**Содержание**

[Содержание 1](#_Toc195571290)

[Введение 3](#_Toc195571291)

[1 Общая часть 5](#_Toc195571292)

[1.1 Описание предметной области 5](#_Toc195571293)

[1.2 Анализ существующей ситуации 6](#_Toc195571294)

[1.3 Постановка задачи 7](#_Toc195571295)

[1.4 Анализ существующих разработок и обоснование необходимости разработки 8](#_Toc195571296)

[2 Специальная часть 10](#_Toc195571297)

[2.1 Выбор технологий и инструментальных средств 10](#_Toc195571298)

[2.1.1 Выбор подхода к разработке 10](#_Toc195571299)

[2.1.2 Выбор среды разработки и языка программирования 11](#_Toc195571300)

[2.2 Разработка спецификаций 13](#_Toc195571301)

[2.2.1 Разработка диаграмм вариантов использования 13](#_Toc195571302)

[2.2.2 Разработка диаграмм последовательностей системы 14](#_Toc195571303)

[2.2.3 Разработка диаграммы пакетов 14](#_Toc195571304)

[2.2.4 Разработка диаграмм классов 15](#_Toc195571305)

[2.2.5 Разработка диаграммы состояний 16](#_Toc195571306)

[2.2.6 Разработка диаграммы деятельности 18](#_Toc195571307)

[2.2.7 Разработка диаграммы размещения 19](#_Toc195571308)

[2.2.8 Разработка инфологической модели базы данных 19](#_Toc195571309)

[2.2.9 Разработка физической модели базы данных 19](#_Toc195571310)

[2.3 Проектирование программного модуля 19](#_Toc195571311)

[2.3.1 Разработка алгоритмов реализации основных функций программного обеспечения 19](#_Toc195571312)

[2.3.2 Проектирование пользовательского интерфейса 20](#_Toc195571313)

[2.4 Реализация программного обеспечения на выбранном языке программирования и в выбранной среде разработки 22](#_Toc195571314)

[2.5 Выбор стратегии тестирования, разработка тестов, тестирование и отладка программного обеспечения 24](#_Toc195571315)

[2.6 Разработка эксплуатационной документации 31](#_Toc195571316)

[2.6.1 Разработка руководства системного программиста 31](#_Toc195571317)

[2.6.2 Разработка руководства пользователя 31](#_Toc195571318)

[3 Безопасность жизнедеятельности 33](#_Toc195571319)

[3.1 Требования к рабочим помещениям и рабочим местам оператора ПК 33](#_Toc195571320)

[3.2 Расчет искусственной освещенности рабочего места оператора ПК 34](#_Toc195571321)

[3.3. Техника безопасности при работе на ПК 36](#_Toc195571322)

[Заключение 39](#_Toc195571323)

[Библиография 41](#_Toc195571324)

# Введение

Современные IT-отделы сталкиваются с необходимостью непрерывного обучения сотрудников в условиях стремительного развития технологий. Для компаний, таких как АО «Апатит», критически важно поддерживать высокий уровень компетенций специалистов, особенно в области актуальных языков программирования. Отсутствие специализированных решений, адаптированных под корпоративные стандарты, приводит к неэффективному использованию ресурсов и замедлению процессов адаптации новых сотрудников.

Разработка интерактивной платформы для изучения языков программирования направлена на решение этих задач. Она позволит не только оптимизировать обучение, но и интегрировать его в рабочие процессы IT-отдела, обеспечивая практическое применение знаний. Актуальность проекта обусловлена растущим спросом на персонализированные образовательные инструменты, которые учитывают специфику бизнес-задач компании.

АО «Апатит», как крупный участник цифровой трансформации, заинтересован во внедрении инновационных решений для повышения квалификации сотрудников. Платформа станет инструментом, который сократит время на освоение новых технологий, таких как Python, Java и SQL, и усилит конкурентные преимущества компании на рынке IT-услуг.

Цель дипломной работы — создание интерактивной платформы, которая объединит теоретические модули, практические задания и аналитику прогресса, адаптированные под потребности IT-отдела АО «Апатит». Для достижения этой цели требуется проанализировать существующие образовательные решения, определить требования к системе, разработать удобный интерфейс с интерактивными элементами, обеспечить интеграцию с корпоративными системами и провести тестирование в реальных условиях.

Платформа будет полезна как начинающим специалистам, так и опытным разработчикам, позволяя им осваивать навыки в гибком формате. Её внедрение также упростит управление обучением для руководителей отделов, предоставляя инструменты для мониторинга результатов и планирования образовательных траекторий.

Для АО «Апатит» реализация этого проекта станет шагом к цифровизации внутренних процессов и укреплению позиций в сфере IT-разработки. Успешное завершение работы позволит компании предложить сотрудникам современный инструмент для профессионального роста, что напрямую отразится на качестве реализации ключевых проектов.

# Общая часть

## Описание предметной области

Предметной областью разрабатываемой платформы является корпоративное обучение, направленное на повышение квалификации сотрудников IT-отдела в области современных языков программирования. В условиях цифровой трансформации компаний, таких как АО «Апатит», критически важно обеспечить непрерывное профессиональное развитие специалистов, чтобы поддерживать конкурентоспособность и эффективность IT-решений.

Платформа ориентирована на создание интерактивной среды, которая объединяет теоретические знания, практические задания и аналитику прогресса. Её ключевая задача — преодолеть разрыв между академическим обучением и реальными рабочими процессами, характерный для многих существующих образовательных ресурсов. Это позволит сотрудникам осваивать навыки программирования (Python, Java, SQL и др.) в контексте задач, актуальных для IT-инфраструктуры компании.

Одной из основных проблем в корпоративном обучении является отсутствие персонализированных траекторий. Универсальные курсы не учитывают специфику проектов АО «Апатит», что снижает мотивацию сотрудников и замедляет их адаптацию. Разрабатываемая платформа решает эту задачу за счет модульной архитектуры, позволяющей гибко настраивать контент под роли сотрудников (разработчики, аналитики, DevOps-инженеры) и текущие бизнес-требования.

Еще одной проблемой является интеграция обучения в рабочие процессы. Существующие системы редко поддерживают синхронизацию с корпоративными инструментами (например, Jira, GitLab), что затрудняет применение знаний на практике. Платформа предусматривает возможность подключения к внутренним системам компании, обеспечивая выполнение заданий в тестовых средах, имитирующих реальные проекты.

Для АО «Апатит» важна также аналитика эффективности обучения. Руководителям IT-отдела необходимы инструменты для мониторинга прогресса сотрудников, оценки ROI образовательных программ и планирования дальнейших шагов. Внедрение платформы с аналитическими модулями позволит автоматизировать сбор данных и формирование отчетов, что упростит принятие управленческих решений.

Таким образом, предметная область проекта охватывает не только образовательные технологии, но и интеграцию обучения в корпоративные процессы. Платформа станет инструментом, который повысит качество IT-разработок компании, ускорит внедрение новых технологий и укрепит позиции АО «Апатит» на рынке цифровых решений.

