

**ОБЩЕСТВО С ОГРАНИЧЕННОЙ ОТВЕТСТВЕННОСТЬЮ
«ИННОПРОГ»**

УТВЕРЖДАЮ

Генеральный директор ООО

«ИННОПРОГ»



Венедиктов Р.В.

/ Венедиктов Р.В.

подпись

инициалы, фамилия

01 февраля 2025 г.

Приказ об утверждении
программы

от 01 февраля 2025 г. № ОБР-3

**ДОПОЛНИТЕЛЬНАЯ ПРОФЕССИОНАЛЬНАЯ ПРОГРАММА
ПРОФЕССИОНАЛЬНОЙ ПЕРЕПОДГОТОВКИ**

«Frontend-разработчик»

Вид программы:	дополнительная профессиональная программа профессиональной переподготовки
Новая квалификация:	Frontend-разработчик
Трудоемкость:	800 академических часов
Форма обучения:	очная, с применением электронного обучения и дистанционных образовательных технологий
Режим реализации:	10 месяцев (40 учебных недель), рекомендуемая нагрузка 20 академических часов в неделю
Язык обучения:	русский

Иннополис, 2025

Содержание

1	Общая характеристика программы	5
1.1	Наименование программы	5
1.2	Нормативно-правовые основания разработки программы	6
1.3	Структура образовательной программы	7
1.4	Вид программы и целевая модель результата	8
1.5	Актуальность программы	9
1.6	Цель реализации программы	10
1.7	Задачи программы	10
1.8	Категория слушателей	11
1.9	Требования к уровню подготовки поступающего на обучение	11
1.10	Порядок приема, зачисления, отчисления, пересдачи и апелляции	12
1.11	Форма обучения, режим и язык обучения	13
1.12	Срок освоения и трудоемкость программы	13
1.13	Документ о квалификации	14
1.14	Характеристика новой квалификации и связанной с ней профессиональной деятельности	15
1.15	Связь Программы с профессиональными стандартами, трудовыми функциями и уровнями квалификации	16
1.16	Планируемые результаты освоения программы	17
2	Содержание программы	21
2.1	Учебный план	21
2.2	Календарный учебный график	22
2.3	Рабочие программы модулей	24
	Модуль 1. Вводный организационно-методический модуль	24
	Модуль 2. HTML, CSS и адаптивная верстка	26
	Модуль 3. JavaScript и TypeScript для frontend-разработки	29
	Модуль 4. Git, командная разработка и качество кода	32
	Модуль 5. Инструменты разработки, сборка и Linux	34
	Модуль 6. React и компонентная архитектура	37
	Модуль 7. Маршрутизация, управление состоянием и работа с API	41
	Модуль 8. Тестирование, доступность, производительность и безопасность frontend-приложений	44
	Модуль 9. Дизайн-системы, UI-kit и командная frontend-практика	47

Модуль 10. Проектная практика и консультации	50
2.4 Проектная практика и консультации	52
3 Организационно-педагогические условия реализации программы	55
3.1 Общие условия реализации	55
3.2 Материально-технические условия	55
3.3 Электронная информационно-образовательная среда	57
3.4 Кадровые условия	58
3.5 Учебно-методическое обеспечение	59
3.6 Применение электронного обучения и дистанционных образо- вательных технологий	59
3.7 Информационная безопасность, персональные данные и этика .	60
3.8 Локальные процедуры обучения	61
3.9 Обновление программы	62
4 Оценка качества освоения программы	63
4.1 Общие положения	63
4.2 Текущий контроль успеваемости	63
4.3 Промежуточная аттестация	64
4.4 Пересдача, апелляция и подтверждение самостоятельности . . .	65
4.5 Допуск к итоговой аттестации	65
4.6 Итоговая аттестация	66
4.7 Внутренняя и внешняя оценка качества реализации программы	69
5 Методические материалы	70
5.1 Методические указания для слушателей	70
5.2 Методические указания для преподавателей и проверяющих . .	70
5.3 Рекомендуемая литература и ресурсы	71
Приложение 1. Матрица формирования компетенций	73
Приложение 2. Оценочные материалы промежуточной аттестации .	75
Приложение 3. Оценочные материалы итоговой аттестации	76
Приложение 4. Перечень локальных нормативных актов, на которые должна опираться реализация программы	85
Приложение 5. Лист актуализации программы	86
Приложение 6. Требования к репозиторию, README и демонстрации	87
Приложение 7. Дорожная карта подготовки выпускного проекта . .	90

Приложение 8. Контрольный лист соответствия требованиям программы	92
--	-----------

1. Общая характеристика программы

1.1. Наименование программы

Дополнительная профессиональная программа профессиональной переподготовки «Frontend-разработчик» разработана для подготовки специалистов, способных проектировать, разрабатывать, тестировать, оптимизировать и сопровождать клиентские части веб-приложений на основе HTML, CSS, JavaScript, TypeScript и React. Программа ориентирована на практическое освоение полного цикла frontend-разработки: от анализа пользовательского сценария и макета до реализации компонентной архитектуры, интеграции с API, тестирования, сборки, публикации и защиты итогового проекта.

Программа имеет прикладную направленность. Освоение материала строится через последовательное выполнение учебных проектов, лабораторных работ, зачетных заданий и итогового frontend-проекта. Каждый результат слушателя фиксируется в репозитории, сопровождается README-файлом, инструкцией по запуску, демонстрационными материалами и проверкой по единым критериям качества.

Продолжение на следующей странице

Продолжение таблицы 2

Показатель	Содержание
------------	------------

Таблица 1 – Основные сведения о программе

Показатель	Содержание
Наименование программы	«Frontend-разработчик»
Вид программы	дополнительная профессиональная программа профессиональной переподготовки
Направленность	разработка клиентских частей веб-приложений, пользовательских интерфейсов, SPA-приложений и frontend-инфраструктуры
Новая квалификация	Frontend-разработчик
Объем программы	800 академических часов
Срок освоения	10 месяцев (40 учебных недель)
Режим занятий	рекомендуемая нагрузка 20 академических часов в неделю
Форма итоговой аттестации	защита итогового frontend-проекта
Документ о квалификации	диплом о профессиональной переподготовке установленного образца при успешном освоении программы

1.2. Нормативно-правовые основания разработки программы

Программа разработана и утверждается организацией самостоятельно с учетом требований законодательства Российской Федерации об образовании, порядка реализации дополнительных профессиональных программ, правил применения электронного обучения и дистанционных образовательных технологий, требований к локальным актам образовательной организации и квалификаци-

онных ориентиров профессиональной деятельности в сфере информационных технологий.

- Федеральный закон от 29.12.2012 № 273-ФЗ «Об образовании в Российской Федерации»;
- Приказ Министерства науки и высшего образования Российской Федерации от 24.03.2025 № 266 «Об утверждении Порядка организации и осуществления образовательной деятельности по дополнительным профессиональным программам»;
- Постановление Правительства Российской Федерации от 11.10.2023 № 1678 «Об утверждении Правил применения организациями, осуществляющими образовательную деятельность, электронного обучения, дистанционных образовательных технологий при реализации образовательных программ»;
- Постановление Правительства Российской Федерации от 18.09.2020 № 1490 «О лицензировании образовательной деятельности»;
- ГОСТ Р 7.0.97-2025 «Система стандартов по информации, библиотечному и издательскому делу. Организационно-распорядительная документация. Требования к оформлению документов»;
- профессиональный стандарт «Разработчик Web и мультимедийных приложений», утвержденный приказом Минтруда России от 18.01.2017 № 44н;
- профессиональный стандарт «Программист», применяемый как дополнительный квалификационный ориентир для задач разработки программного кода, тестирования и сопровождения программных компонентов;
- локальные нормативные акты ООО «ИННОПРОГ», регламентирующие прием, обучение, электронную информационно-образовательную среду, текущий контроль, промежуточную аттестацию, итоговую аттестацию, выдачу документов о квалификации, обработку персональных данных и внесение сведений в ФИС ФРДО.

1.3. Структура образовательной программы

Настоящая Программа «Frontend-разработчик» оформлена как комплекс основных характеристик образования, организационно-педагогических условий,

оценочных и методических материалов, необходимых для реализации дополнительной профессиональной программы профессиональной переподготовки.

В состав Программы включены:

- общая характеристика Программы, включая цель, задачи, категорию слушателей, требования к поступающим, форму обучения, срок освоения, трудоемкость, документ о квалификации и характеристику новой квалификации;
- учебный план, определяющий перечень модулей, трудоемкость, распределение часов по видам учебной работы, формы промежуточной и итоговой аттестации;
- календарный учебный график, определяющий рекомендуемую последовательность освоения модулей и период итоговой аттестации;
- рабочие программы модулей, раскрывающие цели, результаты, тематическое содержание, самостоятельную работу, проектные задания и формы контроля;
- организационно-педагогические условия реализации, включая кадровые, материально-технические, информационно-методические условия, электронную информационно-образовательную среду и порядок применения электронного обучения и дистанционных образовательных технологий;
- оценочные материалы для текущего контроля, промежуточной и итоговой аттестации, критерии допуска, пересдачи, апелляции и подтверждения самостоятельности выполнения работ;
- методические материалы и приложения, обеспечивающие единообразное применение Программы, проверку проектных работ, оформление результатов и актуализацию документа.

Структура Программы обеспечивает проверяемость объема, содержания, планируемых результатов, условий реализации, форм аттестации и документов, выдаваемых по итогам обучения.

1.4. Вид программы и целевая модель результата

Программа относится к дополнительным профессиональным программам профессиональной переподготовки. Освоение программы направлено на получение компетенции, необходимой для выполнения нового вида профессиональ-

ной деятельности в области frontend-разработки. Целевая модель результата предполагает, что выпускник способен реализовать клиентскую часть веб-приложения по техническому заданию, поддерживать качество кода, работать с командным репозиторием, подключать внешние API, применять React, TypeScript, маршрутизацию, управление состоянием, тестирование и базовые средства публикации приложения.

Подготовка строится от фундаментальных основ веб-разработки к разработке сложных интерфейсов. Сначала слушатель осваивает семантическую разметку, стилизацию, адаптивную верстку и JavaScript. Далее формируется умение проектировать типизированный код, работать с React, выделять компоненты, строить маршруты, управлять состоянием, обрабатывать данные, проверять приложение тестами, оптимизировать производительность и готовить проект к демонстрации.

