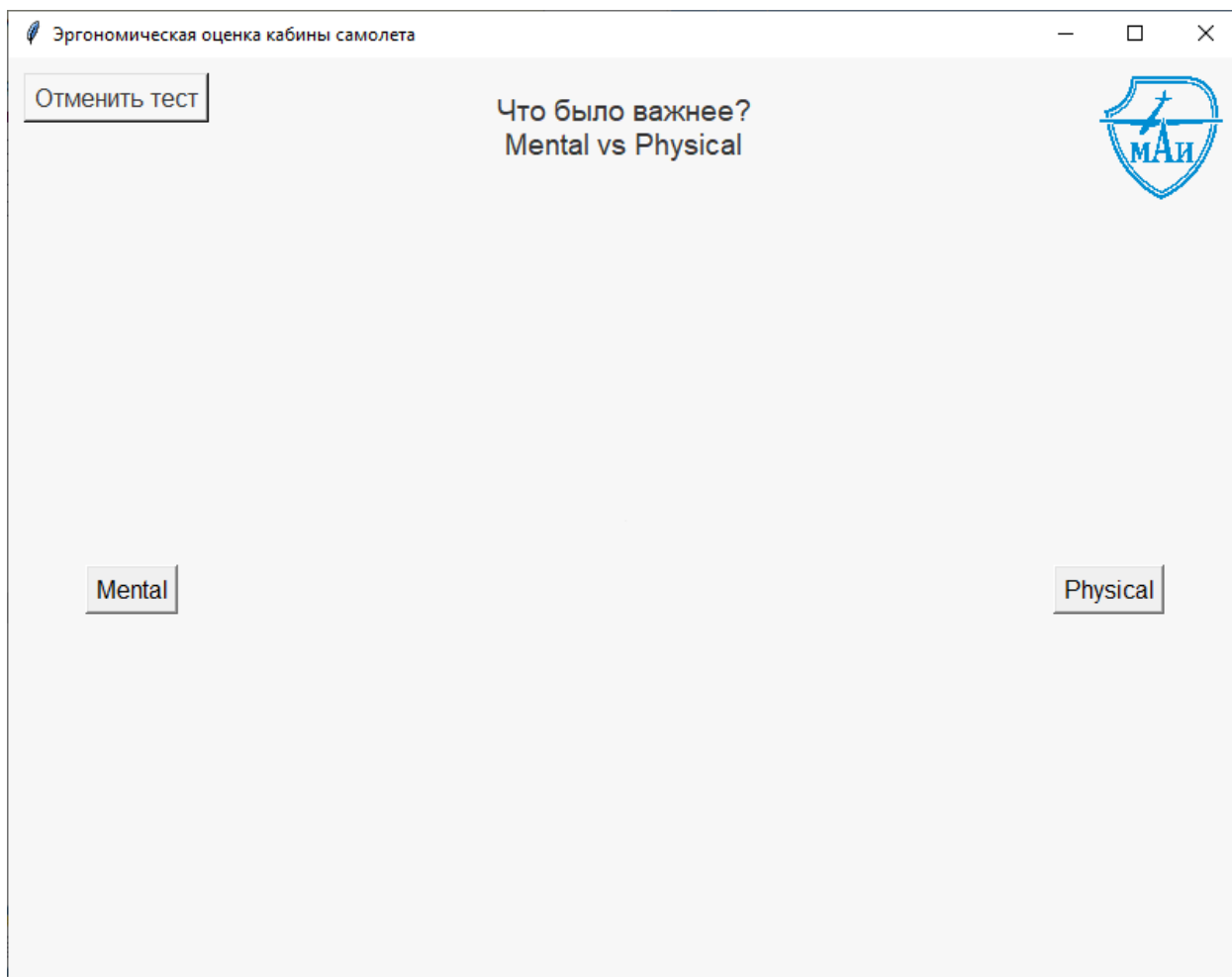


Рисунок 18 – Внешний вид кадра парных сравнений метрик NASA-TLX

На данном этапе пользователь завершил ввод субъективных оценок по всем шести шкалам NASA-TLX, включая умственную нагрузку, физическую нагрузку, временные ограничения, усилия, производительность и уровень стресса. После этого программа переходит к процессу парных сравнений, в котором пользователю последовательно предлагаются пары шкал, чтобы определить, какая из них кажется ему более важной.

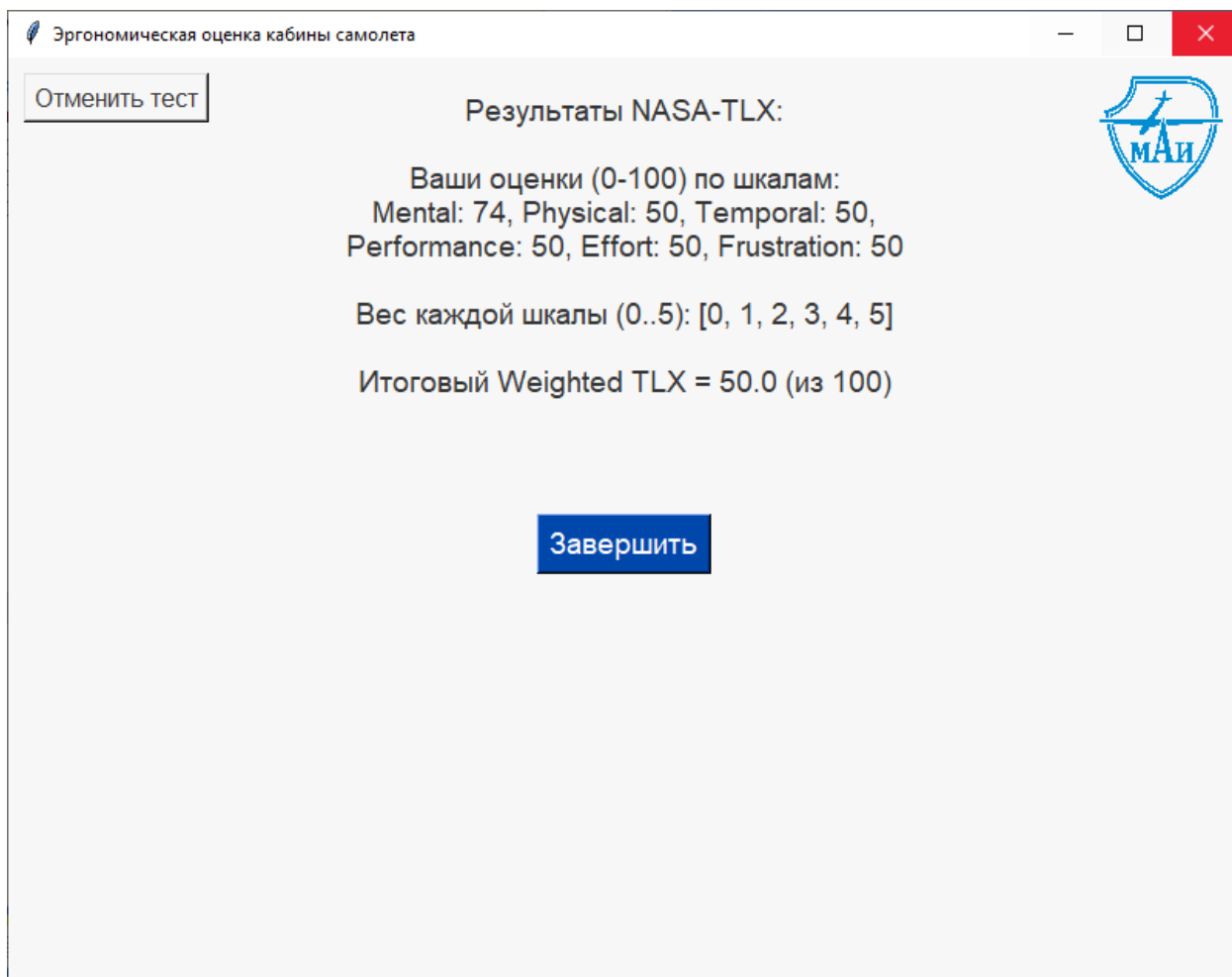


Рисунок 19 – Внешний вид кадра общих результатов

На данном кадре представлен экран с итоговыми результатами теста NASA-TLX. Здесь пользователь может увидеть рассчитанный взвешенный показатель нагрузки (**Weighted TLX**), который учитывает его субъективные оценки и индивидуальные предпочтения в важности различных шкал. Также отображаются отдельные значения по каждой из шести шкал: умственная нагрузка, физическая нагрузка, временные ограничения, усилия, производительность и уровень стресса.