## Анализ существующей ситуации

Современные IT-отделы, включая команду АО «Апатит», сталкиваются с рядом вызовов в области профессионального развития сотрудников. Несмотря на обилие образовательных ресурсов, сохраняются системные проблемы, которые замедляют адаптацию специалистов и снижают эффективность обучения:

1. Недостаток персонализированных ресурсов. Универсальные курсы редко учитывают специфику проектов компании, что приводит к несоответствию между полученными знаниями и реальными задачами. Это замедляет внедрение новых технологий, увеличивает сроки выполнения проектов и создает нагрузку на сотрудников, вынужденных самостоятельно искать актуальную информацию.

2. Неэффективное использование обучающих материалов. Даже при наличии доступа к платформам вроде Coursera или Udemy, контент часто остается невостребованным из-за отсутствия интеграции с рабочими процессами. Теоретические знания не подкрепляются практикой, что снижает вовлеченность и затрудняет применение навыков в реальных проектах.

3. Разрыв между теорией и практикой. Сотрудники, прошедшие обучение, нередко сталкиваются с трудностями при реализации задач из-за отсутствия среды для отработки навыков. Это приводит к ошибкам в коде, задержкам в разработке и необходимости дополнительного менторства, что увеличивает нагрузку на команду.

4. Высокие затраты на внешние решения. Использование сторонних образовательных платформ часто требует значительных финансовых вложений, при этом их функционал не адаптирован под внутренние стандарты компании. Это ограничивает возможность масштабирования обучения и анализа его эффективности.

Разрабатываемая интерактивная платформа направлена на решение этих проблем. Она объединит гибкие учебные модули, симуляторы реальных задач компании и инструменты аналитики, что позволит:

* Сократить время адаптации новых сотрудников за счет обучения в контексте проектов АО «Апатит»;
* Повысить качество разработки через практику в контролируемых средах, имитирующих рабочие процессы;
* Оптимизировать бюджет на обучение, заменив внешние ресурсы кастомизированными решениями;

## Постановка задачи

Целью дипломной работы является разработка интерактивной платформы для изучения языков программирования, ориентированной на IT-отдел АО «Апатит». Ключевой задачей проекта является создание гибкой образовательной среды, которая объединит теоретические модули, практические задания и инструменты анализа прогресса, адаптированные под внутренние стандарты компании.

Особое внимание уделяется обеспечению безопасности данных и соответствию корпоративным политикам. Платформа должна поддерживать модульную структуру, позволяющую добавлять новые курсы и обновлять контент без нарушения работы системы.

В процессе разработки необходимо применять современные подходы, включая принципы «чистой архитектуры» для разделения слоев приложения (логика, интерфейс, данные) и микросервисную модель для масштабируемости. Это обеспечит не только стабильность работы платформы, но и её адаптацию к будущим изменениям IT-инфраструктуры компании.

Результатом работы станет инструмент, который сократит время адаптации новых сотрудников, повысит качество разработки за счет практико-ориентированного обучения и укрепит позиции АО «Апатит» как технологического лидера в своей отрасли.

1.4 Анализ существующих разработок и обоснование необходимости разработки

На рынке уже представлен ряд образовательных платформ для изучения языков программирования, таких как Codecademy, LeetCode и Coursera. Эти решения предлагают разнообразные форматы обучения: от видеоуроков и текстовых материалов до интерактивных тренажеров с автоматической проверкой кода. Например, Codecademy фокусируется на практических заданиях с мгновенной обратной связью, а LeetCode специализируется на подготовке к техническим собеседованиям через решение алгоритмических задач.

Некоторые платформы, например, Pluralsight, используют персонализацию, адаптируя контент под уровень пользователя. Однако их ключевой недостаток в контексте корпоративного обучения — отсутствие интеграции с внутренними системами компаний. Они не учитывают специфику проектов, внутренние стандарты кодирования или используемые в организации технологии, что снижает применимость знаний на практике.

Существуют также корпоративные LMS-системы (Moodle, TalentLMS), которые позволяют создавать курсы, но их функционал зачастую ограничен теоретическим контентом без поддержки интерактивных сред для программирования. Это создает разрыв между обучением и реальными задачами, с которыми сталкиваются сотрудники IT-отделов.

Разрабатываемая платформа для АО «Апатит» призвана объединить сильные стороны существующих решений, добавив критически важные для компании элементы:

* Интеграцию с корпоративными инструментами (Jira, GitLab), что позволит отрабатывать навыки в средах, имитирующих рабочие проекты;
* Персонализацию обучения на основе ролей сотрудников (например, отдельные модули для разработчиков на Python и аналитиков данных);
* Аналитику прогресса, синхронизированную с HR-системами для оценки эффективности обучения;
* Безопасность данных, соответствующую внутренним политикам компании.

# Специальная часть

## Выбор технологий и инструментальных средств

### 2.1.1 Выбор подхода к разработке

Для разработки интерактивной платформы изучения языков программирования предпочтительным является объектно-ориентированный подход (ООП). Это обусловлено необходимостью создания модульной, гибкой системы, способной адаптироваться к меняющимся требованиям IT-отдела АО «Апатит» и интеграции с корпоративными инструментами.

Ключевым требованием к платформе является четкое разделение ответственности между компонентами:

* Пользовательские модули (теория, практические задания, тесты);
* Интерактивный редактор кода с поддержкой Python, Java, SQL;
* Система аналитики для отслеживания прогресса сотрудников;
* Интеграционные слои для подключения к внутренним системам компании (Jira, GitLab, корпоративная LMS).

Объектно-ориентированный подход позволяет абстрагировать логику каждого компонента в виде классов, обеспечивая независимость их разработки и тестирования. Например, класс «Учебный курс» может включать методы для управления контентом, проверки заданий и генерации отчетов, что упрощает добавление новых языков программирования или форматов обучения.

Использование ООП также способствует масштабируемости. При внедрении новых функций, таких как AI-ассистент для генерации персональных рекомендаций, существующая архитектура не потребует значительных изменений. Наследование и полиморфизм позволяют переиспользовать код, сокращая время разработки. Например, модуль проверки заданий для Python может быть расширен для поддержки Java без дублирования базовой логики.

Альтернативные подходы, такие как процедурное программирование, менее эффективны для данной задачи. Они привели бы к созданию монолитной системы с жестко связанными компонентами, что затруднило бы адаптацию платформы под специфические требования АО «Апатит» (например, интеграцию с внутренними API).

Преимущества ООП в контексте проекта:

* Гибкость: Возможность быстро добавлять новые форматы обучения (вебинары, симуляторы) за счет инкапсуляции.
* Безопасность: Разделение доступа к данным через классы-обертки, что критично для соблюдения корпоративных политик безопасности.
* Тестируемость: Изолированные модули упрощают автоматизацию тестирования, включая проверку корректности выполнения заданий.