1.5. Актуальность программы

Frontend-разработка является одной из ключевых областей создания цифровых продуктов. Пользовательский интерфейс определяет доступность сервиса, скорость выполнения сценариев, качество восприятия продукта, эффективность продаж, удобство внутренних бизнес-процессов и возможность масштабирования цифрового решения. Современный frontend-разработчик работает на стыке программирования, проектирования интерфейсов, веб-стандартов, командной разработки, интеграции с API, безопасности, производительности и сопровождения пользовательских сценариев.

Рынок требует специалистов, которые владеют современной архитектурой клиентских приложений. Для практической работы необходимы навыки TypeScript, React, компонентного проектирования, работы с формами, маршрутизацией, состоянием, HTTP-запросами, обработкой ошибок, тестированием интерфейсов, доступностью, оптимизацией загрузки и публикацией приложения. Программа закрывает эти потребности через комплексную траекторию обучения и систему проектных работ.

1.6. Цель реализации программы

Цель программы состоит в формировании у слушателей профессиональных компетенций, необходимых для выполнения нового вида деятельности в сфере frontend-разработки, включая создание адаптивных интерфейсов, разработку интерактивных веб-приложений на JavaScript и TypeScript, применение React, интеграцию с API, тестирование, оптимизацию, документирование и защиту законченного frontend-проекта.

1.7. Задачи программы

Для достижения цели программа решает следующие образовательные и практико-ориентированные задачи:

- сформировать понимание устройства веб-приложений, клиент-серверного взаимодействия, жизненного цикла frontend-проекта и роли frontend-разработчика в команде;
- обеспечить владение HTML, CSS, адаптивной версткой, семантической разметкой, формами, медиазапросами, Flexbox, Grid, препроцессорами и utility-first подходом к стилям;
- сформировать уверенное владение JavaScript и TypeScript для разработки интерактивной логики, работы с DOM, асинхронными операциями, HTTP-запросами и типизированными структурами данных;
- научить применять Git, ветвление, pull request, code review, правила оформления коммитов, ведение issue и командный рабочий процесс;
- сформировать навыки настройки окружения разработки, использования Node.js, пакетных менеджеров, Vite, ESLint, Prettier, переменных окружения, Linux-команд и базовой публикации приложения;
- обеспечить практическое освоение React, JSX, компонентов, props, state, эффектов, пользовательских хуков, композиции, форм и архитектурной декомпозиции интерфейса;
- научить проектировать маршрутизацию, управлять состоянием, организовывать работу с API, обрабатывать загрузку, ошибки, пустые состояния и права доступа пользователя;

- сформировать навыки проверки качества frontend-приложения через модульные, компонентные и end-to-end тесты, статический анализ, доступность, производительность и базовую безопасность;
- научить оформлять проектную документацию, README, инструкции по запуску, демонстрационные материалы, чек-листы проверки и защиту технического решения;
- подготовить слушателя к выполнению итогового проекта, который подтверждает способность реализовать полноценное frontend-приложение с понятной архитектурой, корректной логикой, тестами и демонстрацией результата.

1.8. Категория слушателей

К освоению программы допускаются лица, имеющие среднее профессиональное и/или высшее образование, а также лица, получающие среднее профессиональное и/или высшее образование. Программа может быть полезна начинающим разработчикам, специалистам смежных ИТ-направлений, дизайнерам интерфейсов, аналитикам, менеджерам цифровых продуктов и слушателям, которые планируют перейти к профессиональной разработке клиентских частей веб-приложений.

Для успешного освоения программы желательно наличие базовой компьютерной грамотности, умения работать с файловой системой, браузером, текстовым редактором, онлайн-сервисами и учебной платформой. Специальная подготовка по программированию желательна. Стартовые разделы программы включают повторение ключевых основ, необходимых для дальнейшего изучения JavaScript, TypeScript и React.

1.9. Требования к уровню подготовки поступающего на обучение

Таблица 3 – Требования к поступающим на обучение

Группа требований	Содержание требований
Образовательный уровень	наличие или получение среднего профессионального и/или высшего образования
Компьютерная грамотность	работа с браузером, файлами, архивами, облачными сервисами, электронной почтой и учебной платформой
Техническая готовность	наличие компьютера или ноутбука, стабильного доступа в интернет, возможности установить редактор кода, Git, Node.js и браузеры для тестирования
Мотивационная готовность	готовность выполнять практические задания, вести репозиторий, исправлять замечания, участвовать в консультациях и защищать проект
Языковая готовность	готовность работать с русскоязычными учебными материалами и базовой англоязычной терминологией веб-разработки

1.10. Порядок приема, зачисления, отчисления, пересдачи и апелляции

Прием слушателей осуществляется на основании заявления, договора об образовании и документов, подтверждающих наличие или получение среднего профессионального и/или высшего образования. Зачисление оформляется приказом организации. Обучение проводится в соответствии с утвержденной программой, расписанием, учебным планом, календарным учебным графиком и локальными актами организации.

Слушатель может быть отчислен по основаниям, предусмотренным договором об образовании и локальными актами организации, включая завершение обучения, невыполнение учебного плана, нарушение правил обучения или личное заявление. Пересдача промежуточной и итоговой аттестации допускается в порядке, установленном локальным актом. Апелляция по результатам аттестации рассматривается комиссией или уполномоченным должностным лицом с учетом представленных материалов, репозитория, истории сдачи и критериев оценивания.

1.11. Форма обучения, режим и язык обучения

Обучение по Программе осуществляется в очной форме с применением электронного обучения и дистанционных образовательных технологий. Электронное обучение и дистанционные образовательные технологии применяются при проведении учебных занятий, консультаций, самостоятельной работы, текущего контроля, промежуточной аттестации, проектной практики и итоговой аттестации при условии обеспечения доступа слушателя к электронной информационно-образовательной среде, учебным материалам, заданиям, средствам коммуникации и процедурам идентификации слушателя.

Таблица 5 – Форма, режим и язык обучения

Показатель	Значение
Применяемые технологии	электронное обучение и дистанционные образовательные технологии
Режим реализации	10 месяцев (40 учебных недель)
Рекомендуемая нагрузка	20 академических часов в неделю
Язык обучения	русский
Основные формы учебной работы	видеолекции, практические задания, лабораторные работы, самостоятельная работа, консультации, зачетные задания, проектная практика, итоговая защита

1.12. Срок освоения и трудоемкость программы

Общая трудоемкость программы составляет 800 академических часов. Один академический час принимается равным 45 минутам. Срок освоения программы составляет 10 месяцев (40 учебных недель) при рекомендуемой нагрузке 20 академических часов в неделю. Индивидуальный темп освоения может уточняться локальными актами организации и условиями договора об образовании.

Таблица 7 – Распределение трудоемкости программы

Показатель	Значение	Пояснение
Общая трудоемкость	800 часов	включает лекции, практику, самостоятельную работу, контроль, проектную практику и итоговую аттестацию
Срок освоения	10 месяцев (40 недель)	при рекомендуемой нагрузке 20 академических часов в неделю
Практико-ориентированная часть	более половины программы	включает лабораторные работы, мини-проекты, проектную практику, тестирование и защиту
Итоговая аттестация	24 часа	подготовка, проверка, защита и оформление результатов итогового frontend-проекта

1.13. Документ о квалификации

Лицам, успешно освоившим программу и прошедшим итоговую аттестацию, выдается диплом о профессиональной переподготовке установленного образца. Слушателям, получающим среднее профессиональное или высшее образование, диплом о профессиональной переподготовке выдается после представления документа об основном образовании в порядке, установленном законодательством и локальными актами организации.

Сведения о выданных документах о квалификации передаются в федеральную информационную систему «Федеральный реестр сведений о документах об образовании и (или) о квалификации, документах об обучении» в порядке и сроки, установленные законодательством Российской Федерации и локальными актами организации.

1.14. Характеристика новой квалификации и связанной с ней профессиональной деятельности

Frontend-разработчик выполняет работы по созданию пользовательских интерфейсов веб-приложений, разработке клиентской логики, интеграции с серверными API, реализации компонентной архитектуры, обеспечению доступности, производительности и сопровождаемости интерфейса. Выпускник программы готов участвовать в разработке лендингов, личных кабинетов, административных панелей, сервисных интерфейсов, интерактивных каталогов, дашбордов, SPA-приложений и прототипов цифровых продуктов.

Таблица 9 – Характеристика профессиональной деятельности выпускника

Направление деятельности	Содержание деятельности
Верстка интерфейсов	создание семантической, адаптивной и кроссбраузерной разметки по макету и пользовательскому сценарию
Клиентская логика	реализация интерактивного поведения, обработка событий, работа с формами, данными и состоянием интерфейса
React-разработка	проектирование компонентов, пользовательских хуков, страниц, маршрутов, экранов и переиспользуемых UI-элементов
Интеграция с API	отправка HTTP-запросов, обработка загрузки, ошибок, пустых состояний, пагинации, фильтрации, авторизации и обновления данных
Качество кода	использование TypeScript, линтеров, форматирования, тестов, code review, документации и единых правил репозитория
Публикация и сопровождение	сборка проекта, настройка переменных окружения, базовый деплой, анализ производительности и устранение дефектов

1.15. Связь Программы с профессиональными стандартами, трудовыми функциями и уровнями квалификации

Содержание программы согласовано с квалификационными ориентирами профессиональной деятельности в области веб-разработки. Основным ориентиром является профессиональный стандарт «Разработчик Web и мультимедийных приложений», который отражает деятельность по проектированию, разработке, интеграции и сопровождению веб-ресурсов и мультимедийных приложений. Дополнительно учитываются элементы профессионального стандарта «Программист», связанные с разработкой, тестированием, документированием и сопровождением программного кода.

