|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** |

Институт Информационных технологий

Кафедра Математического обеспечения и стандартизации информационных технологий

**Отчет по практической работе**

по дисциплине «Тестирование и верификация ПО»

|  |  |
| --- | --- |
| **Выполнили:**  Студенты группы ИКБО-50-23 | Гнесь Я.Э.  Суханов Д.В.  Победимова А.П. |
| **Проверил:**  Преподаватель | Ильичев Г.П. |

2025 г.

1. **Введение**

Mente Nova — это настольное приложение, разработанное для помощи студентам в эффективной организации учебных материалов. Оно предоставляет удобный интерфейс для структурирования предметов и загрузки различных типов файлов, включая лекции, практические задания, лабораторные работы, отчеты, фотографии и многое другое. Приложение позволяет организовывать данные по учебным семестрам и предлагает функции управления файлами, такие как переименование, удаление и открытие, с встроенной поддержкой чтения PDF-файлов.

Цели: структуризация материалов в одном приложении с возможностью чтения файлов без сторонних приложений.

Сфера: основной сферой использования является учеба.

1. **Основания для разработки**

* Проблема: разрозненное хранение материалов (мессенджеры, облака, локальные папки), отсутствие единого интерфейса и структуры.
* Необходимость: единое приложение с иерархией семестр → предмет → тип материала, с быстрым доступом и базовым просмотром.
* Технологические предпосылки: кроссплатформенность Java/JavaFX, объектное хранилище MinIO, библиотеки для чтения документов.

1. **Назначение разработки**

* Пользователи: студенты, преподаватели, учебные группы.
* Сценарии: импорт/загрузка материалов; организация по семестрам и предметам; быстрый поиск; базовый просмотр PDF; управление файлами (переименование/удаление/перемещение).
* Ожидаемый эффект: повышение организованности, снижение времени на поиск материалов, единое место хранения.

1. **Требования к программе**

Функциональные:

* Структура данных: семестры → предметы → категории материалов → файлы.
* Операции с хранилищем: загрузка/скачивание/перемещение/переименование/удаление файлов и папок; создание/удаление разделов.
* Просмотр: встроенное чтение PDF; отображение метаданных (размер, дата изменения).
* Поиск и фильтрация: по названию, типу, предмету, семестру.
* Импорт/экспорт: пакетная загрузка папок; скачивание папок.

**Нефункциональные**:

* Кроссплатформенность: Windows/macOS/Linux (JDK 23).
* Производительность: потоковые операции без “заморозки” UI для файлов ≥100 МБ.
* Надёжность: устойчивость к сетевым сбоям MinIO; возобновляемые операции (по возможности).
* Безопасность: хранение конфигурации доступа вне кода; маскирование секретов.
* Логирование: операции и ошибки через TinyLog.

**Технологии:**

* Java 23, Spring Boot, JavaFX (OpenJFX), MinIO (io.minio), Apache PDFBox, Apache POI/ODF Toolkit (по необходимости), Maven.

1. **Требования к интерфейсу**

* Главный экран: дерево/список семестров и предметов, панель файлов, строка поиска, панель действий (загрузить/скачать/переименовать/удалить/открыть).
* Темы: светлая и тёмная.
* Юзабилити: крупные интерактивные элементы, горячие клавиши для частых действий, подтверждения опасных операций.
* Адаптивность: корректная компоновка при изменении размеров окна; сохранение пользовательских предпочтений (последний семестр/предмет).
* Просмотрщик: встроенное окно просмотра PDF с базовой навигацией (страницы, масштаб).

1. **Критерии приемки**

* Функциональность: все заявленные операции с файлами и папками выполняются корректно; структура семестр → предмет работает; PDF открывается во встроенном просмотрщике.
* Стабильность: приложение отрабатывает типовые сценарии без критических ошибок; корректная обработка недоступности MinIO (понятные сообщения).
* Производительность: UI не блокируется при загрузке/скачивании больших файлов; операции завершаются в разумные сроки.
* Удобство: понятная навигация, консистентные статусы и подтверждения; тёмная/светлая тема переключаются без артефактов.
* Сборка и поставка: исполняемый JAR собирается; приложение запускается на поддерживаемых ОС.

1. **Требования к документации**

* Пользовательская: краткое руководство (установка, подключение к MinIO, базовые операции, горячие клавиши).
* Техническая: описание архитектуры, модулей, зависимостей, профилей конфигурации; инструкции сборки и запуска.
* Администрирование: параметры application.properties/yml, переменные окружения, примеры конфигурации MinIO.

1. **Порядок контроля и приемки**

* Тестирование: функциональные тест-кейсы по ключевым сценариям (CRUD файлов/папок, просмотр PDF, поиск); проверка отказоустойчивости (падение сети/MinIO).
* Демо-сессия: показ выполнения критериев приемки на эталонном окружении.
* Отчётность: чек-лист прохождения тестов, логи без критических ошибок; фиксация версий артефактов.
* Принятие: подписание протокола приемки при успешном прохождении всех пунктов.

1. **Этапы и сроки разработки**

* Инициация (1 неделя): уточнение требований, проектирование структуры данных и UX-скетчи.
* Архитектура и настройка (1 неделя): каркас Spring Boot/JavaFX, конфигурация Maven, интеграция MinIO.
* Базовый функционал (2 недели): структура семестров/предметов, CRUD файлов/папок, метаданные, логирование.
* Просмотр PDF и поиск (1 неделя): встроенный ридер PDF, поиск/фильтрация.
* UI/UX и темы (1 неделя): адаптивность, светлая/тёмная тема, горячие клавиши.
* Тестирование и стабилизация (1 неделя): исправления, документация, релизный пакет.
* Итог: ориентировочно 6–7 недель, возможна корректировка по результатам анализа рисков и объёма.