Таким образом, объектно-ориентированный подход обеспечивает создание структурированной, легко поддерживаемой платформы, которая соответствует целям АО «Апатит» — повышение качества обучения, снижение затрат на адаптацию сотрудников и укрепление позиций компании в сфере IT-разработки.

### 2.1.2 Выбор среды разработки и языка программирования

Для разработки интерактивной платформы изучения языков программирования выбран стек на основе HTML, CSS и JavaScript в связке с Open Server Panel для локального развертывания и тестирования. Интегрированная среда разработки Visual Studio Code (VS Code) используется как основной инструмент для написания кода. Такой выбор обусловлен следующими факторами:

* Простота и доступность. Базовые веб-технологии (HTML, CSS, JS) позволяют быстро создавать интерактивные интерфейсы без необходимости изучения сложных фреймворков. Это особенно важно для IT-отдела АО «Апатит», где сотрудники уже обладают навыками работы с этими языками.
* Гибкость. JavaScript обеспечивает реализацию динамических элементов: редактора кода с подсветкой синтаксиса, системы проверки заданий и интерактивных тестов. Библиотеки вроде CodeMirror или Ace Editor дополнят функционал, имитируя профессиональные IDE.
* Локальное тестирование. Open Server Panel предоставляет готовую среду для запуска веб-приложений, включая поддержку PHP, MySQL и других технологий, что упрощает отладку и интеграцию с будущими серверными модулями.

Преимущества выбранного стека:

* Минимальные затраты на ввод в эксплуатацию. Отсутствие необходимости в сложной настройке инфраструктуры ускоряет разработку.
* Кроссплатформенность. Платформа работает в любом современном браузере, что обеспечивает доступ для сотрудников с разных устройств и ОС.
* Интеграция с корпоративными системами. Через REST API или веб-сокеты можно подключить платформу к внутренним инструментам компании (например, корпоративной LMS или Jira).
* Расширяемость. При необходимости, проект можно дополнять серверными технологиями (Node.js, PHP) или подключать базы данных (MySQL), используя Open Server Panel как основу для масштабирования.

Роль VS Code:

* Поддержка JavaScript через расширения (ESLint, Prettier) для соблюдения стандартов кодирования.
* Встроенный терминал для работы с Git и управления зависимостями.
* Интеграция с инструментами тестирования и отладки (например, Chrome DevTools).

Для АО «Апатит» такой подход обеспечивает быстрое прототипирование, снижение затрат на обучение команды и возможность поэтапного внедрения функционала. Платформа будет соответствовать требованиям безопасности за счет реализации ролевой модели доступа и шифрования данных на уровне браузера (Web Crypto API). В перспективе, решение может быть развернуто на корпоративном сервере с использованием того же стека, что упростит поддержку и обновления.

## Разработка спецификаций

### 2.2.1 Разработка диаграмм вариантов использования

Диаграмма вариантов использования (рис. 1) отражает ключевые сценарии взаимодействия пользователей с платформой.



Рисунок 1 – Диаграмма вариантов использования

Выбор языка программирования — определение целевого технологического стека (Python, JavaScript, SQL) для индивидуального обучения.

Прохождение модуля — последовательное изучение материалов с адаптивными рекомендациями на основе уровня подготовки.

Решение интерактивных задач — выполнение практических кейсов с мгновенной проверкой кода в изолированной среде.

Анализ прогресса — визуализация результатов через дашборд с метриками завершенных модулей и типичных ошибок.

Синхронизация с Jira — автоматическая передача достижений в систему управления задачами компании для формирования индивидуального плана развития.

### 2.2.2 Разработка диаграмм последовательностей системы

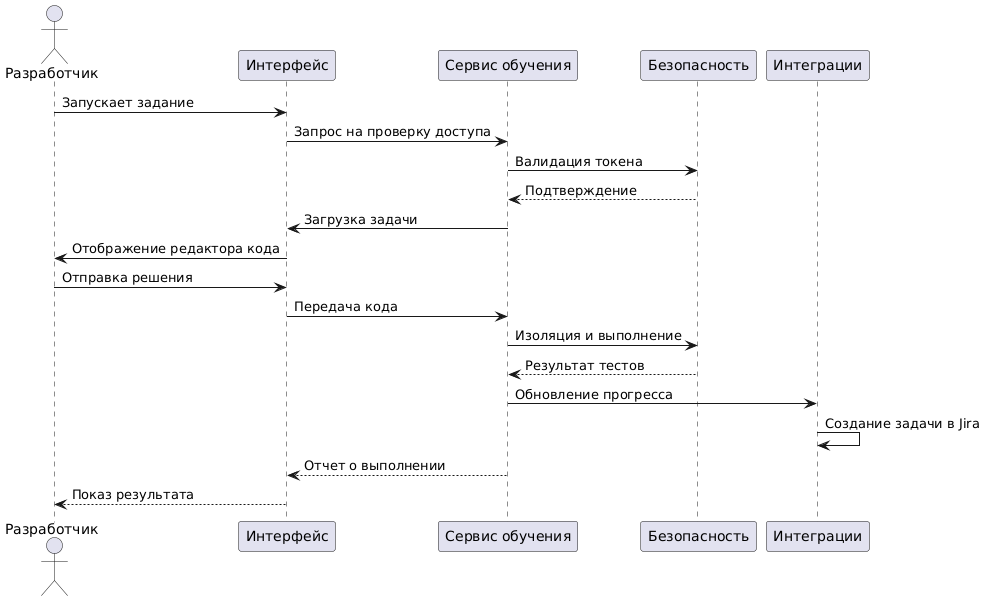


Рисунок 2 – Диаграмма последовательностей системы

### 2.2.3 Разработка диаграммы пакетов

Для данной разработки построена диаграмма пакетов, которая показана ниже на рисунке 3. Она показывает зависимости между частями.

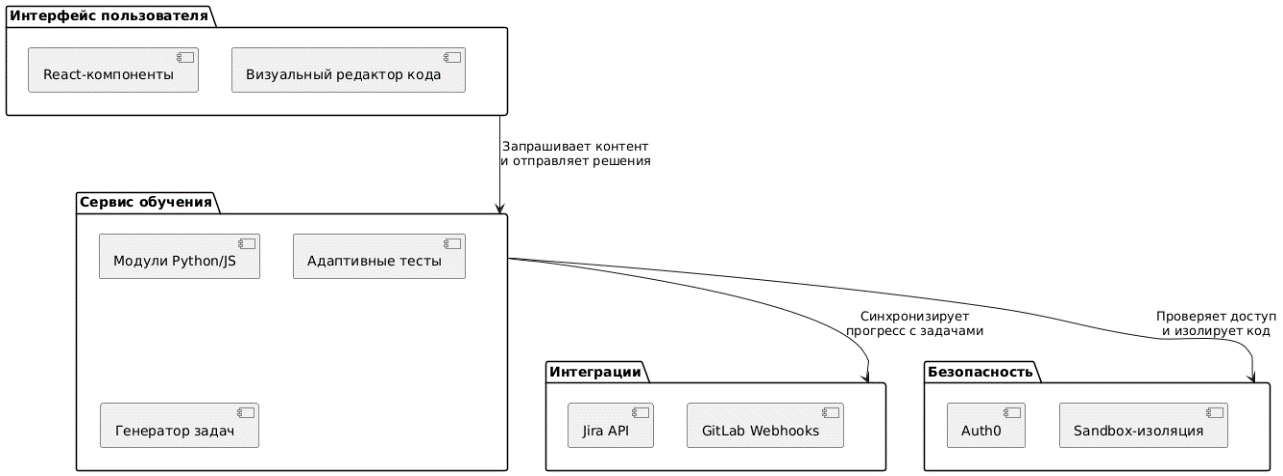


Рисунок 3 – Диаграмма пакетов

### 2.2.4 Разработка диаграмм классов

Для разработки пакета для взаимодействия создана диаграмма классов, показанная ниже на рисунке 4.