Таблица 11 – Связь модулей программы с квалификационными ориентирами

Блок программы	Профессиональный ориентир	Подтверждаемый результат
HTML, CSS и адаптивная верстка	разработка пользовательских интерфейсов и веб-страниц	адаптивная страница по макету с семантической структурой, валидной разметкой и корректным отображением
JavaScript и TypeScript	разработка клиентской логики веб-приложения	типизированные функции, обработчики событий, работа с DOM, API и асинхронными операциями
React и компонентная архитектура	разработка компонентов и интерактивных веб-приложений	SPA-приложение с компонентной структурой, маршрутизацией, состоянием и обработкой данных

Продолжение на следующей странице

Продолжение таблицы 12

Блок программы	Профессиональный ориентир	Подтверждаемый результат
Тестирование и качество	проверка, отладка и сопровождение программного кода	набор тестов, чек-лист качества, исправленные дефекты, документация по запуску
Проектная практика	выполнение законченного проекта по техническому заданию	итоговый проект в репозитории, README, сборка, демонстрация, защита технического решения

1.16. Планируемые результаты освоения программы

В результате освоения программы слушатель должен знать основы веб-стандартов, клиент-серверного взаимодействия, JavaScript, TypeScript, React, командной разработки и проверки качества frontend-приложений. Слушатель должен уметь реализовывать адаптивный интерфейс, создавать компоненты, подключать API, управлять состоянием, писать тесты, анализировать ошибки, оформлять репозиторий и защищать проектное решение.

Продолжение на следующей странице

Продолжение таблицы 14

Код	Компетенция	Результат освоения
-----	-------------	--------------------

Таблица 13 – Планируемые результаты освоения программы

Код	Компетенция	Результат освоения
ПК-1	Проектирование структуры пользовательского интерфейса	слушатель анализирует макет, пользовательский сценарий и техническое задание, выделяет страницы, компоненты, состояния и ограничения интерфейса
ПК-2	Разработка семантической и адаптивной верстки	слушатель создает HTML-структуру, CSS-стили, адаптивные сетки, формы и визуальные состояния с учетом доступности
ПК-3	Разработка клиентской логики на JavaScript	слушатель реализует интерактивное поведение, обработчики событий, работу с DOM, коллекциями, асинхронными запросами и ошибками
ПК-4	Применение TypeScript в frontend-разработке	слушатель описывает типы, интерфейсы, generics, модели данных, пропсы компонентов, ответы API и состояния приложения
ПК-5	Командная разработка и Git-процесс	слушатель ведет репозиторий, оформляет ветки, коммиты, pull request, исправляет замечания и поддерживает читаемую историю изменений

Продолжение на следующей странице

Продолжение таблицы 14

Код	Компетенция	Результат освоения
ПК-6	Настройка окружения и сборки frontend-проекта	слушатель использует Node.js, пакетные менеджеры, Vite, переменные окружения, линтеры, форматирование и базовый деплой
ПК-7	Разработка приложений на React	слушатель создает компоненты, страницы, хуки, формы, пользовательские сценарии и переиспользуемые элементы интерфейса
ПК-8	Маршрутизация, состояние и интеграция с API	слушатель организует маршруты, клиентское состояние, серверные данные, загрузку, ошибки, авторизацию и обновление интерфейса
ПК-9	Обеспечение качества, доступности и производительности	слушатель применяет тесты, статический анализ, DevTools, Lighthouse, базовые правила WCAG и оптимизацию загрузки
ПК-10	Выполнение и защита frontend-проекта	слушатель реализует законченный проект, оформляет документацию, демонстрирует работу приложения и аргументирует технические решения

Таблица 15 – Знания, умения и практический опыт выпускника

Код	Знать	Уметь	Практический опыт
ЗУП-1	назначение HTML, CSS, JavaScript, TypeScript, React, браузера, HTTP и API	выбирать технологию под задачу и объяснять архитектуру интерфейса	разбор технического задания и декомпозиция интерфейса
ЗУП-2	семантические теги, формы, доступность, каскад, Flexbox, Grid, медиазапросы	создавать адаптивные страницы и компоненты по макету	верстка лендинга, формы, карточек, таблиц и навигации
ЗУП-3	синтаксис JavaScript, замыкания, модули, промисы, async/await, fetch	реализовывать интерактивную логику и обмен данными	мини-приложения с обработкой событий и запросами к API
ЗУП-4	типы TypeScript, интерфейсы, union-типы, generics, типизацию компонентов	снижать количество ошибок через типизацию данных и пропсов	типизированный React-проект
ЗУП-5	структуру React-приложения, JSX, props, state, hooks, routing	строить компонентную архитектуру и управлять пользовательским сценарием	SPA-приложение с несколькими страницами и состояниями
ЗУП-6	основы тестирования, доступности, безопасности и производительности	писать тесты, находить дефекты, улучшать UX и скорость работы	проект с тестами, проверкой Lighthouse и чек-листом доступности

2. Содержание программы

2.1. Учебный план

Учебный план построен по принципу последовательного усложнения. Каждый модуль завершает конкретный практический результат: страница, компонент, мини-приложение, репозиторий, React-приложение, тестовый набор, прототип с API или фрагмент итогового проекта. Промежуточная аттестация проводится в форме зачета по модулю. Итоговая аттестация проводится в форме защиты итогового frontend-проекта.

Таблица 17 – Учебный план программы

№	Наименование раздела, дисциплины, модуля	Вс. ч.	Лек. ч.	Прак. ч.	Сам. раб.	Конт- роль	Атт.
1	Вводный организационно-методический модуль	12	6	0	6	0	зачет
2	HTML, CSS и адаптивная верстка	104	29	49	20	6	зачет
3	JavaScript и TypeScript для frontend-разработки	124	34	59	25	6	зачет
4	Git, командная разработка и качество кода	56	14	25	11	6	зачет
5	Инструменты разработки, сборка и Linux	56	14	25	11	6	зачет
6	React и компонентная архитектура	148	40	74	28	6	зачет

Продолжение на следующей странице

Продолжение таблицы 18

№	Наименование раздела, дисциплины, модуля	Вс. ч.	Лек. ч.	Прак. ч.	Сам. раб.	Контроль	Атт.
7	Маршрутизация, управление состоянием и работа с API	104	26	52	20	6	зачет
8	Тестирование, доступность, производительность и безопасность frontend-приложений	68	17	34	11	6	зачет
9	Дизайн-системы, UI-kit и командная frontend-практика	36	9	18	6	3	зачет
10	Проектная практика и консультации	56	0	42	14	0	зачет
11	Итоговая аттестация	36	0	0	0	36	защита проекта
	Итого	800	189	378	152	81	

2.2. Календарный учебный график

Календарный учебный график рассчитан на 10 месяцев (40 учебных недель). Организация может уточнять календарные даты занятий, консультаций и контрольных мероприятий с сохранением общей трудоемкости, последовательности модулей и требований к результатам освоения программы.

Таблица 19 – Календарный учебный график

№	Недели	Модуль	Час.	Контрольный результат
1	1	Вводный организационно-методический модуль	12	изучение регламента, настройка учебной среды, входная диагностика
2	2–6	HTML, CSS и адаптивная верстка	104	лендинг, форма, адаптивная сетка, UI-состояния
3	7–12	JavaScript и TypeScript для frontend-разработки	124	интерактивные мини-приложения, типизированная модель данных
4	13–15	Git, командная разработка и качество кода	56	репозиторий, ветки, pull request, code review
5	16–18	Инструменты разработки, сборка и Linux	56	сборка проекта, переменные окружения, базовый деплой
6	19–25	React и компонентная архитектура	148	SPA-приложение с компонентной структурой и формами
7	26–30	Маршрутизация, управление состоянием и работа с API	104	приложение с маршрутизацией, серверными данными и обработкой ошибок
8	31–33	Тестирование, доступность, производительность и безопасность	68	тесты, чек-лист WCAG, Lighthouse, исправление дефектов
9	34–35	Дизайн-системы, UI-kit и командная практика	36	мини UI-kit, документация компонентов, Storybook-сценарии
10	36–38	Проектная практика и консультации	56	разработка и подготовка итогового frontend-проекта

Продолжение на следующей странице

Продолжение таблицы 20

№	Недели	Модуль	Час.	Контрольный результат
11	39–40	Итоговая аттестация	36	защита проекта, собеседование и оценка результатов

2.3. Рабочие программы модулей

Рабочие программы модулей раскрывают содержание обучения, практическую подготовку, контрольные результаты, типовые ошибки и требования к зачету. Каждый модуль связан с проектной траекторией. Результаты модулей могут использоваться как части итогового frontend-проекта.

Модуль 1. Вводный организационно-методический модуль

Трудоемкость модуля составляет 12 академических часов. Модуль вводит слушателя в структуру программы, правила обучения, порядок работы с электронной информационно-образовательной средой, требования к репозиториям, проектной траектории, промежуточной и итоговой аттестации.

Продолжение на следующей странице

Продолжение таблицы 22

№	Тема	Час.	Лек. ч.	Прак. ч.	Сам. раб.	Конт- роль
---	------	------	------------	-------------	--------------	---------------

Таблица 21 – Тематический план: Модуль 1. Вводный организационно-методический модуль

№	Тема	Час.	Лек. ч.	Прак. ч.	Сам. раб.	Конт- роль
1	Цели программы, структура модулей, итоговый результат и требования к проекту	2	1	0	1	опрос
2	Электронная информационно-образовательная среда, расписание, консультации, фиксация результатов	2	1	0	1	проверка доступа
3	Рабочее место frontend-разработчика: редактор кода, браузер, терминал, Git, Node.js	2	1	0	1	самопроверка
4	Входная диагностика и индивидуальная траектория освоения	2	1	0	1	диагностика

Практическая подготовка.

- создание учебной папки и проверка доступа к платформе;
- установка или проверка редактора кода, браузера, терминала, Git и Node.js;
- заполнение входной анкеты о начальных навыках и целях обучения;
- создание шаблона индивидуального плана проекта.

Планируемый практический результат.

- слушатель понимает структуру программы и последовательность контрольных мероприятий;
- рабочее окружение подготовлено к выполнению практических заданий;
- слушатель знает правила оформления репозитория, сдачи работ и получения обратной связи.

Типовые ошибки, подлежащие исправлению.

- отсутствует доступ к учебной платформе или репозиторию;
- рабочая среда установлена частично;
- слушатель не ознакомился с критериями оценки и сроками сдачи работ.

Условия зачета. Зачет выставляется при подтвержденном доступе к учебной среде, выполнении входной диагностики и готовности рабочего окружения.