13. Тестирование программы

15.1. Сценарии тестирования

Программа была протестирована на всех этапах работы для подтверждения её функциональности, точности расчётов и удобства использования. Тестирование включало проверки авторизации, выполнения тестов PVT и NASA-TLX. Каждый этап тестирования включал сценарии для проверки типичных и граничных случаев, чтобы удостовериться в надёжности приложения.

Авторизация пользователя проверялась на предмет корректной обработки логина и пароля. Сценарии включали успешный вход с правильными данными и ввод неверного логина или пароля, чтобы убедиться, что система корректно отклоняет неподходящие запросы и выводит соответствующее сообщение об ошибке.

Проведение PVT тестировалось путём измерения времени реакции пользователя на визуальные сигналы. В ходе тестирования фиксировались результаты для каждого раунда. Проверялась точность измерений, их корректная запись в базу данных и последующий доступ к сохранённым данным. Это позволило подтвердить, что время реакции фиксируется с высокой точностью и может быть использовано для анализа.

NASA-TLX был протестирован на всех стадиях: от ввода оценок по шкалам до выполнения парных сравнений и расчёта итогового показателя нагрузки. Тестирование подтвердило, что программа правильно рассчитывает Weighted TLX и корректно сохраняет данные в базу для последующего анализа. Также была проверена правильность обработки данных при граничных значениях шкал (например, 0 или 100).

1. Регистрация пользователя:

Для тестирования рассмотрим несколько сценариев:

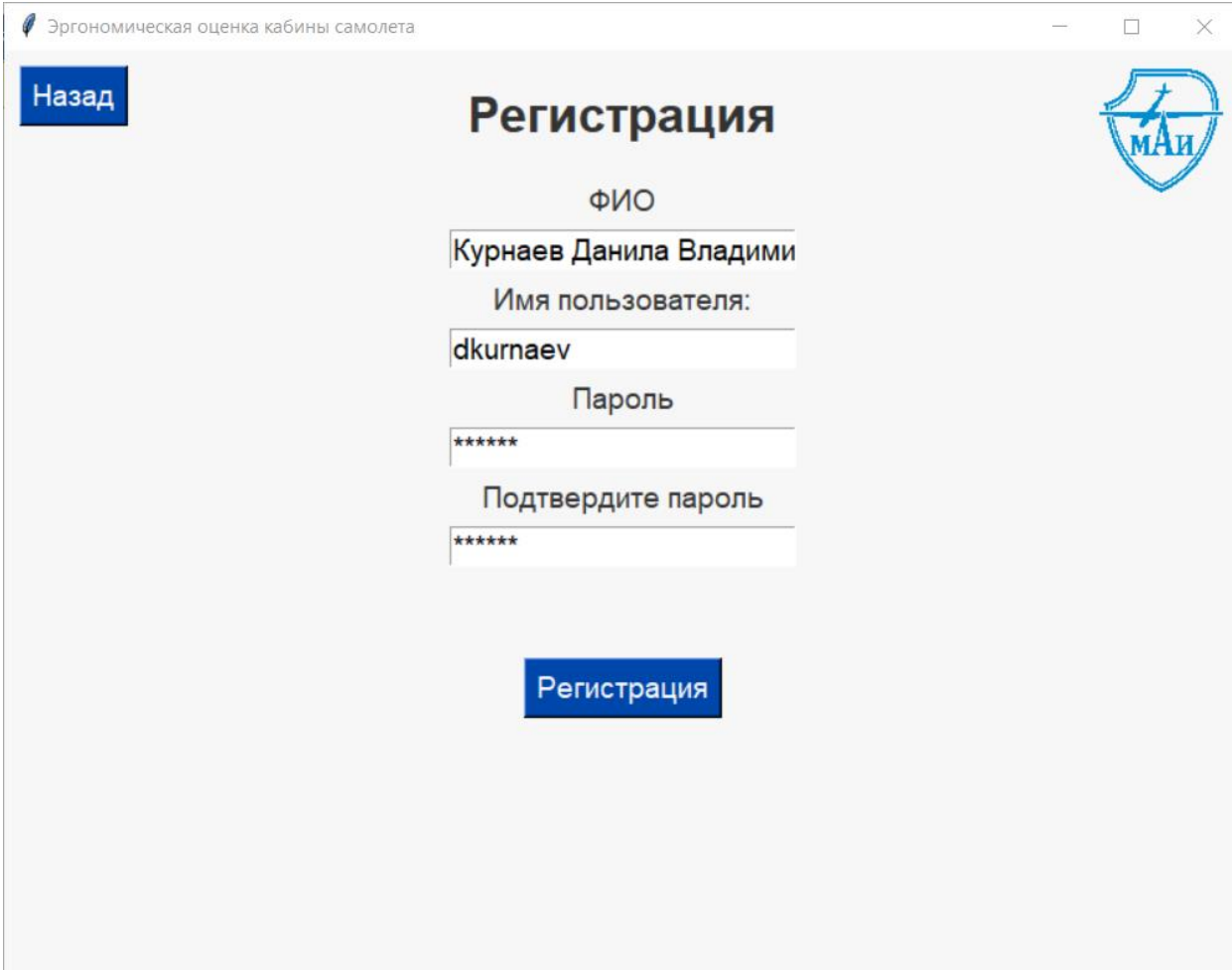
- Проверка регистрации с корректными данными.

При проведении первого теста будем исходить из того, что мы знаем ограничения системы. Будем передавать данные, которые удовлетворяют условиям проверок:

Набор данных №1:

- ФИО: Курнаев Данила Владимирович
- Имя пользователя: dkurnaev
- Пароль: A1b2C3

После ввода «идеальных» данных уведомлений об ошибках отсутствуют.



The screenshot shows a web browser window with the title "Эргономическая оценка кабины самолета". The page is titled "Регистрация" (Registration). In the top left corner, there is a blue button labeled "Назад" (Back). In the top right corner, there is a logo of the MAI (Moscow Aviation Institute). The registration form consists of the following fields and labels:

- ФИО (Full Name): Курнаев Данила Владими
- Имя пользователя: (Username): dkurnaev
- Пароль (Password): *****
- Подтвердите пароль (Confirm Password): *****

At the bottom of the form, there is a blue button labeled "Регистрация" (Registration).

Рисунок 20 – Внешний вид кадра регистрации с «идеальными» данными

В консоли приложения вывелась отладочная информация о создании пользователя: [DEBUG] Регистрация прошла успешно, user_id=3

	id	name	login	password
1	1	1234	1234	1234
2	2	Иванов Иван Ива...	van_ivan	1234
3	3	Курнаев Данила ...	dkurnaev	A1b2C3

Рисунок 21 – Данные пользователей из базы данных

В базе данных появились данные пользователя. Тест можно считать пройденным успешно.

- Проверка регистрации с некорректными данными.

При проведении следующего теста, будем исходить из того, что мы знаем ограничения системы. Будем передавать данные, которые не удовлетворяют условиям проверок:

Набор данных №1:
<ul style="list-style-type: none"> • ФИО: Курнаев Данила Владимирович • Имя пользователя: <u>err</u> • Пароль: 12345
Набор данных №2:
<ul style="list-style-type: none"> • ФИО: Курнаев Данила Владимирович • Имя пользователя: err0r • Пароль: <u>123</u>
Набор данных №3:
<ul style="list-style-type: none"> • ФИО: Курнаев Данила Владимирович • Имя пользователя: err0r • Пароль: <u>12345</u> • Повторный ввод пароля: <u>1234</u>

Примеры уведомлений об ошибках валидации указаны на Рисунках 22-24.

Эргономическая оценка кабины самолета

Назад

Регистрация

ФИО

Курнаев Данила Владими

Имя пользователя:

err

Пароль

Подтвердите пароль

Логин должен быть от 3 до 50 символов

Регистрация





Рисунок 22 – Внешний вид кадра регистрации с набором данных №1

Эргономическая оценка кабины самолета

Назад

Регистрация



ФИО

Курнаев Данила Владими

Имя пользователя:

error

Пароль

Подтвердите пароль

Пароль слишком короткий!