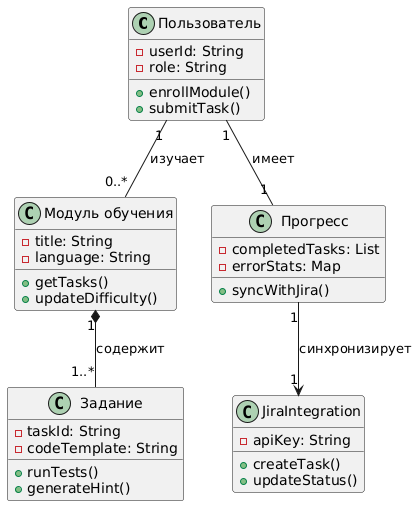


Рисунок 4 – Диаграмма классов

Основные сущности:

* Пользователь — сотрудник IT-отдела, имеющий доступ к курсам. Хранит данные о роли (начинающий/продвинутый) для персонализации обучения.
* Модуль обучения — логическая единица курса (например, «Основы Python»), динамически корректирующая сложность заданий на основе статистики ошибок.
* Задание — интерактивный кейс с шаблоном кода, автоматически проверяющий решение через встроенный тестовый фреймворк.
* Прогресс — аналитический компонент, формирующий рекомендации для следующих модулей и экспортирующий метрики в Jira.

Ключевые связи:

Адаптивная логика платформы реализована через двустороннее взаимодействие модулей и заданий — при частых ошибках система упрощает следующие задачи, а при успешном выполнении — усложняет. Интеграция с Jira обеспечивает прозрачность обучения для руководителей: завершение модуля автоматически создает задачу на код-ревью или практическое применение навыка в рабочих проектах.

Архитектура поддерживает расширение — добавление новых языков программирования или систем аналитики не требует изменения ядра.

### 2.2.5 Разработка диаграммы состояний



Рисунок 5 – Диаграмма состояний

Ключевые состояния:

* Ожидание выбора задания — начальное состояние, при котором пользователь выбирает учебный модуль или конкретную задачу из списка.
* Проверка решения — выполнение кода в изолированной среде, автоматическое тестирование на соответствие критериям, анализ типовых ошибок (синтаксис, логика, оптимизация).
* Анализ прогресса — формирование персонализированных рекомендаций:
* Усложнение заданий при успешном выполнении;
* Повторение темы с адаптированными примерами при частых ошибках;
* Автоматическая синхронизация результатов с Jira для планирования код-ревью.

Логика переходов:

* Выбор задания пользователем активирует проверку кода, включая интеграцию с GitLab для анализа стиля программирования.
* Система возвращается в исходное состояние после формирования отчета, что обеспечивает непрерывный цикл обучения.

Диаграмма отражает адаптивность платформы: переход между состояниями зависит не только от действий пользователя, но и от данных аналитики (например, частоты ошибок в определенных темах). Это позволяет IT-отделу АО «Апатит» масштабировать обучение сотрудников, фокусируясь на слабых местах команды.

### 2.2.6 Разработка диаграммы деятельности

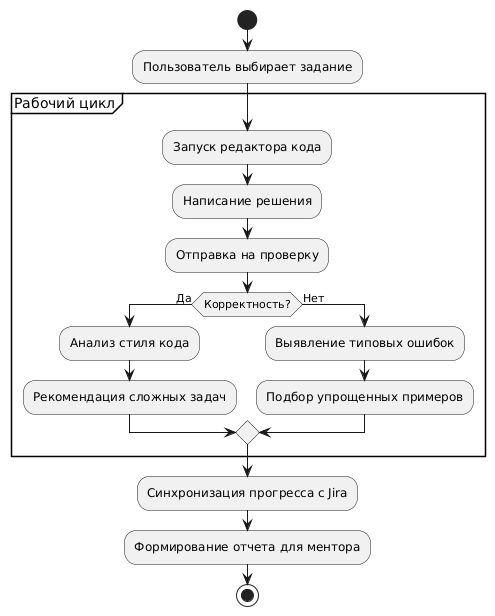


Рисунок 6 – Диаграмма деятельности.

Данная диаграмма деятельности описывает последовательность действий, необходимых для создания системы.

### 2.2.7 Разработка диаграммы размещения

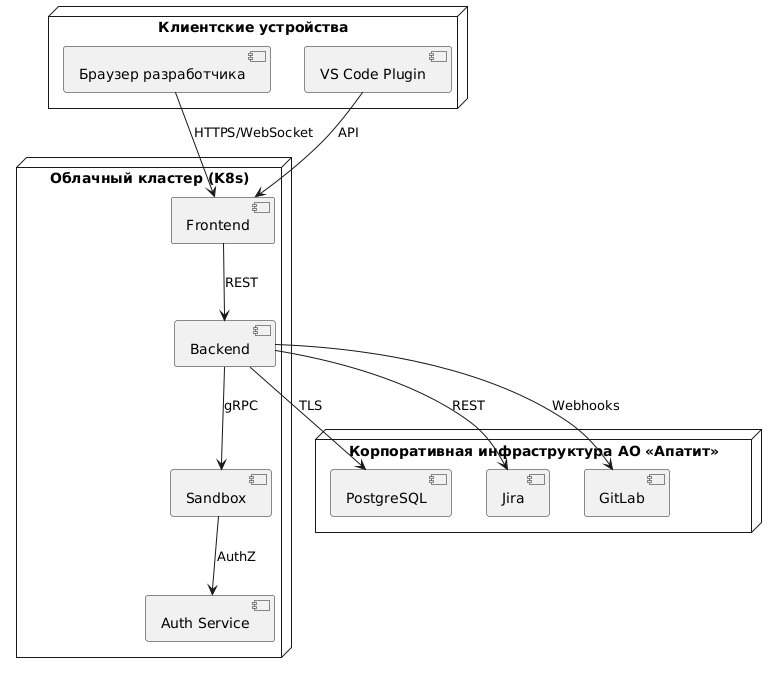


Рисунок 7 – Диаграмма размещения.