Таблица 23 – Уровни освоения по модулю: Модуль 1. Вводный организационно-методический модуль

Уровень освоения	Минимально достаточный результат	Повышенный результат
Минимальный	выполнены обязательные задания, проект запускается, основные требования соблюдены, критические ошибки устранены	слушатель может объяснить основные решения и исправить замечания после проверки
Повышенный	реализованы дополнительные сценарии, код структурирован, учтены граничные состояния, оформлена документация	слушатель аргументирует архитектуру, применяет тесты или автоматические проверки и показывает устойчивость решения

Модуль 2. HTML, CSS и адаптивная верстка

Трудоемкость модуля составляет 104 академических часов. Модуль формирует базу frontend-разработки: семантическая HTML- разметка, структура документа, формы, CSS, каскад, позиционирование, Flexbox, Grid, адаптивность, до-

ступность интерфейса, оформление макета и подготовка верстки к интеграции с JavaScript и React.

Таблица 25 – Тематический план: Модуль 2. HTML, CSS и адаптивная верстка

№	Тема	Час.	Лек. ч.	Прак. ч.	Сам. раб.	Конт- роль
1	Устройство веб-страницы, роль браузера, DOM, CSSOM, HTTP и статические ресурсы	6	2	2	2	практикум
2	Семантическая HTML-разметка: структура документа, заголовки, секции, списки, ссылки, изображения	8	2	4	2	верстка фрагмента
3	Формы, поля ввода, базовая валидация, состояния ошибок и доступные подписи	8	2	4	2	форма
4	CSS: каскад, специфичность, единицы измерения, box model, переменные и наследование	8	2	4	2	задачи
5	Flexbox, Grid, позиционирование, слои, переполнение и типовые макетные задачи	10	3	5	2	макет
6	Адаптивная верстка: mobile-first, медиазапросы, резиновые сетки, изображения и breakpoints	10	3	5	2	адаптив
7	Методологии и инструменты стилизации: БЭМ, Sass, CSS Modules, Tailwind CSS, токены	8	2	4	2	рефакторинг

Продолжение на следующей странице

Продолжение таблицы 26

№	Тема	Час.	Лек. ч.	Прак. ч.	Сам. раб.	Конт- роль
8	Базовая доступность: контраст, фокус, клавиатурная навигация, альтернативные тексты, ARIA-атрибуты	6	2	2	2	чек-лист
9	Промежуточная аттестация по модулю	4	0	0	0	зачет

Практическая подготовка.

- сверстать адаптивный лендинг по макету с несколькими секциями;
- реализовать форму с полями, визуальными состояниями, подсказками и сообщениями об ошибках;
- подготовить компонентную сетку карточек с адаптацией под мобильный, планшетный и десктопный экран;
- провести самопроверку по чек-листу семантики, доступности, адаптивности и кроссбраузерности.

Планируемый практический результат.

- страница соответствует макету, корректно отображается на разных ширинах экрана и использует семантическую структуру;
- стили организованы предсказуемо, классы именованы единообразно, повторяющиеся значения вынесены в переменные или токены;
- интерактивные элементы имеют состояния hover, focus, active, disabled и понятные подписи.

Типовые ошибки, подлежащие исправлению.

- верстка ломается при изменении ширины экрана;
- разметка построена только на универсальных контейнерах без семантики;
- форма не имеет подписей и состояний ошибки;

- стили дублируются, классы названы случайно, отсутствует единый подход к структуре CSS.

Условия зачета. Зачет проводится через защиту адаптивной страницы или набора интерфейсных секций. Проверяются семантика, адаптивность, качество CSS, доступность, соответствие макету и аккуратность репозитория.

Таблица 27 – Уровни освоения по модулю: Модуль 2. HTML, CSS и адаптивная верстка

Уровень освоения	Минимально достаточный результат	Повышенный результат
Минимальный	выполнены обязательные задания, проект запускается, основные требования соблюдены, критические ошибки устранены	слушатель может объяснить основные решения и исправить замечания после проверки
Повышенный	реализованы дополнительные сценарии, код структурирован, учтены граничные состояния, оформлена документация	слушатель аргументирует архитектуру, применяет тесты или автоматические проверки и показывает устойчивость решения

Модуль 3. JavaScript и TypeScript для frontend-разработки

Трудоемкость модуля составляет 124 академических часов. Модуль обеспечивает владение языковой базой frontend-разработчика. Слушатель изучает JavaScript, работу с DOM, функции, массивы, объекты, модули, асинхронные операции, fetch API, обработку ошибок и основы TypeScript для надежной типизации данных и интерфейсной логики.

Таблица 29 – Тематический план: Модуль 3. JavaScript и TypeScript для frontend-разработки

№	Тема	Час.	Лек. ч.	Прак. ч.	Сам. раб.	Конт- роль
1	Синтаксис JavaScript, типы данных, выражения, операторы, условия и циклы	8	2	4	2	задачи
2	Функции, области видимости, замыкания, стрелочные функции и чистые функции	8	2	4	2	практикум
3	Массивы, объекты, деструктуризация, spread/rest, методы коллекций и преобразование данных	10	3	5	2	задачи
4	DOM, события, делегирование, формы, валидация и динамическое обновление интерфейса	10	3	5	2	мини-проект
5	Модули, импорт и экспорт, структура файлов, работа с npm-пакетами	8	2	4	2	практикум
6	Асинхронность: callback, Promise, async/await, fetch, обработка загрузки и ошибок	12	3	6	3	API-проект
7	TypeScript: типы, интерфейсы, union-типы, generics, enum, типизация функций	12	4	6	2	типизация
8	TypeScript во frontend-коде: модели API, типизация DOM, конфигурация проекта, строгий режим	10	3	5	2	рефакторинг

Продолжение на следующей странице

Продолжение таблицы 30

№	Тема	Час.	Лек. ч.	Прак. ч.	Сам. раб.	Конт- роль
9	Самостоятельная доработка, исправление замечаний, оформление README	6	2	3	1	проверка
10	Промежуточная аттестация по модулю	4	0	0	0	зачет

Практическая подготовка.

- разработать мини-приложение на JavaScript с фильтрацией, сортировкой и сохранением данных;
- подключить публичный API и отобразить данные с состояниями загрузки, ошибки и пустого результата;
- переписать часть проекта на TypeScript с типизацией данных, функций и структур ответа API;
- оформить модульную структуру файлов и инструкцию по запуску.

Планируемый практический результат.

- слушатель понимает различия между значениями, ссылками, областями видимости и асинхронным выполнением;
- интерактивная логика отделена от разметки, функции имеют понятные имена и ограниченную ответственность;
- данные API типизированы, ошибки обрабатываются, пользователь получает понятные сообщения.

Типовые ошибки, подлежащие исправлению.

- глобальные переменные используются без необходимости;
- асинхронные ошибки не обрабатываются;
- типизация TypeScript сведена к универсальному типу any;

- код содержит дублирование и смешивает получение данных, преобразование и отрисовку.

Условия зачета. Зачет проводится через мини-приложение на JavaScript/TypeScript. Проверяются корректность логики, типизация, работа с API, обработка ошибок, структура файлов, читаемость кода и документация.

Таблица 31 – Уровни освоения по модулю: Модуль 3. JavaScript и TypeScript для frontend-разработки

Уровень освоения	Минимально достаточный результат	Повышенный результат
Минимальный	выполнены обязательные задания, проект запускается, основные требования соблюдены, критические ошибки устранены	слушатель может объяснить основные решения и исправить замечания после проверки
Повышенный	реализованы дополнительные сценарии, код структурирован, учтены граничные состояния, оформлена документация	слушатель аргументирует архитектуру, применяет тесты или автоматические проверки и показывает устойчивость решения

Модуль 4. Git, командная разработка и качество кода

Трудоемкость модуля составляет 56 академических часов. Модуль формирует рабочие навыки командной разработки. Слушатель изучает Git, удаленные репозитории, ветвление, pull request, code review, оформление коммитов, работу с задачами, правила качества кода и базовые процедуры сопровождения проекта.

Таблица 33 – Тематический план: Модуль 4. Git, командная разработка и качество кода

№	Тема	Час.	Лек. ч.	Прак. ч.	Сам. раб.	Конт- роль
1	Git: репозиторий, индекс, коммит, история, статус, diff, восстановление изменений	6	2	3	1	практикум
2	Удаленные репозитории, SSH/HTTPS-доступ, clone, push, pull, fetch	6	1	3	2	практикум
3	Ветвление, merge, rebase, конфликты, правила безопасной работы с историей	8	2	4	2	задача
4	Pull request, code review, issue, task board, правила описания изменений	8	2	4	2	PR
5	Единые правила качества: naming, структура проекта, линтеры, форматирование, документация	8	3	4	1	чек-лист
6	Промежуточная аттестация по модулю	4	0	0	0	зачет

Практическая подготовка.

- создать репозиторий учебного проекта и настроить структуру веток;
- оформить pull request с описанием задачи, скриншотами и чек-листом проверки;
- исправить конфликт слияния и объяснить порядок действий;
- применить единый стиль коммитов и подготовить README.

Планируемый практический результат.

- история коммитов понятна и отражает этапы разработки;

- изменения проходят через ветки и pull request;
- замечания code review фиксируются и исправляются;
- README содержит назначение проекта, стек, запуск и статус готовности.

Типовые ошибки, подлежащие исправлению.

- все изменения выполняются в основной ветке;
- коммиты имеют случайные названия;
- в репозитории находятся временные файлы, секреты или сборочные артефакты;
- pull request не содержит описания, скриншотов и инструкции проверки.

Условия зачета. Зачет проводится по репозиторию и pull request. Оцениваются история изменений, структура веток, решение конфликта, качество README, соблюдение правил коммитов и способность объяснить командный процесс. Таблица 35 – Уровни освоения по модулю: Модуль 4. Git, командная разработка и качество кода

Уровень освоения	Минимально достаточный результат	Повышенный результат
Минимальный	выполнены обязательные задания, проект запускается, основные требования соблюдены, критические ошибки устранены	слушатель может объяснить основные решения и исправить замечания после проверки
Повышенный	реализованы дополнительные сценарии, код структурирован, учтены граничные состояния, оформлена документация	слушатель аргументирует архитектуру, применяет тесты или автоматические проверки и показывает устойчивость решения

Модуль 5. Инструменты разработки, сборка и Linux

Трудоемкость модуля составляет 56 академических часов. Модуль посвящен инструментальной среде frontend-разработчика: Node.js, npm/pnpm, Vite, переменные окружения, конфигурация проекта, линтеры, форматирование, базовые

Linux-команды, статический хостинг, Nginx, Docker-окружение для проверки и элементы CI/CD.