Регистрация

Рисунок 23 – Внешний вид кадра регистрации с набором данных №2

Рисунок 24 – Внешний вид кадра регистрации с набором данных №3

В консоли приложения не вывелась отладочная информация о создании пользователя.

	id	name	login	password
1	1	1234	1234	1234
2	2	Иванов Иван Иванович	van_ivan	1234
3	3	Курнаев Данила Владимирович	dkurnaev	A1b2C3

Рисунок 25 – Данные пользователей из базы данных

В базе данных не появились данные пользователя. Тест можно считать пройденным успешно.

2. Авторизация пользователя:

Для тестирования рассмотрим несколько сценариев:

- Попытка входа с неверными данными.

Для тестирования будем использовать такой набор данных:

Набор данных №1:
<ul style="list-style-type: none"> Имя пользователя: <u>error</u> Пароль: <u>12345</u>
Набор данных №2:
<ul style="list-style-type: none"> Имя пользователя: <u>dkurnaev</u> Пароль: <u>A1b2C3</u>

На Рисунках 21, 24 показаны пользователи, данных которых существуют в базе данных.

Эргономическая оценка кабины самолета

Назад

Вход

Имя пользователя:

error

Пароль:

Войти

Неверный логин или пароль!

МАИ

Рисунок 26 – Внешний вид кадра неудачной авторизации с набором данных №1

В консоли приложения не вывелась отладочная информация о создании пользователя. В базе данных отсутствует пользователь с набором данных №1. Тест можно считать пройденным успешно.

- Проверка входа с корректными данными.

Рисунок 27 – Внешний вид кадра авторизации с набором данных №2

В консоли приложения вывелась отладочная информация о создании пользователя: [DEBUG] Авторизация прошла успешно. user_id=3

	id	name	login	password
1	1	1234	1234	1234
2	2	Иванов Иван Ива...	van_ivan	1234
3	3	Курнаев Данила ...	dkurnaev	A1b2C3

Рисунок 28 – Данные пользователей из базы данных

В базе данных присутствует пользователь с набором данных №2. Тест можно считать пройденным успешно. Тест можно считать пройденным успешно.

3. Проведение PVT:

Для проверки работы функционала тестирования PVT проведем несколько этапов:

- Тестирование времени реакции на каждом этапе.
- Проверка корректности сохранения результатов в базе данных.

На первом этапе пройдем все 5 раундов и зафиксируем значения, который приложение отобразит на экране.

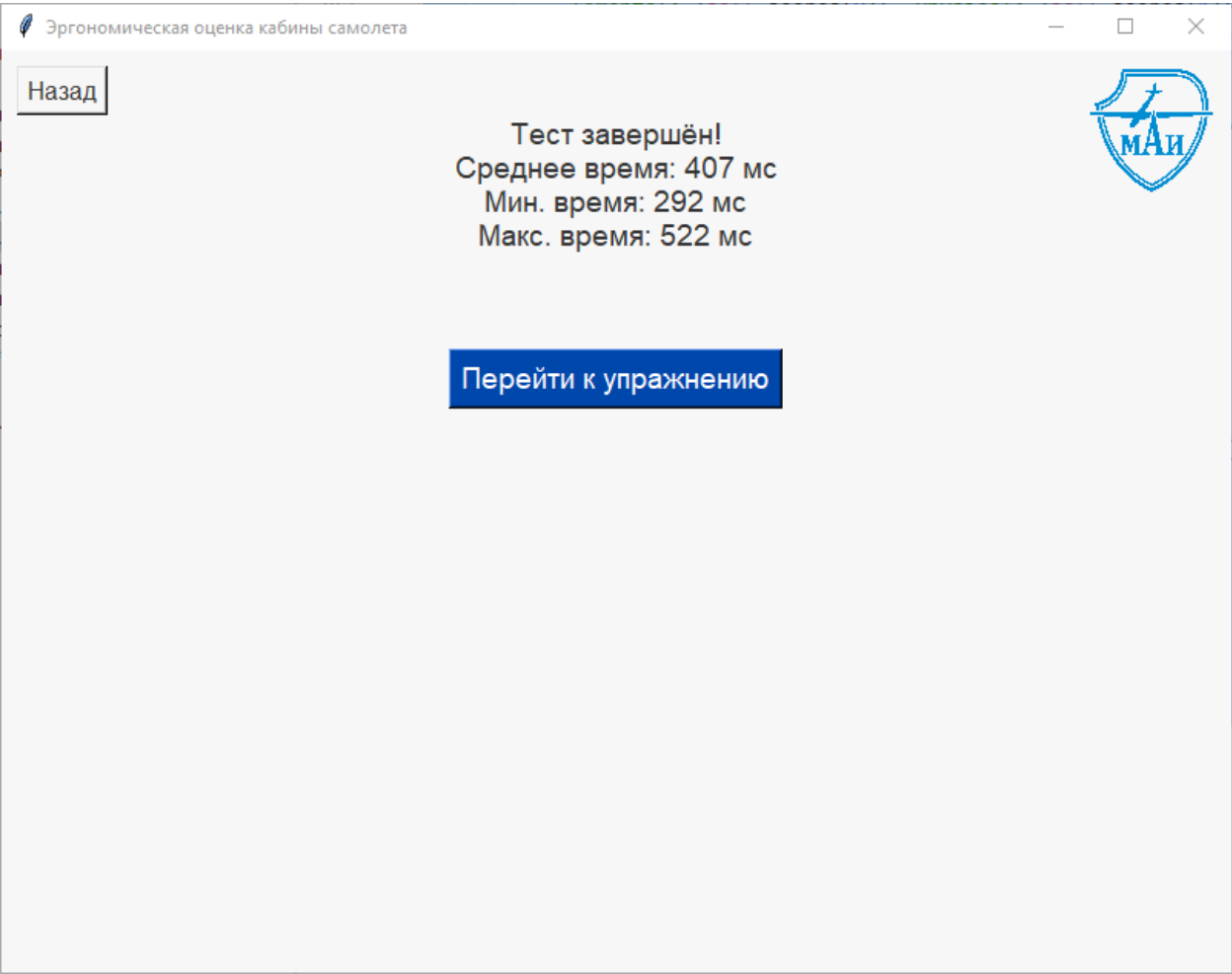


Рисунок 28 – Внешний вид кадра общих результатов после прохождения PVT

Далее проверим значения последнего PVT теста от пользователя в базе данных.

	id	user_id	exercis...	task_nu...	type_test	round_i...	reactio...	timestamp
1	37	1	Взлет	2	before	4	0.40744085884...	2024-12-26 21:55:55.241334
2	36	1	Взлет	2	before	3	0.29230284690...	2024-12-26 21:55:51.612499
3	35	1	Взлет	2	before	2	0.52189087867...	2024-12-26 21:55:48.012602
4	34	1	Взлет	2	before	1	0.38600277900...	2024-12-26 21:55:43.562511

Рисунок 28 – Результаты тестирования методом PVT в базе данных

Исходя из того, что данные, собранные в результате теста, совпадают с данными, сохранёнными в базе данных. Тест можно считать пройденным успешно.

14. Организационно-экономическая часть

15.1. Введение

На современном этапе развития авиационной промышленности особое внимание уделяется обеспечению безопасных условий труда пилотов, что имеет важное значение для повышения эффективности их деятельности и снижения рисков, связанных с человеческим фактором. Одним из перспективных направлений в данной области является разработка эргономических методов оценки условий труда в кабине самолета, которые позволяют оптимизировать рабочее пространство пилота и улучшить взаимодействие с бортовыми системами.

В данной дипломной работе рассматривается система эргономической оценки кабины самолета на основе теста психомоторной бдительности (PVT) и методики NASA-TLX. Применение этих методов позволит количественно оценить психофизиологическую нагрузку пилотов, выявить наиболее критические факторы, влияющие на их производительность, и предложить рекомендации по снижению нагрузки и повышению эргономичности рабочей среды.