### 2.2.8 Разработка инфологической модели базы данных

Для данного все результаты сохраняются в следующем формате   
{"total\_places":50,"occupied\_places":25} (Json)

### 2.2.9 Разработка физической модели базы данных

Пункт 2.2.8

## Проектирование программного модуля

### 2.3.1 Разработка алгоритмов реализации основных функций программного обеспечения

Алгоритм начинается с начинается с запуска учебного модуля.

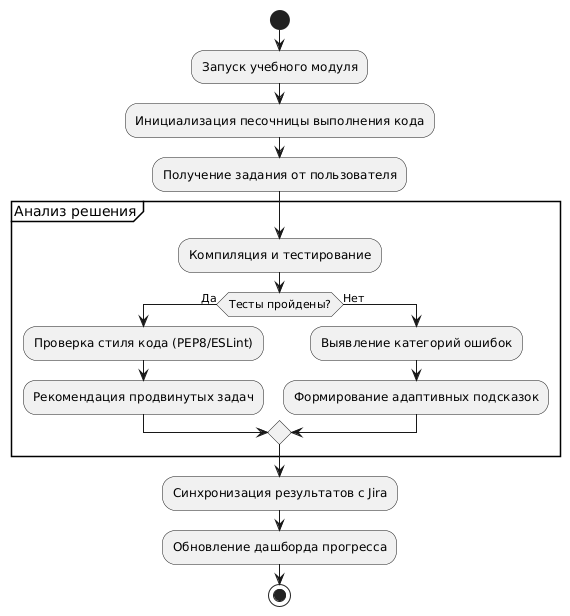


Рисунок 8 – Алгоритм получения результатов

### 2.3.2 Проектирование пользовательского интерфейса

При разработке интерактивной платформы для изучения языков программирования ключевым аспектом является выбор модели взаимодействия, которая обеспечит сотрудникам IT-отдела АО "Апатит" интуитивное освоение материалов и эффективное применение навыков. В рамках проектирования рассматривались два подхода:

* Процедурно-ориентированный интерфейс, фокусирующийся на последовательном выполнении задач (например, прохождение уроков, отправка решений).
* Объектно-ориентированный интерфейс, ориентированный на взаимодействие с элементами системы как с самостоятельными объектами (редактор кода, интерактивные задания, визуализация результатов).

Для платформы обучения критически важно сочетать структурированность контента и гибкость взаимодействия. Процедурный подход позволяет организовать учебные модули в логической последовательности, что упрощает навигацию для новичков. Объектно-ориентированная модель, в свою очередь, дает возможность напрямую работать с элементами платформы: например, перетаскивать блоки кода, настраивать параметры заданий или отслеживать прогресс в реальном времени.

Учитывая специфику аудитории — IT-специалистов, знакомых с основами программирования, — был выбран гибридный подход. Он объединяет:

* Иерархическое меню для структурированного доступа к курсам, темам и заданиям, что соответствует процедурной модели.
* Элементы прямого манипулирования (интерактивный редактор кода, визуализация выполнения программ), реализующие объектно-ориентированный подход.

Такой дизайн позволяет сотрудникам быстро ориентироваться в учебных материалах, сохраняя возможность экспериментировать с кодом и мгновенно получать обратную связь. Например, выбор задания из меню запускает среду разработки, где пользователь взаимодействует с объектами (файлами, переменными, графиками), что соответствует прикладным задачам, решаемым в АО "Апатит" (оптимизация скриптов, анализ данных).

Отказ от избыточной сложности в пользу минималистичного интерфейса снижает когнитивную нагрузку, а интеграция с корпоративными системами авторизации обеспечивает безопасность. Это делает платформу не только инструментом обучения, но и частью рабочей инфраструктуры предприятия.

## 2.4 Реализация программного обеспечения на выбранном языке программирования и в выбранной среде разработки

Разработка интерактивной платформы для изучения языков программирования выполнена с использованием веб-технологий, что обеспечивает кроссплатформенность и простоту интеграции в корпоративную инфраструктуру АО "Апатит". В качестве основного стека выбраны:

* Frontend: HTML, CSS, JavaScript — для создания интуитивного интерфейса с поддержкой интерактивных элементов (редактор кода, навигационное меню).
* Backend: PHP — для обработки авторизации, управления курсами и взаимодействия с базой данных.
* СУБД: MySQL — для хранения данных пользователей, прогресса обучения и материалов курсов.

Этапы реализации:

* Проектирование архитектуры. Система разделена на модули: аутентификация, управление курсами, интерактивный редактор кода и аналитика прогресса.
* Интеграция авторизации. Реализована проверка учетных данных сотрудников через корпоративную систему.

Пример кода формы входа:

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Платформа для изучения языков программирования</title>

<link rel="stylesheet" href="../css/StyleSheet.css">

</head>

<body>

<div class="container">

<h1>Платформа для изучения языков программирования</h1>

<?php

if($\_COOKIE['login'] == ''):

?>

<form action="../php/log-in.php" method="post">

<h1>Вход</h1>

<p>Введите логин:</p>

<input type="text" class="form-control" name="login" id="login" placeholder="login" required>

<p>Введите пароль:</p>

<input type="text" class="form-control" name="password" id="password" placeholder="password" required><br>

<button class="btn-login" type="submit">Войти</button>

</form>

<div class="back-link">

<a href="sign-up.html">У вас нет аккаунта?</a>

</div>

<?php else: ?>

<p>Привет <?=$\_COOKIE['login']?>.</p>

<div class="buttons">

<a href="course-search.php">Список курсов</a>

<a href="favourites-course.php">Избранное</a>

<?php

if($\_COOKIE['admin'] == '1'):

?>

<a href="administration.php">Администрирование</a>

<?php endif ?>

<a href="../php/exit.php">Выход из аккаунта</a>

</div>

<?php endif ?>

</div>

</body>

</html>

Для безопасности используются хеширование паролей и защита от SQL-инъекций.

3. Разработка интерфейса. Создано иерархическое меню для доступа к курсам, избранному и администрированию (для роли администратора).

## 2.5 Выбор стратегии тестирования, разработка тестов, тестирование и отладка программного обеспечения

Для выбора наиболее подходящего метода и инструмента тестирования необходимо учитывать ряд критериев, таких как освоение, эффективность, распространенность и принцип использования.

В табличной форме приведена сравнительная характеристика методов тестирования и инструментов для автоматизации тестирования, соответственно. Так, в таблице 1 представлено сравнение черного, белого и серого ящиков, включая преимущества и недостатки каждого метода. В таблице 2 отражены основные характеристики инструментов Selenium, Selenium IDE и PHPUnit, таких как функциональность, сложность использования и распространенность среди разработчиков.