Таблица 37 – Тематический план: Модуль 5. Инструменты разработки, сборка и Linux

№	Тема	Час.	Лек. ч.	Прак. ч.	Сам. раб.	Конт- роль
1	Node.js, npm, pnpm, package.json, scripts, dependencies, lock-файлы	6	2	3	1	практикум
2	Vite и сборка frontend-проекта: dev-сервер, build, preview, public assets	6	2	3	1	сборка
3	ESLint, Prettier, Stylelint, editorconfig, единые правила форматирования	6	1	3	2	настройка
4	Переменные окружения, конфигурация API, режимы разработки и сборки	6	1	3	2	конфигурация
5	Linux-команды, файловая система, права доступа, процессы, архивы, SSH	6	2	3	1	практикум
6	Публикация статического приложения: Nginx, Docker для проверки, GitHub Pages, базовый CI/CD	6	2	3	1	деплой
7	Промежуточная аттестация по модулю	4	0	0	0	зачет

Практическая подготовка.

- настроить Vite-проект с TypeScript, ESLint и Prettier;
- создать набор npm-скриптов для разработки, проверки, сборки и предпросмотра;
- вынести адрес API и режимы приложения в переменные окружения;
- собрать приложение и опубликовать его на выбранной учебной платформе или статическом хостинге.

Планируемый практический результат.

- проект устанавливается и запускается по README без ручных неописанных действий;
- команды build и preview проходят без критических ошибок;
- линтер и форматирование применяются единообразно;
- переменные окружения описаны через пример файла и не раскрывают секреты.

Типовые ошибки, подлежащие исправлению.

- зависимости устанавливаются случайно, lock-файл отсутствует;
- проект запускается только на компьютере автора;
- конфигурация API захардкожена в коде;
- сборка содержит предупреждения, которые мешают публикации.

Условия зачета. Зачет проводится по настроенному frontend-проекту. Проверятся установка, запуск, сборка, переменные окружения, линтеры, форматирование, базовый деплой и инструкция в README.

Таблица 39 – Уровни освоения по модулю: Модуль 5. Инструменты разработки, сборка и Linux

Уровень освоения	Минимально достаточный результат	Повышенный результат
Минимальный	выполнены обязательные задания, проект запускается, основные требования соблюдены, критические ошибки устранены	слушатель может объяснить основные решения и исправить замечания после проверки
Повышенный	реализованы дополнительные сценарии, код структурирован, учтены граничные состояния, оформлена документация	слушатель аргументирует архитектуру, применяет тесты или автоматические проверки и показывает устойчивость решения

Модуль 6. React и компонентная архитектура

Трудоемкость модуля составляет 148 академических часов. Модуль формирует ключевые навыки разработки на React: JSX, компоненты, props, state, события, формы, эффекты, пользовательские хуки, композиция, декомпозиция интерфейса, разделение ответственности, переиспользуемые компоненты и проектирование структуры приложения.

Продолжение на следующей странице

Продолжение таблицы 42

№	Тема	Час.	Лек. ч.	Прак. ч.	Сам. раб.	Конт- роль
---	------	------	------------	-------------	--------------	---------------

Таблица 41 – Тематический план: Модуль 6. React и компонентная архитектура

№	Тема	Час.	Лек. ч.	Прак. ч.	Сам. раб.	Конт- роль
1	React как библиотека интерфейсов: JSX, Virtual DOM, рендеринг, точка входа приложения	8	2	4	2	практикум
2	Компоненты, props, children, композиция, переиспользование и границы ответственности	10	3	5	2	компоненты
3	State, события, controlled components, формы и базовая валидация	10	3	5	2	форма
4	Списки, ключи, условный рендеринг, пустые состояния и состояния загрузки	8	2	4	2	задача
5	useEffect, жизненный цикл данных, подписки, очистка эффектов, ошибки зависимостей	10	3	5	2	практикум
6	Пользовательские хуки, выделение логики, работа с локальным хранилищем	10	3	5	2	hook

Продолжение на следующей странице

Продолжение таблицы 42

№	Тема	Час.	Лек. ч.	Прак. ч.	Сам. раб.	Конт- роль
7	Архитектура React-проекта: pages, widgets, features, entities, shared, импортные границы	12	3	6	3	структура
8	Стилизация компонентов: CSS Modules, Tailwind CSS, дизайн-токены, варианты компонентов	10	3	5	2	UI-блок
9	Формы повышенной сложности: динамические поля, ошибки, маски, отправка и сброс	10	3	5	2	форма
10	Обработка ошибок, fallback UI, error boundary, защищенные состояния интерфейса	8	2	4	2	задача
11	Самостоятельная доработка React-проекта и подготовка демонстрации	4	1	2	1	проверка
12	Промежуточная аттестация по модулю	4	0	0	0	зачет

Практическая подготовка.

- разработать React-приложение с несколькими страницами или режимами работы;
- выделить переиспользуемые компоненты, пользовательские хуки и общие UI-элементы;
- реализовать формы, списки, фильтрацию, состояния загрузки, пустого результата и ошибки;

- оформить архитектурную схему проекта и README с описанием структуры.

Планируемый практический результат.

- компоненты имеют ясную ответственность и повторно используются;
- состояние хранится на корректном уровне и не дублируется без причины;
- эффекты имеют корректные зависимости и очистку;
- структура проекта помогает сопровождать код и расширять функциональность.

Типовые ошибки, подлежащие исправлению.

- большие компоненты содержат разметку, бизнес-логику, запросы и стили одновременно;
- props передаются через большое количество уровней без архитектурного решения;
- useEffect используется для логики, которая может быть вычислена напрямую;
- отсутствуют состояния загрузки, ошибки и пустого результата.

Условия зачета. Зачет проводится по React-приложению. Проверяются компонентная структура, работа со state и props, формы, хуки, эффекты, стилизация, обработка состояний и объяснение архитектурных решений.

Продолжение на следующей странице

Продолжение таблицы 44

Уровень освоения	Минимально достаточный результат	Повышенный результат
------------------	----------------------------------	----------------------

Таблица 43 – Уровни освоения по модулю: Модуль 6. React и компонентная архитектура

Уровень освоения	Минимально достаточный результат	Повышенный результат
Минимальный	выполнены обязательные задания, проект запускается, основные требования соблюдены, критические ошибки устранены	слушатель может объяснить основные решения и исправить замечания после проверки
Повышенный	реализованы дополнительные сценарии, код структурирован, учтены граничные состояния, оформлена документация	слушатель аргументирует архитектуру, применяет тесты или автоматические проверки и показывает устойчивость решения

Модуль 7. Маршрутизация, управление состоянием и работа с API

Трудоемкость модуля составляет 104 академических часов. Модуль раскрывает разработку SPA-приложений с несколькими страницами, маршрутизацией, защищенными маршрутами, клиентским и серверным состоянием, интеграцией с REST API, пагинацией, фильтрами, авторизацией, кешированием и обработкой ошибок.

Таблица 45 – Тематический план: Модуль 7. Маршрутизация, управление состоянием и работа с API

№	Тема	Час.	Лек. ч.	Прак. ч.	Сам. раб.	Конт- роль
1	SPA-подход, маршруты, nested routes, layout, страницы и навигация	8	2	4	2	маршруты
2	React Router: параметры маршрута, query string, redirect, protected routes	8	2	4	2	задача
3	Клиентское состояние: Context, useReducer, локальное состояние, хранение настроек	8	2	4	2	state
4	Сторонние решения управления состоянием: Redux Toolkit или Zustand, selectors, actions	8	2	4	2	практикум
5	Работа с REST API: сервисный слой, fetch/axios, типизация запросов и ответов	10	3	5	2	API
6	Server state: кеширование, повторные запросы, optimistic update, TanStack Query	8	2	4	2	практикум
7	Авторизация в frontend-приложении: токены, хранение сессии, refresh-сценарии, logout	8	2	4	2	сценарий
8	Пагинация, фильтрация, сортировка, поиск и синхронизация состояния с URL	6	2	3	1	задача

Продолжение на следующей странице

Продолжение таблицы 46

№	Тема	Час.	Лек. ч.	Прак. ч.	Сам. раб.	Конт- роль
9	Самостоятельная доработка интеграции и документации API	4	1	3	0	проверка
10	Промежуточная аттестация по модулю	4	0	0	0	зачет

Практическая подготовка.

- разработать SPA-приложение с несколькими маршрутами и общим layout;
- подключить API и реализовать обработку загрузки, ошибок, пустых состояний и повторных запросов;
- реализовать фильтрацию, поиск или пагинацию с сохранением параметров в URL;
- добавить сценарий авторизации или имитацию защищенного маршрута.

Планируемый практический результат.

- пользователь может перемещаться между страницами без потери контекста;
- данные API загружаются через отдельный сервисный слой;
- серверное и клиентское состояние разделены;
- ошибки API обрабатываются с понятным пользовательским сообщением.

Типовые ошибки, подлежащие исправлению.

- запросы к API размещены хаотично внутри разных компонентов;
- URL не отражает состояние фильтрации и поиска;
- загрузка данных повторяется лишний раз при каждом рендеринге;
- токены и чувствительные значения хранятся без анализа рисков.

Условия зачета. Зачет проводится по SPA-приложению с маршрутами, состоянием и API. Проверяются сценарии пользователя, устойчивость к ошибкам,

типизация данных, структура сервисного слоя и качество пользовательской обратной связи.

Таблица 47 – Уровни освоения по модулю: Модуль 7. Маршрутизация, управление состоянием и работа с API

Уровень освоения	Минимально достаточный результат	Повышенный результат
Минимальный	выполнены обязательные задания, проект запускается, основные требования соблюдены, критические ошибки устранены	слушатель может объяснить основные решения и исправить замечания после проверки
Повышенный	реализованы дополнительные сценарии, код структурирован, учтены граничные состояния, оформлена документация	слушатель аргументирует архитектуру, применяет тесты или автоматические проверки и показывает устойчивость решения

Модуль 8. Тестирование, доступность, производительность и безопасность frontend-приложений

Трудоемкость модуля составляет 68 академических часов. Модуль формирует навыки проверки качества frontend-приложения. Слушатель изучает модульные и компонентные тесты, end-to-end сценарии, отладку, DevTools, доступность, производительность, базовую безопасность, защиту от XSS-рисков и подготовку проекта к приемке.