15.2. Расчет времени на проведение работ

Проведем расчет времени выполнения дипломной работы, построив план-график работ и представим его в виде таблицы:

Таблица 1.2-1 План-график работ

п/п	Название работ	Начало работ (дд.мм.гг)	Окончание работ (дд.мм.гг)	Продолжительность (раб. дни)	Исполнители	Трудоемкость (чел. дни)
1	Ознакомление с заданием и исходными данными	02.09.24	03.09.24	2	Программист 3 категории	2
2	Сбор теоретических данных по методам	04.09.24	06.09.24	3	Программист 3 категории	3

	PVT и NASA-TLX					
3	Анализ научных публикаций и существующих подходов к эргономической оценке	09.09.24	13.09.24	5	Программист 3 категории	5
4	Разработка требований к системе эргономической оценки	16.09.24	18.09.24	3	Ведущий программист, Программист 3 категории	6
5	Подготовка программной среды для проведения тестов PVT	19.09.24	26.09.24	6	Ведущий программист, Программист 3 категории	12
6	Разработка интерфейса для проведения оценки по методике NASA-TLX	27.09.24	03.10.24	5	Ведущий программист, Программист 3 категории	10
7	Подготовка тестовых сценариев для проведения PVT и NASA-TLX	04.10.24	08.10.24	3	Программист 3 категории	3
8	Разработка модуля обработки и анализа результатов тестов	09.10.24	18.10.24	8	Ведущий программист, Программист 3 категории	16
9	Тестирование и доработка модуля обработки результатов	21.10.24	25.10.24	5	Ведущий программист, Программист 3 категории	10
10	Интеграция модуля PVT и NASA-TLX с системой визуализации результатов	28.10.24	05.11.24	7	Ведущий программист, Программист 3 категории	14
11	Проведение тестирования	06.11.24	12.11.24	5	Ведущий программист,	10

	разработанной системы на стенде				Программист 3 категории	
12	Анализ полученных результатов и выявление критических факторов эргономичности	13.11.24	18.11.24	4	Ведущий программист, Программист 3 категории	8
13	Подготовка отчета по результатам оценки и рекомендаций по улучшению эргономики кабины самолета	19.11.24	27.11.24	7	Программист 3 категории, Ведущий программист	14
14	Оформление итоговой документации и завершение проекта	28.11.24	29.11.24	2	Ведущий программист, Программист 3 категории,	4
Итого			65 рабочих дней		117 человеко-дней	

Для построение сетевого графика понадобятся данные состава и последовательности работ в виде начального и конечного события. Продолжительность дней возьмем из Таблицы 1.2–1 План-график работ.

Таблица 1.2-2 Состав и последовательность работ

п/п	Название работ	Начальное событие	Конечное событие	Продолжительность (раб. дни)	Исполнители
1	Ознакомление с заданием и исходными данными	1	2	2	Программист 3 категории
2	Сбор теоретических данных по методам PVT и NASA-TLX	2	3	3	Программист 3 категории
3	Анализ научных публикаций и	3	4	5	Программист 3 категории

	существующих подходов к эргономической оценке				
4	Разработка требований к системе эргономической оценки	4	5	3	Ведущий программист, Программист 3 категории
5	Подготовка программной среды для проведения тестов PVT	5	6	6	Ведущий программист, Программист 3 категории
6	Разработка интерфейса для проведения оценки по методике NASA-TLX	6	7	5	Ведущий программист, Программист 3 категории
7	Подготовка тестовых сценариев для проведения PVT и NASA-TLX	7	8	3	Программист 3 категории
8	Разработка модуля обработки и анализа результатов тестов	8	9	8	Ведущий программист, Программист 3 категории
9	Тестирование и доработка модуля обработки результатов	9	10	5	Ведущий программист, Программист 3 категории
10	Интеграция модуля PVT и NASA-TLX с системой визуализации результатов	10	11	7	Ведущий программист, Программист 3 категории
11	Проведение тестирования	11	12	5	Ведущий программист,

	разработанной системы на стенде				Программист 3 категории
12	Анализ полученных результатов и выявление критических факторов эргономичности	12	13	4	Ведущий программист, Программист 3 категории
13	Подготовка отчета по результатам оценки и рекомендаций по улучшению эргономики кабины самолета	13	14	7	Программист 3 категории, Ведущий программист
14	Оформление итоговой документации и завершение проекта	14	15	2	Ведущий программист, Программист 3 категории,

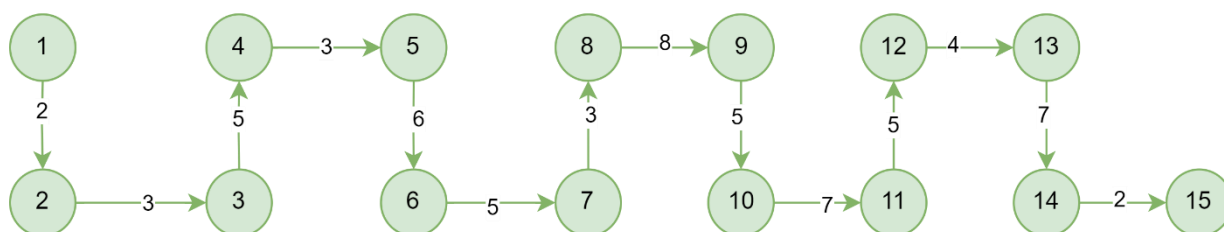


Рисунок 1.2-1 Сетевой график

Из Таблица 1.2–1 План-график работ видно, что продолжительность работ составит 65 рабочих дней. В работе задействованы:

- ведущий программист
- программист 3 категории;

За 65 рабочих дней их трудоемкость составит 117 человеко-дней. Из них приходится:

- 65 рабочих дней на программиста 3 категории;

- 52 рабочих дней на ведущий программист;

15.3. Расчет заработной платы

Заработная плата i -го специалиста рассчитывается по следующей формуле:

$$ЗП_i = ОЗП_i + ДЗП_i \quad (2)$$

где $ОЗП_i$ – основная заработная плата i -го специалиста, $ДЗП_i$ – дополнительная заработная плата i -го специалиста.

Основная заработная плата для специалистов считается по формуле (2). Заработная плата для специалистов разной квалификации будет отличаться.

$$ОЗП_i = СЧС_i \cdot t_i \quad (3)$$

где $СЧС_i$ – средняя ставка i -го специалиста за час работы, t_i – количество часов, отработанных i -ым специалистом.

Для расчета количества часов, отработанных i -ым специалистом, воспользуемся формулой (3):

$$t_i = N_i \cdot t_{\text{рабочего дня}} \quad (4)$$

где N_i – количество рабочих дней i -го специалиста, $t_{\text{рабочего дня}}$ – продолжительность рабочего дня.

Формула для расчета дополнительной заработной платы:

$$ДЗП_i = ОЗП_i \cdot 0,2 \quad (5)$$

Средняя ставка сотрудников:

- ведущего программист – 1710 рублей в час,
- программиста 3 категории – 826 рублей в час,

Рассчитаем основную заработную плату каждого сотрудника:

- $ОЗП_{\text{ведущего программист}} = 1710 \frac{\text{руб}}{\text{час}} \cdot 52 \text{ дней} \cdot 8 \text{ часов} = 711\,360 \text{ рубля}$
- $ОЗП_{\text{программиста 3 категории}} = 826 \frac{\text{руб}}{\text{час}} \cdot 65 \text{ дней} \cdot 8 \text{ часов} = 429\,520 \text{ рубля}$

Пользуясь формулой (5), найдем ДЗП каждого сотрудника:

- $\text{ДЗП}_{\text{ведущего программист}} = 711\,360 \text{ рублей} \cdot 0.2 = 142\,272 \text{ рублей}$
- $\text{ДЗП}_{\text{программиста 3 категории}} = 429\,520 \text{ рублей} \cdot 0.2 = 85\,904 \text{ рублей}$

Пользуясь формулой (1), найдем ЗП каждого сотрудника:

- $\text{ЗП}_{\text{ведущего программист}} = 711\,360 \text{ рублей} + 142\,272 \text{ рублей} = 853\,632 \text{ рублей}$
- $\text{ЗП}_{\text{программиста 3 категории}} = 429\,520 \text{ рублей} + 85\,904 \text{ рублей} = 515\,424 \text{ рублей}$

Таблица 1.3–1 Общие затраты на оплату труда

Должность	ОЗП, руб.	ДЗП, руб.	ЗП, руб.	ФОТ, руб.
Ведущего программист	711 360	142 272	853 632	1 369 056
Программиста 3 категории	429 520	85 904	515 424	

15.4. Расходные материалы

Таблица 1.4–1 Расходные материалы

п/п	Виды расходных материалов	Расход (кол-во)	Цена (ед.) Руб.	Общая цена
1	Канцелярский набор для рабочего стола	2	500	1 000
2	Бумага для печати формата А4	2	500	1 000
Итого:				2 000 рублей