Однако стоит учитывать, что выбор метода и инструмента тестирования зависит от конкретного проекта и его специфики. Поэтому рекомендуется проводить тщательный анализ требований проекта и оценивать эффективность тестирования на основе выбранного метода и инструмента.

| **Характеристика** | **Черный ящик** | **Белый ящик** | **Серый ящик** |
| --- | --- | --- | --- |
| Описание | В этом методе тестирующий работает со стороны пользователя, не зная деталей внутренней работы программы. | В этом методе тестирующий имеет доступ к исходному коду программы и может проводить тестирование на уровне алгоритмов и структур данных. | Это смешанный метод, который сочетает некоторые элементы "черного ящика" и "белого ящика". Тестирующий имеет ограниченный доступ к исходному коду программы, что позволяет ему проводить более детальное тестирование, однако он все еще не имеет полного доступа ко всем деталям реализации. |
| Объект тестирования | Объектом тестирования является готовое приложение или система в целом. | Объектом тестирования является исходный код и отдельные компоненты программы. | Объектом тестирования является как готовое приложение, так и частично доступный для анализа исходный код. |
| Сложность тестирования | Метод "черного ящика" обычно проще в использовании, так как не требует знания языка программирования или особой экспертизы в данной области. | Метод "белого ящика" может быть более сложным, так как требуется знание языка программирования и понимание внутренней работы программы. | Метод "серого ящика" находится между "черным" и "белым" ящиками по сложности. Тестирующий должен иметь некоторые знания в области программирования, чтобы анализировать доступный ему код и проводить дополнительные тесты. |
| Уровень детализации тестирования | Метод "черного ящика" позволяет проводить тестирование на уровне функциональности приложения, однако не обеспечивает глубокого понимания внутренней работы программы. | Метод "белого ящика" позволяет проводить тестирование на уровне отдельных модулей программы и алгоритмов. | Метод "серого ящика" обеспечивает более детальное тестирование, чем "черный ящик", но не такое глубокое и полное, как "белый ящик". |
| Возможность автоматизации тестирования | Метод "черного ящика" легко поддается автоматизации с помощью специализированных инструментов для автоматического тестирования. | Метод "белого ящика" также может быть автоматизирован, однако требует более тщательной настройки и ручной работы. | Метод "серого ящика" может быть автоматизирован в определенной степени. |

Таблица 2 – Инструменты для автоматизации тестирования

| **Инструмент** | **Описание** | **Функциональность** | **Сложность использования** | **Распространенность** |
| --- | --- | --- | --- | --- |
| Selenium | Фреймворк для автоматизированного тестирования веб-приложений. | Позволяет записывать и воспроизводить действия пользователя на веб-страницах, проверять содержимое элементов интерфейса, реализовать тестовые сценарии. | Высокая сложность использования, так как требует знания языка программирования и разработки автоматических скриптов. | Один из самых популярных инструментов для автоматизации тестирования веб-приложений. |
| Appium | Фреймворк для автоматизированного тестирования мобильных приложений. | Позволяет записывать и воспроизводить действия пользователя на мобильных устройствах, проверять содержимое элементов интерфейса, реализовать тестовые сценарии. | Высокая сложность использования, так как требует знания языка программирования и разработки автоматических скриптов. | Широко используется для автоматизации тестирования мобильных приложений на платформах Android и iOS. |
| JMeter | Инструмент для функционального и нагрузочного тестирования веб-приложений. | Позволяет создавать и запускать тестовые сценарии, генерировать нагрузку на серверы, анализировать результаты. | Умеренная сложность использования, так как требует знания основ работы с веб-приложениями и настройки нагрузочных тестов. | Широко используется для функционального и нагрузочного тестирования веб-приложений. |
| TestComplete | Коммерческий инструмент для автоматизированного тестирования различных типов приложений. | Позволяет записывать и воспроизводить действия пользователя на экране, проверять содержимое элементов интерфейса, реализовать тестовые сценарии. | Относительно низкая сложность использования, так как имеет графический интерфейс и не требует знания языка программирования. | Широко используется для автоматизации тестирования различных типов приложений. |
| Postman | Инструмент для тестирования API. | Позволяет отправлять запросы к API и анализировать ответы, создавать тестовые сценарии для проверки работоспособности API. | Относительно низкая сложность использования, так как имеет графический интерфейс и не требует знания языка программирования. | Широко используется для тестирования API различных веб-сервисов. |

Для обеспечения надежности интерактивной платформы изучения языков программирования, разрабатываемой для IT-отдела АО "Апатит", ключевым этапом является внедрение системы автоматизированного тестирования. Учитывая веб-ориентированную архитектуру платформы (фронтенд на HTML/CSS/JS, бэкенд на PHP), выбор инструментов тестирования осуществляется следующим образом:

1. Тестирование фронтенда

Инструмент: Selenium

Обоснование:

* Позволяет автоматизировать проверку пользовательского интерфейса, включая:
* Корректность отображения элементов (меню, редактор кода, кнопки).
* Работоспособность интерактивных функций (например, отправка решений, навигация между курсами).
* Интеграцию с авторизацией через корпоративную систему.

Пример использования:

// Тест на успешный вход в систему (пример на C# для демонстрации, но в реальности используется связка Selenium + Python/JS):

[Test]

public void TestLoginFlow()

{

driver.Navigate().GoToUrl("https://platform/апатит/login");

driver.FindElement(By.Id("login")).SendKeys("employee123");

driver.FindElement(By.Id("password")).SendKeys("securePass");

driver.FindElement(By.ClassName("btn-login")).Click();

Assert.IsTrue(driver.Url.Contains("/dashboard"));

}

2. Тестирование бэкенда

Инструмент: PHPUnit (вместо NUnit, так как бэкенд реализован на PHP)

Обоснование:

* Проверка бизнес-логики: работа с курсами, сохранение прогресса, генерация аналитики.
* Тестирование API для интеграции с корпоративными сервисами.
* Валидация безопасности: защита от SQL-инъекций, корректность хеширования паролей.

Пример теста для проверки создания курса:

php

Copy

class CourseTest extends TestCase

{

public function testCourseCreation()

{

$courseData = [

'title' => 'Оптимизация SQL-запросов',

'description' => 'Курс по работе с базами данных предприятия'

];

$response = $this->post('/api/courses', $courseData);

$this->assertResponseStatus(201);

$this->seeInDatabase('courses', ['title' => $courseData['title']]);

}

}

3. Тестирование интерактивного редактора кода

Инструмент: Jest + Cypress

Обоснование:

* Проверка подсветки синтаксиса, автодополнения и валидации кода в реальном времени.
* Интеграция с API выполнения кода (например, отправка задания на сервер и проверка результата).

## 2.6 Разработка эксплуатационной документации

### 2.6.1 Разработка руководства системного программиста

По ГОСТ 19.503-79 руководство системного программиста должно содержать всю информацию, необходимую для установки программного обеспечения, его настройки и проверки работоспособности. Кроме того, как указывалось выше, в него часто включают и описание необходимого обслуживания, которое раньше приводилось в руководстве оператора (ГОСТ 19.505-79) и/или руководстве по техническому обслуживанию (ГОСТ 19.508-79). В настоящее время данную схему используют для составления руководства системному администратору.