Таблица 49 – Тематический план: Модуль 8. Тестирование, доступность, производительность и безопасность frontend-приложений

№	Тема	Час.	Лек. ч.	Прак. ч.	Сам. раб.	Конт- роль
1	Пирамида тестирования frontend-приложений, критерии качества и приемочные сценарии	4	1	2	1	опрос
2	Vitest: модульные тесты функций, мокирование, покрытие, граничные случаи	6	2	3	1	тесты
3	React Testing Library: компонентные тесты, user events, формы, состояния	8	2	4	2	тесты
4	Playwright: end-to-end сценарии, маршруты, фикстуры, проверка пользовательского пути	8	2	4	2	e2e
5	Доступность: WCAG-ориентиры, клавиатура, фокус, aria-атрибуты, семантика	6	2	3	1	чек-лист
6	Производительность: Lighthouse, lazy loading, code splitting, изображения, bundle-анализ	6	1	3	2	оптимизация
7	Безопасность frontend: XSS, хранение токенов, CORS, секреты, зависимости, пользовательский ввод	6	2	3	1	аудит
8	Промежуточная аттестация по модулю	4	0	0	0	зачет

Практическая подготовка.

- написать тесты для функций преобразования данных и для ключевых React-компонентов;
- создать end-to-end сценарий для основного пользовательского пути;
- провести аудит доступности и производительности с фиксацией найденных проблем;
- исправить минимум три дефекта качества и описать изменения в README.

Планируемый практический результат.

- основные пользовательские сценарии покрыты тестами;
- приложение проверено по доступности клавиатурной навигации и контрастности;
- сборка оптимизирована с учетом размера ресурсов и скорости загрузки;
- пользовательский ввод обрабатывается безопасно, секреты не попадают в репозиторий.

Типовые ошибки, подлежащие исправлению.

- тесты проверяют только наличие элементов без пользовательского сценария;
- недоступны фокус и управление с клавиатуры;
- изображения и ресурсы загружаются без оптимизации;
- в проекте раскрыты токены, ключи или внутренние адреса.

Условия зачета. Зачет проводится по приложению с тестами и отчетом качества. Проверяются покрытые сценарии, корректность тестов, доступность, производительность, безопасность и способность слушателя объяснить найденные и исправленные дефекты.

Таблица 51 – Уровни освоения по модулю: Модуль 8. Тестирование, доступность, производительность и безопасность frontend-приложений

Уровень освоения	Минимально достаточный результат	Повышенный результат
Минимальный	выполнены обязательные задания, проект запускается, основные требования соблюдены, критические ошибки устранены	слушатель может объяснить основные решения и исправить замечания после проверки
Повышенный	реализованы дополнительные сценарии, код структурирован, учтены граничные состояния, оформлена документация	слушатель аргументирует архитектуру, применяет тесты или автоматические проверки и показывает устойчивость решения

Модуль 9. Дизайн-системы, UI-kit и командная frontend-практика

Трудоемкость модуля составляет 36 академических часов. Модуль связывает разработку интерфейсов с проектированием пользовательского опыта и командной работой. Слушатель изучает работу с макетами, дизайн-токенами, UI-kit, документацией компонентов, Storybook, правилами совместимости и передачей интерфейсных решений между дизайном, аналитикой, backend и frontend.

Продолжение на следующей странице

Продолжение таблицы 54

№	Тема	Час.	Лек. ч.	Прак. ч.	Сам. раб.	Конт- роль
---	------	------	------------	-------------	--------------	---------------

Таблица 53 – Тематический план: Модуль 9. Дизайн-системы, UI-kit и командная frontend-практика

№	Тема	Час.	Лек. ч.	Прак. ч.	Сам. раб.	Конт- роль
1	Чтение макета, состояния компонентов, пользовательские сценарии и ограничения интерфейса	4	1	2	1	разбор
2	Дизайн-токены: цвета, типографика, отступы, радиусы, тени, темы	4	1	2	1	практикум
3	UI-kit: кнопки, поля, модальные окна, таблицы, уведомления, карточки	6	2	3	1	компоненты
4	Storybook или аналогичная документация компонентов, состояния и примеры использования	4	1	2	1	документация
5	Командные практики: декомпозиция задач, оценка, приемка, взаимодействие с backend и дизайном	4	1	3	0	сценарий
6	Промежуточная аттестация по модулю	2	0	0	0	зачет

Практическая подготовка.

- создать мини UI-kit из базовых компонентов;
- описать дизайн-токены и состояния компонентов;
- подготовить документацию по использованию компонентов;
- оформить декомпозицию задачи и критерии приемки интерфейсного блока.

Планируемый практический результат.

- компоненты имеют единый визуальный стиль и набор состояний;
- токены позволяют менять визуальные параметры без массовой правки компонентов;
- документация объясняет назначение и ограничения UI-элементов;
- критерии приемки понятны для разработчика, наставника и проверяющего.

Типовые ошибки, подлежащие исправлению.

- компоненты дублируются вместо переиспользования;
- визуальные параметры задаются случайно без токенов;
- документация компонента отсутствует;
- состояния ошибки, загрузки и отключения не предусмотрены.

Условия зачета. Зачет проводится по мини UI-kit и документации. Проверяются единый стиль, полнота состояний, переиспользуемость компонентов, качество описания и связь с пользовательскими сценариями.

Таблица 55 – Уровни освоения по модулю: Модуль 9. Дизайн-системы, UI-kit и командная frontend-практика

Уровень освоения	Минимально достаточный результат	Повышенный результат
Минимальный	выполнены обязательные задания, проект запускается, основные требования соблюдены, критические ошибки устранены	слушатель может объяснить основные решения и исправить замечания после проверки

Продолжение на следующей странице

Продолжение таблицы 56

Уровень освоения	Минимально достаточный результат	Повышенный результат
Повышенный	реализованы дополнительные сценарии, код структурирован, учтены граничные состояния, оформлена документация	слушатель аргументирует архитектуру, применяет тесты или автоматические проверки и показывает устойчивость решения

Модуль 10. Проектная практика и консультации

Трудоемкость модуля составляет 56 академических часов. Модуль обеспечивает сборку результатов обучения в итоговый проект. Слушатель выбирает тему, уточняет требования, декомпозирует функциональность, проектирует архитектуру, реализует ключевые сценарии, подключает API или mock-данные, пишет тесты, оформляет README, готовит демонстрацию и защиту.

Таблица 57 – Тематический план: Модуль 10. Проектная практика и консультации

№	Тема	Час.	Лек. ч.	Прак. ч.	Сам. раб.	Контроль
1	Выбор темы, постановка цели проекта, описание пользователей и сценариев	6	0	4	2	консультация
2	Техническое задание, декомпозиция страниц, компонентов, данных и состояний	6	0	4	2	проверка
3	Реализация основной функциональности и интеграция с API или mock-сервисом	10	0	8	2	ревью

Продолжение на следующей странице

Продолжение таблицы 58

№	Тема	Час.	Лек. ч.	Прак. ч.	Сам. раб.	Конт- роль
4	Тестирование, доступность, производительность, исправление дефектов	8	0	6	2	ревью
5	Документация, сборка, публикация, демонстрационные материалы	6	0	5	1	проверка
6	Предзащита и доработка по замечаниям	4	0	3	1	предзащита

Практическая подготовка.

- подготовить паспорт итогового проекта;
- создать структуру репозитория и дорожную карту задач;
- реализовать ключевой пользовательский сценарий;
- провести предзащиту и доработать проект по замечаниям.

Планируемый практический результат.

- итоговый проект имеет понятную цель, функциональный состав и критерии приемки;
- репозиторий отражает ход разработки и содержит документацию;
- приложение запускается и демонстрирует основные сценарии;
- слушатель готов к итоговой защите и техническому собеседованию.

Типовые ошибки, подлежащие исправлению.

- выбрана тема без проверяемого пользовательского сценария;
- проект не имеет README и инструкции запуска;
- в проекте отсутствуют заявленные в описании маршруты, состояния или тесты;
- предзащита пройдена формально без устранения замечаний.

Условия зачета. Зачет по проектной практике выставляется при наличии работоспособного прототипа, паспорта проекта, репозитория, README, плана доработок, тестового или проверочного сценария и материалов предзащиты.

Таблица 59 – Уровни освоения по модулю: Модуль 10. Проектная практика и консультации

Уровень освоения	Минимально достаточный результат	Повышенный результат
Минимальный	выполнены обязательные задания, проект запускается, основные требования соблюдены, критические ошибки устранены	слушатель может объяснить основные решения и исправить замечания после проверки
Повышенный	реализованы дополнительные сценарии, код структурирован, учтены граничные состояния, оформлена документация	слушатель аргументирует архитектуру, применяет тесты или автоматические проверки и показывает устойчивость решения

2.4. Проектная практика и консультации

Проектная практика является сквозным элементом программы. Начиная с первых модулей слушатель постепенно формирует портфолио frontend-разработчика: адаптивная верстка, интерактивные JavaScript-приложения, типизированные модули, React-компоненты, SPA-приложение, интеграция с API, тесты, UI-kit, итоговый проект. Консультации используются для уточнения требований, разбора ошибок, проверки архитектуры, подготовки к защите и формирования профессиональной обратной связи.

Таблица 61 – Этапы проектной траектории

№	Этап	Практический результат	Фиксация результата
1	Стартовая настройка	рабочее окружение, учебный репозиторий, входная диагностика	скриншоты, ссылка на репозиторий, чек-лист
2	Верстка и интерфейсные основы	адаптивная страница, форма, карточки, сетка, доступные состояния	pull request, README, демонстрация страницы
3	JavaScript и TypeScript	интерактивное мини-приложение с API или локальными данными	код, типы, инструкция запуска, краткий отчет
4	React-приложение	компоненты, формы, хуки, эффекты, структура проекта	репозиторий, демонстрация, разбор архитектуры
5	SPA и API	маршруты, серверные данные, состояние, ошибки и авторизация	прототип, тестовые данные, сценарии проверки
6	Качество и приемка	тесты, аудит доступности, производительности и безопасности	отчет, исправления, чек-лист
7	Итоговый проект	полноценное frontend-приложение, README, защита, ответы на вопросы	комиссионная оценка

Итоговая аттестация. Итоговая аттестация проводится в форме защиты итогового frontend-проекта. Проект должен демонстрировать владение основными результатами программы: адаптивной версткой, JavaScript/TypeScript, React, маршрутизацией, управлением состоянием, API-интеграцией, тестированием, доступностью, качеством репозитория и способностью слушателя объяснить принятые технические решения.