15.5. Покупные комплектующие изделия

Расходы на ПЭВМ сотрудников в сборке представлены в таблице:

Таблица 1.4-1 Расходы на ПЭВМ сотрудников в сборке

п/п	Название ПКИ	Количество, шт.	Цена за 1 шт., руб.	Сумма, руб.
ПЭВМ ведущего программиста				
1	Корпус: Corsair 4000D Airflow	1	7 000	7 000

2	Материнская плата: ASUS ROG STRIX X570-E	1	18 000	18 000
3	Процессор: AMD Ryzen 9 5900X	1	28 000	28 000
4	Оперативная память: G.Skill Ripjaws 16GB DDR4	2	5 000	10 000
5	Жесткий диск: Samsung 980 Pro 2TB NVMe	1	15 000	15 000
6	Видеокарта: NVIDIA GeForce RTX 3080 Ti	1	70 000	70 000
7	Блок питания: be quiet! Straight Power 11 850W	1	10 000	10 000
8	Монитор: Dell UltraSharp U2723QE 27"	2	25 000	50 000
9	Компьютерная мышь: Logitech MX Master 3	1	5 000	5 000
Итого:				213 000
ПЭВМ программиста 3 категории				
1	Корпус: DeepCool Matrexx 55 Mesh	1	4 500	4 500
2	Материнская плата: MSI MAG B660M Mortar	1	12 000	12 000
3	Процессор: Intel Core i5-12600KF	1	20 000	20 000
4	Оперативная память: Kingston Fury 16GB DDR4	2	4 500	9 000
5	Жесткий диск: WD Black SN770 1TB NVMe	1	8 000	8 000
6	Видеокарта: AMD Radeon RX 6800	1	45 000	45 000
7	Блок питания: Thermaltake Toughpower GF1 750W	1	7 000	7 000
8	Монитор: ASUS ProArt Display PA248QV 24"	2	18 000	36 000
9	Компьютерная мышь: Razer DeathAdder V2	1	3 000	3 000
Итого:				144 500
Количество ПЭВМ, шт:		2	Сумма:	357 500

15.6. Электроэнергия на технологические цели

Расходы на электроэнергию рассчитываются по формуле:

$$C_{\text{эл}} = P \cdot T \cdot Z \quad (1)$$

где P – суммарная мощность оборудования, T – общее время работы оборудования, Z – цена одного кВт в ч.

$$P_{\text{ПЭВМ}} = 0.350 \text{ кВт}, T_{\text{ПЭВМ1}} = 520 \text{ ч}, T_{\text{ПЭВМ2}} = 416 \text{ ч}.$$

Суммарное количество часов работы специалистов ПЭВМ составляет 880 часов.

В соответствии с ПАО «Мосэнергосбыт» от 2024 года, тариф на электроэнергию для предприятий составляет 6,08 кВт · ч.

Затраты на электроэнергию будут следующими: 182 126

$$\begin{aligned} C_{\text{эл}} &= (P_{\text{ПЭВМ}} \cdot T_{\text{ПЭВМ1}} + P_{\text{ПЭВМ}} \cdot T_{\text{ПЭВМ2}}) \cdot Z = (0.35 \cdot 520 + 0.35 \cdot 416) \cdot 6.08 \\ &= 1992 \text{ рубля} \end{aligned}$$

15.7. Отчисления во внебюджетные фонды

Отчисления во внебюджетные фонды представляют из себя обязательные страховые взносы, которые отчисляются в Пенсионный фонд (ПФР), Медицинский фонд (ФФОМС) и Фонд социального страхования (ФСС). Отчисления составляют 30% от ФОТ. Также 0,2% от ФОТ составляет страхование от профзаболеваний и несчастных случаев, которые тоже входят в отчисления во внебюджетные фонды.

Проведём расчёт социальных отчислений по формуле:

$$\text{ВФ} = (30\% + 0,2\%) \times \text{ФОТ}. \quad (6)$$

Тогда получим:

$$\text{ВФ} = 30,2\% \times 1\,369\,056 \text{ руб} = 413\,455 \text{ руб}.$$

15.8. Отчисления на накладные расходы

Согласно приказу Минстроя РФ от 04.08.2020 №421/ПР п.18 накладными расходами являются общепроизводственные и общехозяйственные расходы, то есть это расходы, которые прямо не связаны с производством продукции. Они составят 30 % от суммы затрат на оплату труда, а именно 410 717 рублей.

15.9. Амортизационные отчисления

Амортизационные отчисления будем рассчитывать с помощью линейного способа по следующей формуле:

$$AO = \sum_{i=1}^k \frac{C_i}{T_i} \times T_p, (7)$$

где C_i – первоначальная стоимость i -го оборудования в рублях; T_i – срок полезного использования i -го оборудования в месяцах; T_p – срок работы оборудования в проекте (в данном случае 3 месяца).

Таблица 1.9-1 Описание КПИ

Оборудование	Срок полезного использования, мес	Количество, шт	Стоимость, руб
ПЭВМ ведущего программиста	36	1	213 000
ПЭВМ программиста 3 категории	36	1	144 500

$$AO = \left(\frac{213\,000 + 144\,500}{36} \right) \times 3 \text{ мес} = 29\,791 \text{ рублей}$$

15.10. Смета затрат

Смета затрат представлена в Таблица 1.10-1 Смета затрат:

Таблица 1.10-1 Смета затрат

п/п	Статья расходов	Величина, руб.	Удельный вес в общей сумме затрат, %
1	Расходные материалы	2 000	0,1
2	Электроэнергия на технологические цели	1 992	0,1
3	Заработная плата	1 369 056	61,5
4	Внебюджетные фонды	413 455	18,6
5	Накладные расходы	410 717	18,4

6	Амортизационные отчисления	29 791	1,3
	Итого:	2 227 011	100,00

Из Таблица 1.10-1 Смета затрат следует, что себестоимость проведения работ составляет **2 227 011 рубля**. Основная доля затрат приходится на **заработную плату специалистов**, которая составляет **61,5%** от общей суммы. Это обусловлено необходимостью привлечения высококвалифицированных специалистов, чей труд требует достойной оплаты. Помимо этого, значительные затраты связаны с использованием качественного оборудования, что способствует повышению эффективности работ, а также обеспечивает долговечность и надежность конечного продукта.

15.11.Вывод

Разработанная система эргономической оценки кабины самолета на основе теста психомоторной бдительности (PVT) и методики NASA-TLX обладает следующими преимуществами:

- **Оптимизация условий труда пилотов** позволяет снизить психофизиологическую нагрузку за счет выявления и устранения эргономических недостатков в кабине самолета. Это способствует повышению производительности пилотов и сокращению времени на выполнение задач на **10–15%**. Если в среднем стоимость работы экипажа составляет **3 000 000 рублей в год**, улучшение условий труда позволит сэкономить **300 000–450 000 рублей** ежегодно.
- **Снижение рисков, связанных с человеческим фактором**, достигается за счет уменьшения утомляемости и повышения концентрации пилотов на критически важных задачах. Это способствует снижению числа инцидентов и аварий на **8–12%**. При страховых расходах в размере **12 000 000 рублей в год**, экономия может составить **960 000–1 440 000 рублей** ежегодно.

- **Повышение долговечности оборудования** благодаря более комфортным условиям работы пилотов, что позволяет сократить износ систем и необходимость частого обслуживания на **5–7%**. Если годовые затраты на обслуживание составляют **4 000 000 рублей**, экономия может достигнуть **200 000–280 000 рублей**.

Таким образом, внедрение системы эргономической оценки не только способствует обеспечению безопасных условий труда, но и позволяет достичь значительного экономического эффекта за счет оптимизации времени работы, снижения числа инцидентов и уменьшения затрат на обслуживание оборудования.

15. Охрана труда и окружающей среды

Обеспечение безопасных условий труда при разработке системы эргономической оценки кабины самолета.

данной главе рассматриваются условия труда специалистов, участвующих в разработке системы эргономической оценки кабины самолета, основанной на тестах психомоторной бдительности (PVT) и методике NASA-TLX.