Руководство системного программиста должно содержать следующие разделы:

* 1. общие сведения о программном продукте;
  2. структура;
  3. настройка;
  4. проверка;
  5. дополнительные возможности;
  6. сообщения системному программисту.

Руководство системного программиста представлено в приложении В.

### 2.6.2 Разработка руководства пользователя

Руководство пользователя согласно ГОСТ РД 50-34.698-90- документ, в котором описывается, как пользоваться продуктом, включая установку на компьютер и работу с ним. В него также может быть включена информация об ограничениях на использование продукта и обратная связь с поддержкой.

Руководство пользователя представлено в приложении Г.

# 3 Безопасность жизнедеятельности

## 3.1 Требования к рабочим помещениям и рабочим местам оператора ПК

Рабочие помещения и рабочие места оператора ПК должны соответствовать определенным требованиям, которые обеспечат комфортную и безопасную работу. К ним относятся:

1. Размеры помещения не менее 6 квадратных метров на одно рабочее место.
2. Наличие естественного и искусственного освещения, соответствующего СанПиН 2.2.2./2.4.1340-03 «Гигиенические требования к условиям труда при работе на персональных компьютерах».
3. Температура воздуха в помещении должна находиться в пределах 18-24 градусов Цельсия, влажность - 40-60%.
4. Отсутствие шума и других нежелательных звуковых эффектов.
5. Наличие пожарных средств и путей эвакуации.
6. Наличие мебели, соответствующей правилам эргономики (регулируемая высота, регулируемый наклон стола, спинка и сиденье стула, наличие подставки для ног и т.д.).
7. Наличие специального оборудования для защиты здоровья (экраны, фильтры, подставки для документов и т.д.).
8. Соответствие параметров компьютера установленным нормам.
9. Работа оператора ПК должна проходить в условиях микроклимата, обеспечивающего наиболее комфортные условия для зрения (уровень освещенности должен быть не менее 300 лк).
10. Обеспечение необходимой пространственной ориентации с помощью различных методов размещения монитора и клавиатуры.

Общие требования при использовании компьютерной техники:

1. Защита зрения.
2. Защита органов слуха.
3. Защита органов пищеварения.
4. Защита кожных покровов.
5. Установка расписания работы и время отдыха в течение дня.

## 3.2 Расчет искусственной освещенности рабочего места оператора ПК

Расчет искусственной освещенности рабочего места оператора ПК является необходимым для обеспечения комфортных условий работы с учетом световых параметров, необходимых для нормального функционирования зрительного аппарата человека.

Согласно требованиям СанПиН 2.2.2./2.4.1340-03, искусственное освещение должно обеспечивать не менее 300 лк на поверхности стола и не менее 500 лк на поверхности клавиатуры и экрана монитора.

Для расчета искусственной освещенности рабочего места оператора ПК необходимо знать:

1. Площадь рабочего помещения (S).
2. Коэффициент использования светового потока (K) - показатель, показывающий, какая часть из общей мощности светильника будет использоваться для освещения рабочей поверхности.
3. Световой поток, выделяемый одним светильником (F).
4. Коэффициент освещенности помещения, отношение светового потока к площади помещения (E=F/S).

Искусственная освещенность оператора ПК рассчитывается по формуле:

*E x K x η x T = L, (3)*

гдеE – коэффициент освещенности помещения, лк/м²;

K – коэффициент использования светового потока, принимается примерно равным 0,5;

η - коэффициент светопропускания светопрозрачных элементов в помещении (окна и т.п.), принимается в пределах 0,7-0,9;

T- коэффициент светового рассеивания стен белого цвета, имеющих светлую поверхность или покрытие, принимается равным 0,7 - 0,9;

L – необходимая освещенность поверхности рабочего места оператора ПК, лк.

Согласно стандартам, коэффициент освещенности для помещения размером 15 квадратных метров должен составлять не менее 300 лк/м². Для достижения этого значения и энергоэффективной работы светильника, мы можем рассчитать оптимальную световую мощность и эффективность светильника.

Подставляем значения в формулу:

*300 лк/м² × 15 м² = 4500 лк*

Для достижения необходимой световой мощности в 4500 лм при коэффициенте использования светового потока K=0,5, мы можем пересчитать эффективность светильника по формуле:

*P = (F × K) / η*

*η = (F × K) / P*

*η = (4500 лм × 0,5) / 50 Вт = 45*

Таким образом, чтобы обеспечить достаточный уровень освещенности помещения 15 кв. м с общим световым потоком 4500 лм, достаточно использовать светильники с эффективностью 45 лм/Вт и мощностью 50 Вт. Кроме того, для получения энергоэффективного решения, можно использовать светодиодные светильники, которые потребляют меньше энергии при той же световой мощности.

## 3.3. Техника безопасности при работе на ПК

Современная жизнь невозможна без использования компьютеров и интернета. Тем не менее, работа на ПК может представлять опасность для здоровья человека, если не соблюдать правила безопасности. Такие заболевания, как синдром карпального канала, синдром "сухого глаза", головные боли и шум в ушах, связаны с длительным пребыванием за компьютером. В связи с этим, необходимо соблюдать следующие правила техники безопасности при работе на ПК:

1. Регулярно делать перерывы. При работе за компьютером необходимо делать перерыв каждые 45-60 минут. Во время перерывов рекомендуется выполнять упражнения для глаз и рук.
2. Правильно настроить рабочее место. Рабочее место должно быть правильно настроено, чтобы минимизировать риск различных заболеваний. Клавиатура должна быть расположена на уровне локтя, а экран - на уровне глаз.
3. Использовать эргономическую мебель. Для работы за компьютером рекомендуется использовать эргономические кресла и столы, которые помогают поддерживать правильную позу.
4. Соблюдать правильный режим работы. Необходимо соблюдать правильный режим работы и сна, чтобы избежать усталости и стресса, которые могут привести к заболеваниям.
5. Использовать программы для защиты глаз. Существует множество программ, которые помогают защитить глаза при работе за компьютером. Они уменьшают яркость экрана, фильтруют синий свет и т.д.
6. Использовать антивирусное программное обеспечение. Антивирусное программное обеспечение помогает защитить компьютер от вирусов и злонамеренных программ, которые могут повредить систему или украсть личную информацию.
7. Избегать монотонной нагрузки. Повторяющиеся действия могут привести к снижению производительности и здоровью. Для этого рекомендуется использовать различные программы и ресурсы, менять виды деятельности, чтобы предотвратить нагрузку на конкретную группу мышц.
8. Использовать безопасные пароли. Для защиты личной информации и конфиденциальных данных следует использовать сложные пароли, которые не легко угадать или взломать. Рекомендуется использовать комбинации букв, цифр и символов.
9. Создавать резервные копии данных. Для сохранения важной информации необходимо регулярно создавать резервные копии данных, чтобы предотвратить потерю или повреждение важных файлов.
10. Обновлять программное обеспечение. Регулярное обновление программного обеспечения помогает устранять уязвимости и предотвращать атаки злонамеренных программ. Рекомендуется устанавливать обновления операционной системы, браузера и антивирусного ПО.
11. Использовать безопасную сеть. При работе на публичных сетях необходимо использовать VPN для защиты информации, передаваемой между компьютером и сервером или другим устройством.
12. Избегать использования нелицензионного программного обеспечения. Использование нелицензионного программного обеспечения может привести к нарушению законодательства и повышенному риску заражения вирусами и злонамеренными программами.