Таблица 63 – Состав итоговой аттестации

Элемент итоговой аттестации	Содержание	Оценивание
Итоговый проект	frontend-приложение по выбранной теме с ключевыми пользовательскими сценариями	до 50 баллов
Код и архитектура	компоненты, типизация, состояние, маршруты, API, структура файлов, качество решений	до 20 баллов
Тестирование и качество	тесты, доступность, производительность, безопасность, исправление дефектов	до 15 баллов
Документация	README, инструкция запуска, описание стека, сценариев, ограничений и дальнейшего развития	до 10 баллов
Защита	демонстрация проекта, ответы на вопросы, объяснение архитектуры и компромиссов	до 5 баллов

3. Организационно-педагогические условия реализации программы

3.1. Общие условия реализации

Реализация программы обеспечивается учебно-методическими материалами, электронными образовательными ресурсами, практическими заданиями, консультациями, системой проверки результатов, репозиториями проектов и итоговой аттестацией. Организация обеспечивает доступ слушателей к материалам программы, фиксацию результатов обучения и возможность получения обратной связи.

Образовательный процесс строится на сочетании самостоятельного изучения материалов, выполнения практических заданий, проектной разработки, консультаций с наставником и контрольных мероприятий. Такая модель позволяет проверять не только усвоение отдельных тем, но и способность применять их в законченном frontend-проекте.

3.2. Материально-технические условия

Таблица 65 – Материально-техническое обеспечение

Элемент обеспечения	Требования
Компьютер слушателя	современный ноутбук или персональный компьютер с возможностью запуска редактора кода, браузеров, Node.js, Git и инструментов сборки
Интернет-доступ	стабильное подключение для работы с учебной платформой, репозиториями, консультациями, API и облачными сервисами
Браузеры	Google Chrome, Chromium, Mozilla Firefox или аналоги с доступом к DevTools
Редактор кода	Visual Studio Code, WebStorm или другой редактор с поддержкой TypeScript, ESLint, Prettier и Git

Продолжение на следующей странице

Продолжение таблицы 66

Элемент обеспечения	Требования
Средства коммуникации	доступ к системе видеосвязи, чату, электронной почте или иной системе уведомлений организации
Репозиторий	индивидуальный или учебный Git-репозиторий для фиксации кода, проверки и защиты проекта

Рекомендуемое программное обеспечение. Для выполнения заданий используется актуальная стабильная версия инструментов разработки. Конкретные версии могут уточняться преподавателем с учетом совместимости учебных материалов.

Таблица 67 – Рекомендуемый программный стек

Группа	Инструменты	Назначение
Языки и стандарты	HTML5, CSS3, JavaScript ES2020+, TypeScript	разметка, стилизация, клиентская логика и типизация
Фреймворк и библиотеки	React, React Router, Redux Toolkit или Zustand, TanStack Query при необходимости	разработка SPA, маршрутизация, состояние и серверные данные
Сборка	Node.js LTS, npm или pnpm, Vite	установка зависимостей, dev-сервер, сборка и предпросмотр
Качество	ESLint, Prettier, Stylelint, TypeScript strict mode	статический анализ, форматирование, единые правила кода
Тестирование	Vitest, React Testing Library, Playwright	модульные, компонентные и end-to-end проверки

Продолжение на следующей странице

Продолжение таблицы 68

Группа	Инструменты	Назначение
Документация UI	Storybook или аналогичный инструмент	демонстрация и описание компонентов
Деплой	GitHub Pages, Vercel, Netlify, Nginx, Docker при необходимости	публикация и проверка собранного приложения
Контроль версий	Git, GitHub или GitLab	история изменений, ветки, pull request, code review
Дизайн и макеты	Figma в режиме просмотра, экспорт ресурсов, дизайн-токены	чтение макета и подготовка интерфейса к разработке

3.3. Электронная информационно-образовательная среда

Электронная информационно-образовательная среда обеспечивает размещение учебных материалов, доступ к заданиям, фиксацию результатов, обратную связь, консультации, контрольные мероприятия и хранение цифрового следа обучения. В среде могут размещаться видеозаписи, конспекты, тесты, практические задания, критерии оценивания, ссылки на репозитории, результаты проверки и итоговые документы.

- идентификация слушателя осуществляется через учетную запись, выданную или подтвержденную организацией;
- результаты выполнения заданий фиксируются через учебную платформу, репозиторий, файлы, ссылки, протоколы проверки или комментарии наставника;
- доступ к материалам и результатам разграничивается по ролям слушателя, преподавателя, наставника, администратора и комиссии;
- при проведении контрольных мероприятий могут использоваться видеосвязь, запись демонстрации экрана, проверка истории репозитория, устное собеседование и дополнительные вопросы;

- порядок идентификации, аутентификации, прокторинга при необходимости и хранения результатов определяется локальными актами организации.

3.4. Кадровые условия

К реализации программы привлекаются преподаватели, наставники и проверяющие, обладающие профессиональной компетентностью в сфере frontend-разработки, веб-технологий, JavaScript, TypeScript, React, тестирования и проектной работы. Кадровые условия должны обеспечивать передачу теоретических знаний, проверку практических решений, ревью кода, методическую поддержку и объективную оценку итоговых проектов.

Таблица 69 – Кадровое обеспечение программы

Роль	Функции	Квалификационные требования
Преподаватель	объяснение тем, подготовка учебных материалов, проведение консультаций и промежуточного контроля	практический опыт frontend-разработки или преподавания веб-разработки
Наставник	ревью заданий, разбор ошибок, поддержка проектной траектории, подготовка к защите	владение JavaScript, TypeScript, React, Git и инструментами проверки качества
Проверяющий	оценивание зачетных заданий, итогового проекта и материалов защиты	понимание критериев программы и опыт проверки кода
Администратор программы	организация доступа, расписания, документооборота, фиксации результатов и коммуникации	знание локальных процедур реализации ДПО

3.5. Учебно-методическое обеспечение

Учебно-методическое обеспечение включает рабочую программу, учебный план, календарный график, видеоматериалы, конспекты, практические задания, тесты, шаблоны проектов, чек-листы, требования к репозиторию, критерии оценивания, задания для промежуточной аттестации, паспорт итогового проекта и форму оценочного листа защиты.

- каждый модуль сопровождается описанием целей, тем, практических результатов и условий зачета;
- задания формулируются через проверяемый результат, критерии приемки и формат сдачи;
- слушатель получает обратную связь по ошибкам, качеству кода, архитектуре, документации и демонстрации;
- итоговый проект оценивается по единой 100-балльной шкале и защищается перед преподавателем или комиссией.

3.6. Применение электронного обучения и дистанционных образовательных технологий

Программа реализуется с применением электронного обучения и дистанционных образовательных технологий. Слушатель получает доступ к материалам, выполняет практические работы, направляет результаты на проверку, участвует в консультациях и проходит аттестацию с использованием цифровых образовательных сервисов.

Продолжение на следующей странице

Продолжение таблицы 72

Механизм	Описание
----------	----------

Таблица 71 – Механизмы применения ЭО и ДОТ

Механизм	Описание
Доступ к материалам	предоставление видеолекций, конспектов, заданий, шаблонов, чек-листов и критериев оценивания
Практические задания	выполнение кода в локальной среде, отправка ссылок на репозиторий, загрузка файлов и демонстрация результата
Консультации	онлайн-разборы, ответы на вопросы, ревью кода, обсуждение проекта и подготовка к защите
Контроль	проверка тестов, кода, репозитория, README, демонстрации, устных ответов и индивидуальных пояснений
Фиксация результатов	сохранение отметок, комментариев, сроков сдачи, протоколов аттестации и материалов итоговой защиты

3.7. Информационная безопасность, персональные данные и этика

При реализации программы соблюдаются требования к обработке персональных данных, разграничению доступа, защите учетных записей, корректному использованию учебных материалов, самостоятельности выполнения работ и недопущению передачи третьим лицам учетных данных слушателя. В проектах запрещается размещать реальные персональные данные, секретные ключи, токены, пароли и закрытую информацию.

- в учебных проектах используются тестовые, обезличенные или самостоятельно подготовленные данные;
- переменные окружения и ключи доступа описываются через пример конфигурационного файла;
- репозитории проверяются на отсутствие секретов, персональных данных и запрещенного контента;

- использование внешних библиотек должно сопровождаться пониманием их назначения и соблюдением лицензионных условий;
- заимствованный код, дизайн или материалы должны быть допустимы к использованию в учебных целях и указаны в README.

3.8. Локальные процедуры обучения

Порядок реализации программы уточняется локальными актами организации. К таким процедурам относятся прием и зачисление, ведение личных дел слушателей, предоставление доступа к учебной среде, учет посещаемости и активности, проверка заданий, пересдача, апелляция, итоговая аттестация, выдача документов о квалификации и внесение сведений в ФИС ФРДО.

Таблица 73 – Локальные процедуры, связанные с реализацией программы

Процедура	Содержание
Прием и зачисление	проверка документов, заключение договора, оформление приказа, выдача доступа
Текущий контроль	проверка практических заданий, тестов, репозиторий, сроков сдачи и исправления замечаний
Промежуточная аттестация	зачеты по модулям через практические работы, собеседование или комбинированную проверку
Пересдача	повторная сдача задания или доработка по замечаниям в установленные сроки
Апелляция	рассмотрение несогласия с оценкой по материалам проверки и критериям оценивания
Итоговая аттестация	допуск, защита проекта, оценочный лист, протокол и итоговое решение
Выдача документа	оформление диплома о профессиональной переподготовке и передача сведений в ФИС ФРДО

3.9. Обновление программы

Программа подлежит регулярному обновлению с учетом изменений законодательства, профессиональных стандартов, требований работодателей, развития frontend-технологий, обновления React-экосистемы, практики реализации образовательной программы и результатов внутренней оценки качества. Обновление может касаться тем модулей, практических заданий, версий инструментов, оценочных материалов, проектных тем и методических рекомендаций.

4. Оценка качества освоения программы

4.1. Общие положения

Оценка качества освоения программы проводится через текущий контроль, промежуточную аттестацию, проектную практику и итоговую аттестацию. Оценивание направлено на проверку способности слушателя применять знания в практической разработке frontend-приложений, поддерживать качество кода, объяснять технические решения и оформлять результат в виде проверяемого проекта.