Процесс разработки включает анализ, проектирование, тестирование и внедрение компонентов системы, включая программное обеспечение, обрабатывающее данные тестов и оценивающее эргономические параметры кабины. Учитывая характер работы, специалисты работают в закрытом помещении, в положении сидя за рабочими местами, используя персональные компьютеры. Для обеспечения их безопасности и комфорта необходимо создать условия труда, соответствующие современным стандартам.

15.1. Описание выполняемых работ

В ходе выполнения данной работы был проведен анализ существующих аналогов систем, предназначенных для эргономической оценки.

Подготовлены:

- 1) список параметров для оценки психомоторной бдительности и уровня нагрузки на операторов;
- 2) перечень требований к программному обеспечению, обеспечивающему обработку результатов тестов PVT и NASA-TLX;
- 3) алгоритмы анализа и представления данных для их дальнейшей интерпретации.

Данный вид работ по уровню энергозатрат относится к категории Ia ввиду того, что работы производятся в положении сидя и с интенсивностью энергозатрат до 120 ккал/ч (до 139 Вт).

15.2. Описание рабочего помещения

Разработка проводилась в офисном помещении. Его характеристики указаны в таблице:

Таблица 1. Характеристики рабочего помещения

Длина, м	Ширина, м	Высота, м	Количество рабочих мест, шт	Количество окон, шт
22	15	3	15	5

Площадь рабочего помещения составляет 330 м², а объём помещения составляет 990 м³.

Характеристики окон представлены в таблице:

Таблица 2. Характеристики окон в рабочем помещении

Размеры оконных проёмов, мм		Размеры окон, мм	
Высота	Ширина	Высота	Ширина
1500	2000	1470	1970

Площадь пола и объём воздуха, приходящиеся на одного работающего, составляют 22 м² и 66 м³.

Площадь пола и объём воздуха, приходящиеся на одного работающего при разработке системы эргономической оценки, составляют **22 м²** и **66 м³** соответственно.

Согласно пункту 5.1 раздела «Требования к производственным зданиям, помещениям и сооружениям» документа **СП 2.2.3670-20**, минимальный объём помещений на одного работника, в зависимости от категории энергозатрат Ia, должен составлять не менее **15 м³**. В соответствии с пунктом 5.2 того же документа, минимальная площадь на одного работника должна быть не менее **4,5 м²**.

Рабочее помещение полностью соответствует указанным нормативным требованиям, обеспечивая безопасные условия труда для специалистов.

15.3. Описание рабочего места

Разработка системы эргономической оценки проводилась на рабочем месте. Его характеристики приведены в таблице:

Таблица 3. Характеристики рабочего места

Сведения	Характеристика [величина]
Тип ПЭВМ	Стационарный компьютер с системным блоком [1 штука]
Тип монитора	Жидкокристаллический [2 штуки]

Периферийные устройства	Клавиатура [1 штука]
	Компьютерная мышь [1 штука]
Площадь рабочего места	10.5 [м ²]

Фактическая площадь рабочего места равна 10.5 м², что соответствует п.249 раздела XXII «Требования к организации работ» Приложения №1 СП 2.2.3670-20: «Площадь на одно постоянное рабочее место пользователей персональных компьютеров на базе электронно-лучевой трубки, должна составлять не менее 6 м², в помещениях культурно-развлекательных учреждений, на базе плоских дискретных экранов (жидкокристаллические, плазменные) - не менее 4,5 м²».

Помещение оснащено горизонтальными пластиковыми жалюзи, размещёнными на каждом оконном проёме, что соответствует п.250 раздела XXII «Требования к организации работ» Приложения №1 СП 2.2.3670-20: «Оснащение светопроницаемых конструкций и оконных проёмов должно позволять регулировать параметры световой среды в помещении».

Эргономические характеристики стола, за которым проводилась разработка системы эргономической оценки, приведены в таблице:

Таблица 4. Эргономические характеристики рабочего стола

Высота, мм	Ширина, мм	Глубина на уровне стоп, мм	Глубина на уровне колен, мм
700	650	620	550

Работы выполнялись в положении сидя. Согласно п.6.3 раздела VI «Требования к организации технологических процессов и рабочих мест»: «На рабочем месте, предназначенном для работы в положении сидя, производственное оборудование и рабочие столы должны иметь пространство для размещения ног высотой не менее 600 мм, глубиной – не менее 450 мм на уровне колен и 600 мм на уровне стоп, шириной не менее 500 мм». Фактические эргономические характеристики полностью соответствуют данному пункту.

Фактическая площадь и характеристики рабочего места программиста-разработчика соответствуют СП 2.2.3670-20, что позволяет комфортно осуществлять рабочий процесс.

15.4. Специальная оценка условий труда

15.4.1. Шум

Оценка уровня шума представляет собой вычисление суммарного уровня звука от всех одновременно работающих источников широкополосного шума. Согласно п.37 раздела IV параграфа «Отнесение условий труда к классу (подклассу) условий труда при воздействии виброакустических факторов» приказа Минтруда России от 24.01.2014 №33 для оценки уровня шума допускается использование уровня звука, измеряемого как дБА.

Сведения об источниках шума, которые расположены в рабочем помещении, и их характеристики представлены в таблице:

Таблица 5. Сведения об источниках шума и их характеристиках

№ п/п	Наименование источника шума	Количество, шт	Уровень звука, дБА
1	Персональный компьютер с системных блоком	15	40
2	Кондиционер	2	35
3	Принтер	1	45

Для проверки соответствия фактического уровня звука требованиям Приложения №11 приказа Минтруда России от 24.01.2014 №33, необходимо рассчитать его суммарный уровень. Вышеперечисленные источники являются неравногромкими, следовательно, используется следующая формула для расчёта:

$$L_{\Sigma} = 10 \lg \sum_{i=1}^{k=1} 10^{\frac{L_i}{10}},$$

где:

L_{Σ} – суммарный уровень звука от всех одновременно работающих источников, дБА;

L_i – уровень звука, создаваемый i -ым источником, дБА;

k – число типов источников шума, шт.

Вычислим суммарный уровень звука в помещении, где проводилась разработка с описанными ранее источниками шума, по этой формуле:

$$L_{\Sigma} = 10 \lg \left(15 \times 10^{\frac{40}{10}} + 2 \times 10^{\frac{35}{10}} + 1 \times 10^{\frac{45}{10}} \right) = 52,74 \text{ [дБА]}$$

Воспользуемся выдержкой из таблицы «Отнесение условий труда по классу (подклассу) условий труда при воздействии виброакустических факторов» в Приложении №11 к Методике проведения специальной оценки условий труда, утверждённой приказом Минтруда России от 24.01.2014 №33, представленной ниже:

Таблица 6. Соответствие уровня шума классу (подклассу) условий труда

Наименование показателя, единица измерения	Класс (подкласс) условий труда					
	допустимый	вредный				опасный
	2	3.1	3.2	3.3	3.4	4
Шум, эквивалентный уровень звука, дБА	≤ 80	$> 80 - 85$	$> 85 - 95$	$> 95 - 105$	$> 105 - 115$	> 115

Суммарный уровень шума в рабочем помещении составляет 52,74 дБА, а значит он удовлетворяет требованиям, установленным в приказе Минтруда России от 24.01.2014 №33. Так как суммарный уровень шума менее 80 дБА, то класс условий труда считается допустимым. Таким образом суммарный уровень шума от всех работающих источников удовлетворяет требованиям.

15.4.2. Освещенность

В рабочем помещении применяется совмещённое освещение, представляющее собой совокупность искусственного и естественного освещений.

В помещении, где проводилась разработка ПО, применено двухстороннее естественное освещение, сведения о котором приведены в таблице.

Таблица 7. Сведения об естественном освещении в рабочем помещении

Сведения	Характеристика
Оконные проёмы	Количество: 5 шт. Размеры: <ul style="list-style-type: none"> • Высота 1500 мм; • Ширина 2000 мм.
Окна	Количество: 5 шт. Размеры: <ul style="list-style-type: none"> • Высота 1470 мм; • Ширина 1970 мм.

Фонари в помещении отсутствуют. На каждом оконном проёме установлены пластиковые жалюзи.

Помещение оборудовано 40 потолочными светильниками с люминесцентными лампами. Один светильник включает в себя 4 лампы. Характеристики ламп указаны в таблице.

Таблица 8. Технические характеристики люминесцентной лампы

Характеристика	Значение
Мощность	18 [Вт]
Температура света	5000 [К]
Световой поток	1350 [лм]
Длина	590 [мм]
Диаметр	26 [мм]

Светильники местного освещения не предусмотрены.