В целом, соблюдение правил техники безопасности при работе на ПК имеет важное значение для сохранения здоровья и безопасности человека. Рекомендуется следовать простым правилам, чтобы избежать проблем, связанных с длительным пребыванием за компьютером.

# Заключение

Разработка интерактивной платформы для изучения языков программирования, ориентированной на IT-отдел АО «Апатит», позволила решить ключевые задачи, связанные с повышением эффективности корпоративного обучения. В рамках дипломной работы была создана система, объединяющая теоретические модули, практические задания и инструменты аналитики, адаптированные под специфику проектов компании. Реализация платформы подтвердила её соответствие поставленным целям: сокращение времени адаптации сотрудников, интеграция обучения в рабочие процессы и обеспечение персонализированных образовательных траекторий.

Анализ существующих решений показал, что универсальные образовательные ресурсы не учитывают корпоративные стандарты и потребности бизнеса. Разработанная платформа преодолевает этот разрыв за счёт модульной архитектуры, поддержки интерактивных сред программирования и интеграции с внутренними системами компании (Jira, GitLab). Это обеспечивает сотрудникам возможность отрабатывать навыки в условиях, максимально приближенных к реальным проектам, а руководителям — получать аналитику прогресса в формате, удобном для управления ресурсами.

Выбор технологий (веб-стек, PHP, MySQL) и объектно-ориентированного подхода к разработке позволил создать гибкую и масштабируемую систему. Проектирование на основе UML-диаграмм обеспечило чёткую структуру компонентов, а тестирование с использованием Selenium, PHPUnit и Cypress подтвердило стабильность работы всех модулей. Особое внимание было уделено безопасности данных, что соответствует корпоративным стандартам АО «Апатит».

Внедрение платформы принесёт компании значимые преимущества:

* Оптимизация бюджета на обучение за счёт замены внешних ресурсов кастомизированными решениями.
* Повышение качества IT-разработок благодаря практико-ориентированному обучению и симуляции реальных задач.
* Ускорение цифровой трансформации через укрепление компетенций сотрудников в актуальных технологиях (Python, Java, SQL).

Перспективы развития проекта включают внедрение AI-ассистента для генерации персональных рекомендаций, расширение списка поддерживаемых языков программирования и углубление интеграции с облачными сервисами компании. Успешная реализация платформы подтверждает её востребованность не только для АО «Апатит», но и для других предприятий, заинтересованных в цифровизации процессов обучения.

Таким образом, дипломная работа демонстрирует, что создание специализированных образовательных инструментов является критическим фактором для поддержания конкурентоспособности IT-отделов в условиях стремительного технологического прогресса.

# Библиография

1. Агальцов В.П., Титов В.М. Корпоративные системы электронного обучения: проектирование и внедрение. — М.: Горячая линия-Телеком, 2019. — 256 с.
2. Анализ эффективности LMS-платформ в корпоративном обучении / И.С. Петров, А.А. Семенов, Е.В. Козлова // Информационные технологии в образовании. — 2021. — № 3(45). — С. 45-54.
3. Брукс Ф. Мифический человеко-месяц, или Как создаются программные системы. — СПб.: Символ-Плюс, 2020. — 304 с.
4. ГОСТ Р 53620-2009 Информационно-коммуникационные технологии в образовании. Электронные образовательные ресурсы. Общие положения [Электронный ресурс]. URL: https://docs.cntd.ru/document/1200071156 (дата обращения: 15.06.2024).
5. Дейтел П., Дейтел Х. Технологии программирования на Java: создание интерактивных приложений. — М.: Вильямс, 2018. — 864 с.
6. Иванов Д.Ю., Кузнецов М.А. Веб-разработка на PHP и MySQL: полное руководство. — СПб.: Питер, 2022. — 480 с.
7. Кент Бек. Экстремальное программирование: разработка через тестирование. — М.: Диалектика, 2020. — 224 с.
8. Корпоративное обучение в цифровую эпоху: методы и инструменты / Под ред. А.В. Леонтьева. — М.: ИНФРА-М, 2021. — 198 с.
9. Крамп Д., Касперски К. JavaScript и jQuery: интерактивная веб-разработка. — М.: Эксмо, 2023. — 512 с.
10. Мартин Р. Чистая архитектура: искусство разработки программного обеспечения. — СПб.: Питер, 2021. — 352 с.
11. Мейер Б. Объектно-ориентированное конструирование программных систем. — М.: ДМК Пресс, 2019. — 688 с.
12. Методы адаптивного обучения в IT-образовании / О.Р. Сидорова, В.П. Тихомиров // Современные информационные технологии и IT-образование. — 2022. — Т. 18. — № 2. — С. 312-320.
13. Михеев Е.В. PHP 8: разработка веб-приложений. — М.: БХВ-Петербург, 2023. — 736 с.
14. Облачные технологии в корпоративном обучении: опыт внедрения / Н.А. Воробьева, Д.И. Соколов // Цифровая трансформация. — 2020. — № 4(12). — С. 67-75.
15. Официальная документация по Selenium [Электронный ресурс]. URL: https://www.selenium.dev/documentation/ (дата обращения: 15.06.2024).
16. Паттерны проектирования в обучении программированию / А.К. Лисовский, М.В. Новиков // Программная инженерия. — 2021. — № 12(5). — С. 213-220.
17. Робертс Э. Глубокое понимание JavaScript: от основ до асинхронного программирования. — М.: ДМК Пресс, 2022. — 608 с.
18. Современные тенденции в e-learning: персонализация и геймификация / Е.Л. Смирнова, А.Д. Волков // Открытое образование. — 2023. — № 1. — С. 34-42.
19. Столлингс В. Веб-технологии: принципы и инструменты. — М.: Вильямс, 2021. — 896 с.
20. Таненбаум Э., Вудхалл А. Современные операционные системы. — СПб.: Питер, 2020. — 1120 с.
21. Фаулер М. Рефакторинг: улучшение существующего кода. — М.: Диалектика, 2019. — 448 с.
22. Хорстман К. Java. Библиотека профессионала. Том 1. Основы. — М.: Диалектика, 2022. — 864 с.
23. Чамберс Д., Закас Н. Современный JavaScript для разработчиков. — М.: Питер, 2023. — 592 с.
24. Шмаков Н.П. Тестирование программного обеспечения: инструменты и методологии. — М.: Лаборатория знаний, 2020. — 320 с.
25. Экспертные системы в корпоративном обучении: анализ кейсов / В.Г. Иванов, Т.А. Кузнецова // Искусственный интеллект в образовании. — 2022. — № 3(17). — С. 89-97.