Проведём оценку соответствия искусственного освещения требованиям СанПиН 1.2.3685-21 «Гигиенические нормативы и требования к обеспечению безопасности и (или) безвредности для человека факторов среды обитания». Для этого воспользуемся формулой расчёта фактической освещённости, создаваемой системой общего искусственного освещения:

$$E = \frac{F_{\text{л}} \times N \times n}{S \times x},$$

где:

$F_{\text{л}}$ – световой поток одной лампы, лм;

N – количество светильников в системе искусственного освещения;

n – число ламп в одной светильнике;

S – площадь пола помещения, м²;

x – эмпирический коэффициент ($x = 1.64 \div 2.36$).

Примем x равным 2, тогда значение фактической освещённости будет следующим:

$$E = \frac{1350 \text{ лм} \times 40 \times 4}{330 \text{ м}^2 \times 2} = 327 \text{ [лк]}.$$

Для оценки воспользуемся таблицей 5.25 «Требования к освещению рабочих мест в помещениях общественных зданий, а также сопутствующих им производственных помещениях» из СанПиН 1.2.3685-21:

Таблица 9. Выдержка из таблицы 5.25 СанПиН 1.2.3685-21

Помещение	Рабочая поверхность и плоскость нормирования КЕО и освещённости	Искусственное освещение		
		Освещённость, лк		
		при комбинированном освещении		при общем освещении
		всего	от общего	
Залы персональных компьютеров, машинописное бюро	Горизонтальная – 0.8	500	300	400

Для определения условий труда по классу (подклассу) условий труда при воздействии световой среды воспользуемся таблицей из Приложения №16 приказа Минтруда России от 24.01.2014 №33:

Таблица 10. Отнесение условий труда по классу (подклассу) условий труда при воздействии световой среды

Наименование показателя	Класс (подкласс) условий труда	
	допустимый	вредный

	2	3.1	3.2
Искусственное освещение			
Освещённость рабочей поверхности E, лк	$\geq E_n$	$\geq 0.5 E_n$	$< E_n$

Полученное фактическое значение общей освещённости 327 лк при комбинированном освещении рабочего помещения, в котором проводилась разработка программного модуля голосового управления, согласно таблице 10, соответствует допустимому классу условий труда.

Анализ физиологичности данного вида освещения показал, что система является физиологичной, поскольку фактическое значение освещённости попадает в диапазон величин от -10% до +20% (от 270 до 360 лк) от E_n .

15.4.3. Электробезопасность

В рабочем помещении используется трёхфазная сеть переменного тока частотой 50 Гц, напряжением 220 В с заземлённой нейтралью.

Для проведения оценки помещения в отношении опасности поражения людей электрическим током воспользуемся пунктами 1.1.6–1.1.13 правил устройства электроустановок ПУЭ-7.

Характеристики микроклимата в рабочем помещении, в котором проводилась разработка программного модуля по распознаванию голоса, представлены в таблице.

Таблица 11. Характеристики микроклимата в рабочем помещении

Характеристика	Значение
Относительная влажность воздуха	50 %
Температура воздуха	23 °С

Основываясь на данных в приведённой таблице, можно определить, что помещение не является влажным, сырым и особо влажным, так как относительная влажность воздуха меньше 60 %, а также не является жарким, так как температура воздуха меньше 35 °С.

В рабочем помещении не производится механическое измельчение твёрдых тел, транспортировка и перегрузка пылящих материалов, обработка поверхностей материалов, следовательно, в помещении отсутствуют

источники выделения технологической пыли, поэтому помещение не является пыльным.

В помещении не содержатся агрессивные пары, газы, жидкости, не образуются отложения или плесень, разрушающие изоляцию и токоведущие части электрооборудования.

Все комплектующие части электрооборудований и заземлённая часть закрыты от прямого прикосновения к ним человека.

Руководствуясь п. 1.1.13 ПУЭ-7 в отношении опасности поражения людей электрическим током рассматриваемое помещение является помещением без повышенной опасности.

15.4.4. Пожарная опасность

Пожар представляет угрозу здоровью сотрудников и может привести к разрушению оборудования и здания. В помещении с большим количеством ПЭВМ существует риск возгорания из-за дефектов оборудования, короткого замыкания или неправильной эксплуатации.

Согласно своду правил СП 12.13130.2009 «Определение категорий помещений, зданий и наружных установок по взрывопожарной и пожарной опасности» категории помещений по взрывопожарной и пожарной опасности принимаются в соответствии с таблицей:

Таблица 12. Категории помещений по взрывопожарной и пожарной опасности

Категория помещения	Характеристика веществ и материалов, находящихся (обращающихся) в помещении
А повышенная взрывопожаро- опасность	Горючие газы, легковоспламеняющиеся жидкости с температурой вспышки не более 28 °С в таком количестве, что могут образовывать взрывоопасные парогазовоздушные смеси, при воспламенении которых развивается расчетное избыточное давление взрыва в помещении, превышающее 5 кПа, и (или) вещества и материалы, способные взрываться и гореть при взаимодействии с водой, кислородом воздуха или друг с другом, в таком количестве, что расчетное избыточное давление взрыва в помещении превышает 5 кПа
Б взрывопожаро- опасность	Горючие пыли или волокна, легковоспламеняющиеся жидкости с температурой вспышки более 28 °С, горючие жидкости в таком количестве, что могут образовывать взрывоопасные пылевоздушные или паровоздушные смеси, при воспламенении которых развивается расчетное избыточное давление взрыва в помещении, превышающее 5 кПа
В1-В4 пожаро- опасность	Горючие и трудногорючие жидкости, твердые горючие и трудногорючие вещества и материалы (в том числе пыли и волокна), вещества и материалы, способные при взаимодействии с водой, кислородом воздуха или друг с другом только гореть, при условии, что помещения, в которых они находятся (обращаются), не относятся к категории А или Б
Г умеренная пожароопасность	Негорючие вещества и материалы в горячем, раскаленном или расплавленном состоянии, процесс обработки которых сопровождается выделением лучистого тепла, искр и пламени, и (или) горючие газы, жидкости и твердые вещества, которые сжигаются или утилизируются в качестве топлива
Д пониженная пожароопасность	Негорючие вещества и материалы в холодном состоянии

В рабочем помещении не имеются материалы, описанные в пунктах А-Б таблицы, но имеются твёрдые горючие вещества, такие как столы и ковровлин. В соответствии с таблицей помещение относится к категории В.

Для определения точной подкатегории воспользуемся Приложением Б СП 12.13130.2009, в котором указывается определение категорий помещений В1-В4 путём сравнения максимального значения удельной временной пожарной нагрузки. Соотношение категории помещения и удельной пожарной нагрузки приведено в таблице.

Таблица 13. Удельная пожарная нагрузка

Категория	Удельная пожарная нагрузка g на участке, МДж/м ²
B1	Более 2200
B2	1401-2200
B3	181-1400
B4	1-180

Для определения категории помещения, в котором размещены твёрдые горючие материалы, необходимо воспользоваться формулой расчёта пожарной нагрузки Q :

$$Q = \sum_{i=1}^n G_i \times Q_{hi},$$

где:

G_i – количество i -го материала пожарной нагрузки, кг;

Q_{hi} – низшая теплота сгорания i -го материала пожарной нагрузки, МДж/кг.

Для расчёта удельной пожарной нагрузки g , МДж/м², воспользуемся следующей формулой:

$$g = \frac{Q}{S},$$

где S – площадь размещения пожарной нагрузки, м².

Рассчитаем пожарную нагрузку пожароопасного участка:

Для столов из древесины, $G_i = 320$ кг, $Q_{hi} = 16.5$ МДж/кг:

$$Q = 320 \times 16.5 = 5\,280 \text{ МДж.}$$

Для линолеума, $G_i = 650$ кг, $Q_{hi} = 27$ МДж/кг:

$$Q = 650 \times 27 = 17\,550 \text{ МДж.}$$

Таким образом, общая пожарная нагрузка Q составляет 22 830 МДж.

Рассчитаем удельную пожарную нагрузку по формуле:

$$g = \frac{22\,830}{330} = 69.2 \frac{\text{МДж}}{\text{м}^2}$$

Согласно таблице 14 полученное значение удельной пожарной нагрузки соответствует помещению категории В4.

15.5. Итоговая оценка условий труда

В данном разделе рассмотрена тема «Обеспечение безопасных условий труда при разработке системы эргономической оценки кабины самолета на основе теста психомоторной бдительности, методики PVT и NASA-TLX».

Подробно описаны условия труда на рабочем месте и проведен анализ их соответствия нормативным требованиям. Установлено, что все показатели соответствуют необходимым стандартам, обеспечивая комфортные и безопасные условия для специалиста при выполнении исследовательской работы.

Уровни воздействия потенциально вредных факторов находятся в пределах допустимых норм и не представляют угрозы для здоровья программиста-разработчика. Дополнительных улучшений условий труда не требуется.

Заключение

В процессе выполнения дипломной работы были:

1. Сформулированы требования к системе эргономической оценки кабины самолета;
2. Проведен анализ существующих решений и обоснован выбор стека технологий для реализации тестов PVT и NASA-TLX;
3. Разработаны ключевые модули системы, включая проведение тестов психомоторной бдительности (PVT) и субъективной оценки рабочей нагрузки (NASA-TLX);
4. Создан графический пользовательский интерфейс (GUI), обеспечивающий интуитивно понятное взаимодействие с системой и визуализацию результатов;
5. Проведено тестирование разработанной системы, подтвердившее корректность ее работы и соответствие поставленным требованиям.

Реализация данной системы в будущем позволит эффективно анализировать эргономические характеристики кабины самолета, снизить число ошибок, связанных с человеческим фактором, и повысить безопасность полетов, обеспечив оптимальные условия работы для пилотов в условиях высокой нагрузки.

Список использованных источников и литературы

1. Basner, M., Dinges, D. F. Maximizing Sensitivity of the Psychomotor Vigilance Test (PVT) to Sleep Loss // Sleep. – 2011.
2. Loh, S., Lamond, N., Dorrian, J., Roach, G., Dawson, D. The Psychomotor Vigilance Test: A Comparison of Different Testing Protocols // Behavior Research Methods. – 2004.
3. Hart, S. G., Staveland, L. E. Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research // Advances in Psychology. – 1988.
4. Nygren, T. E. Psychometric Properties of Subjective Workload Measurement Techniques: Implications for Their Use in the Assessment of Perceived Mental Workload // Human Factors. – 1991.
5. Sanders, M. S., McCormick, E. J. Human Factors in Engineering and Design. – McGraw-Hill, 1993.
6. Wickens, C. D., Hollands, J. G., Banbury, S., Parasuraman, R. Engineering Psychology and Human Performance. – Routledge, 2015.
7. Reason, J. Human Error. – Cambridge University Press, 1990.
8. Salas, E., Maurino, D. Human Factors in Aviation. – Academic Press, 2010.
9. Norman, D. A. The Design of Everyday Things. – Basic Books, 2013.
10. ICAO. Human Factors Training Manual. – International Civil Aviation Organization, 2019.
11. Eurocontrol. Human Factors in Aviation Safety. – Eurocontrol Report, 2015.
12. Грешников, И. И., Златомрежев, В. И. Использование передовых технологий для оптимизации информационно-управляющего поля кабины перспективного самолета // Нейрокомпьютеры и их применение. – XVIII Всероссийская научная конференция. – Москва, 2020. – С. 66-69.
13. Грешников, И. И., Златомрежев, В. И. Внедрение функции управления взглядом в информационно-управляющее поле кабины пилотов //

- Информационные технологии. – 2021. – Т. 27. – № 8. – С. 445-448. DOI: 10.17587/it.27.445-448.
14. Себряков, Г. Г., Корсун, О. Н., Лавров, А. О. Современные аудиотехнологии в информационно-управляющем поле кабины пилота. – Москва: ИД Академии Жуковского, 2021. – 360 с.
 15. Kirwan, B. A., Ainsworth, L. K. Guide to Task Analysis. – London: Taylor & Francis, 1992.
 16. Promplace. Минпромторг ищет разработчиков бортового оборудования нового поколения для самолётов и вертолётов [Электронный ресурс]. – Режим доступа: https://promplace.ru/news/minpromtorg-ishjet-razrabotchikov-bortovogo-oborudovaniya-novogo-pokoleniya-dlya-samoletov-i-vertoletov_501254.html.
 17. Honeywell. Hey Siri, take off! Get ready for more-advanced planes [Электронный ресурс]. – URL: <https://www.cnet.com/news/honeywell-tests-gear-for-even-more-high-tech-planes>.
 18. Garmin. Talking to Your Airplane With Telligence Voice Control [Электронный ресурс]. – URL: <https://www.garmin.com/en-US/blog/aviation/talking-to-your-airplane/>.
 19. Trzos, M., Dostl, M., Machkov, P., Eitlerov, J. Voice Control in a Real Flight Deck Environment // Lecture Notes in Computer Science. – Vol. 11107. – Springer, Cham, 2018.
 20. Dekker, S. The Field Guide to Understanding Human Error. – CRC Press, 2017.
 21. Helmreich, R. L., Merritt, A. C., Wilhelm, J. A. The Evolution of Crew Resource Management Training in Commercial Aviation // The International Journal of Aviation Psychology. – 1999.
 22. Шрейдер, А. В., Шилов, А. М. Оценка эффективности когнитивной нагрузки пилотов в сложных условиях // Авиация и космонавтика. – 2020. – № 5. – С. 45-49.

23. Миронов, А. С., Ильина, Е. В. Применение методов анализа данных для оптимизации когнитивных процессов пилотов // Вестник авиационной техники. – 2021. – № 3. – С. 56-62.
24. Davis, K. H., Biddulph, R., Balashek, S. Automatic Recognition of Spoken Digits // The Journal of the Acoustical Society of America. – 1952. – Vol. 24, No. 6. – P. 637–642.
25. Sakoe, H., Chiba, S. Dynamic Programming Algorithm Optimization for Spoken Word Recognition // Acoustics, Speech and Signal Processing, IEEE Transactions on. – 1978. – Vol. 26, No. 1. – P. 43–49.
26. Young, M. S., Brookhuis, K. A., Wickens, C. D., Hancock, P. A. State of Science: Mental Workload in Ergonomics // Ergonomics. – 2015.
27. Sweller, J. Cognitive Load During Problem Solving: Effects on Learning // Cognitive Science. – 1988.
28. Paas, F., van Merriënboer, J. J. Instructional Control of Cognitive Load in the Training of Complex Cognitive Tasks // Educational Psychology. – 1994.
29. Casner, S. M., Schooler, J. W. Vigilance Impossible: Diligence, Distraction, and Daydreaming All Lead to Failures in Aircraft Monitoring // Human Factors. – 2015.
30. Parasuraman, R., Riley, V. Humans and Automation: Use, Misuse, Disuse, Abuse // Human Factors. – 1997.
31. Грешников, И. И. Перспективное информационно-управляющее поле кабины, реализующее новые способы информационного обеспечения экипажа // Материалы докладов 5-й Международной конференции. – Москва, 2019. – С. 77-87.
32. Патент на промышленный образец № 92303 Российской Федерации. Комплекс оборудования (стенд) прототипирования интерфейса кабины самолета MC-21. – 2015.
33. Руководство ICAO по эргономике авиационных систем. – ICAO, 2019.
34. Boeing. Flight Deck Design Philosophy. – Boeing Technical Report, 2017.

35. Airbus. Human-Machine Interaction in the Cockpit. – Airbus White Paper, 2016.
36. Roach, G., Petrilli, R., Lamond, N., Dawson, D. Managing Fatigue in Flight Operations: An Evidence-Based Approach // Aviation, Space, and Environmental Medicine. – 2006.
37. Rubio, S., Díaz, E., Martín, J., Puente, J. M. Evaluation of Subjective Mental Workload: A Comparison of SWAT, NASA-TLX, and Workload Profile Methods // Applied Psychology. – 2004.
38. Mitchell, L., Alderman, P. The Role of Simulation in Training Pilots for Cognitive Load Management // Journal of Aviation Research. – 2017.
39. Себряков, Г. Г., Корсун, О. Н., Лавров, А. О. Современные аудиотехнологии в кабине пилота. – Москва: ИД Академии Жуковского, 2021.
40. Anokhin, A. N., Nazarenko, N. A. Designing Interfaces for Human-Machine Systems // Biotechnosphere. – 2010.