Таблица 75 – Формы оценки качества освоения программы

Форма контроля	Что проверяется	Материалы проверки
Текущий контроль	выполнение практических заданий, понимание темы, исправление ошибок	код, ответы, скриншоты, ссылка на репозиторий, чек-лист
Промежуточная аттестация	освоение модуля и готовность применять результат в проекте	зачетное задание, README, демонстрация, устные вопросы
Проектная практика	интеграция модулей в единый проект и способность дорабатывать решение	паспорт проекта, репозиторий, предзащита, список замечаний
Итоговая аттестация	готовность к новой профессиональной деятельности	итоговый проект, защита, оценочный лист, ответы на вопросы

4.2. Текущий контроль успеваемости

Текущий контроль проводится в ходе каждого модуля. Он включает проверку практических работ, домашних заданий, тестовых вопросов, мини-проектов, репозиторийев, демонстраций и участия в консультациях. Текущий контроль помогает выявлять пробелы до промежуточной аттестации и формировать индивидуальную траекторию доработки.

- практические задания проверяются по критериям функциональности, качества кода, структуры, запуска и документации;
- ошибки фиксируются в комментариях, чек-листе или системе проверки;
- слушатель дорабатывает работу и повторно направляет результат на проверку;
- результаты текущего контроля учитываются при допуске к промежуточной и итоговой аттестации.

4.3. Промежуточная аттестация

Промежуточная аттестация проводится по каждому содержательному модулю в форме зачета. Зачет может включать практическое задание, проверку репозитория, демонстрацию результата, устные вопросы, исправление ошибок и анализ кода. Минимальное условие зачета - выполнение обязательного практического результата без критических ошибок.

Таблица 77 – Промежуточная аттестация по модулям

Модуль	Зачетное задание	Критерии проверки
HTML, CSS и адаптивная верстка	адаптивная страница по макету	семантика, адаптивность, CSS-структура, доступность, визуальные состояния
JavaScript и TypeScript	мини-приложение с интерактивной логикой и API	модульность, типизация, асинхронность, обработка ошибок, читаемость
Git и командная разработка	репозиторий с ветками и pull request	история коммитов, PR, README, исправление замечаний, отсутствие секретов
Инструменты и сборка	настроенный Vite-проект	установка, scripts, lint, build, env, базовый деплой

Продолжение на следующей странице

Продолжение таблицы 78

Модуль	Зачетное задание	Критерии проверки
React	React-приложение с компонентной архитектурой	components, props, state, hooks, forms, effects, структура
Маршрутизация, состояние и API	SPA с маршрутами и серверными данными	routing, state, service layer, loading/error/empty, авторизация
Тестирование и качество	набор тестов и отчет качества	Vitest, RTL, Playwright, доступность, производительность, безопасность
UI-kit и командная практика	мини UI-kit и документация компонентов	токены, состояния, переиспользование, Storybook или описание

4.4. Пересдача, апелляция и подтверждение самостоятельности

Пересдача промежуточной аттестации проводится в форме доработки задания, повторной демонстрации, дополнительного практического задания или устного собеседования. Формат пересдачи определяется преподавателем или наставником с учетом характера выявленных ошибок. Апелляция рассматривается на основании критериев оценивания, сохраненных материалов, истории репозитория и пояснений слушателя.

Самостоятельность выполнения работ подтверждается историей репозитория, устной защитой, способностью объяснить код, ответить на вопросы, внести изменения в проект и обосновать архитектурные решения. При выявлении признаков несамостоятельного выполнения организация вправе назначить дополнительную проверку.

4.5. Допуск к итоговой аттестации

К итоговой аттестации допускается слушатель, освоивший учебный план, выполнивший обязательные практические задания, получивший зачеты по моду-

лям, подготовивший итоговый проект, оформивший репозиторий и README, прошедший предзащиту или консультационную проверку проекта.

Таблица 79 – Условия допуска к итоговой аттестации

№	Условие допуска	Форма подтверждения
1	выполнены обязательные практические задания по модулям	отметки в ЭИОС
2	получены зачеты по промежуточной аттестации	ведомость
3	итоговый проект размещен в репозитории	ссылка
4	README содержит инструкцию запуска, стек, сценарии и ограничения	файл README
5	приложение запускается локально или опубликовано для демонстрации	демонстрация
6	подготовлены материалы защиты	презентация или план выступления

4.6. Итоговая аттестация

Итоговая аттестация проводится в форме защиты итогового frontend-проекта. Защита включает демонстрацию приложения, описание задачи, пользователей, архитектуры, компонентов, состояния, маршрутов, API, тестов, доступности, производительности, безопасности и дальнейших направлений развития проекта. Комиссия или проверяющий вправе задать вопросы по любому модулю программы и предложить внести небольшое изменение в код для проверки самостоятельности.

Таблица 81 – Критерии итоговой оценки проекта

Критерий	Баллы	Пояснение
Функциональность и пользовательские сценарии	20	реализованы заявленные сценарии, интерфейс работает устойчиво, предусмотрены граничные состояния
HTML, CSS и адаптивность	10	семантика, адаптивная верстка, доступность, визуальные состояния, соответствие макету
JavaScript, TypeScript и React	20	типизация, компоненты, хуки, state, формы, эффекты, структура приложения
Маршрутизация, состояние и API	15	route-структура, сервисный слой, server state, загрузка, ошибки, фильтры, авторизация при наличии
Качество кода и архитектура	10	декомпозиция, читаемость, отсутствие дублирования, линтеры, форматирование, единые правила
Тестирование, доступность, производительность, безопасность	10	тесты, чек-листы, Lighthouse, исправленные дефекты, безопасная работа с данными
Репозиторий и документация	10	README, инструкция запуска, история коммитов, описание стека, ограничения и сценарии проверки
Защита и ответы на вопросы	5	ясное объяснение решений, демонстрация самостоятельности, аргументация доработок

Шкала итоговой оценки.

Таблица 83 – Шкала итоговой оценки

Баллы	Результат	Характеристика результата
90–100	отлично	проект полностью соответствует требованиям, имеет качественную архитектуру, тесты, документацию и уверенную защиту
80–89	хорошо	проект соответствует основным требованиям, содержит отдельные замечания, которые не мешают демонстрации результата
70–79	удовлетворительно	проект выполняет ключевые сценарии, при этом содержит заметные недочеты в архитектуре, качестве или документации
менее 70	неудовлетворительно	проект не подтверждает освоение программы или содержит критические ошибки

Критические основания для неудовлетворительного результата. Критические основания фиксируются при наличии ошибок, которые не позволяют признать итоговый проект подтверждением освоения программы. При наличии одного или нескольких критических оснований итоговый результат может быть признан неудовлетворительным независимо от частичных баллов.

- проект не запускается по инструкции или отсутствует инструкция запуска;
- отсутствует репозиторий с историей разработки;
- основной пользовательский сценарий не реализован;
- код не соответствует заявленному стеку программы;
- слушатель не может объяснить ключевые фрагменты кода и архитектуры;
- в проекте размещены секреты, персональные данные или материалы, запрещенные к использованию;
- имеются признаки несамостоятельного выполнения без подтверждения авторства результата.

4.7. Внутренняя и внешняя оценка качества реализации программы

Внутренняя оценка качества проводится организацией через анализ результатов текущего контроля, промежуточной аттестации, итоговых проектов, обратной связи слушателей, отзывов преподавателей, динамики доработок и качества методических материалов. Внешняя независимая оценка качества может проводиться через экспертную оценку программы, проверку итоговых проектов внешними специалистами, анализ соответствия профессиональным стандартам и сопоставление результатов выпускников с требованиями рынка труда.

Таблица 85 – Показатели оценки качества программы

Показатель	Источник данных	Применение результата
Доля успешно завершивших обучение	ведомости, протоколы, ЭИОС	оценка достижимости программы
Качество итоговых проектов	оценочные листы, репозитории, защиты	обновление заданий и критериев
Типовые ошибки слушателей	комментарии наставников, чек-листы	доработка материалов и консультаций
Актуальность стека	анализ технологий, обратная связь специалистов	обновление модулей и практик
Удовлетворенность слушателей	анкеты и обращения	улучшение организации обучения
Внешняя экспертная оценка	заключения, ревью проектов, консультации работодателей	подтверждение качества программы

5. Методические материалы

5.1. Методические указания для слушателей

Слушателю рекомендуется изучать материалы программы последовательно, выполнять практические задания сразу после соответствующих тем, вести репозиторий с первого модуля, фиксировать вопросы, регулярно запускать проверки, оформлять README и демонстрационные материалы. Особое внимание следует уделять объяснимости решений: разработчик должен понимать, почему выбрана структура компонентов, где хранится состояние, как обрабатываются ошибки и каким образом пользователь проходит основной сценарий.

- после каждой темы фиксировать краткий конспект и список новых команд или приемов;
- каждое практическое задание выполнять в отдельной ветке или отдельном учебном репозитории по указанию преподавателя;
- проверять проект перед сдачей через запуск, сборку, линтер, тесты и пользовательский сценарий;
- оформлять README сразу, дополняя его по мере развития проекта;
- перед защитой подготовить короткий сценарий демонстрации, список решений и список известных ограничений.

5.2. Методические указания для преподавателей и проверяющих

Преподавателю и проверяющему рекомендуется оценивать конечный вид интерфейса, процесс разработки, структуру проекта, историю изменений, качество декомпозиции, обработку ошибок, тестируемость, доступность, производительность и способность слушателя объяснить решения. Проверка должна быть единообразной и опираться на критерии программы.

- перед проверкой задания уточнить, какой результат должен быть достигнут в конкретном модуле;
- фиксировать замечания по категориям: функциональность, верстка, логика, типизация, архитектура, тесты, документация, безопасность;
- отделять критические ошибки от рекомендаций по улучшению;

- давать слушателю понятный путь доработки: что исправить в первую очередь, какие файлы проверить, как подтвердить исправление;
- на итоговой защите задавать вопросы по проекту, по стеку программы и по изменению небольшого фрагмента функциональности.

5.3. Рекомендуемая литература и ресурсы

Рекомендуемые ресурсы используются как учебно-методическая поддержка. Конкретный перечень материалов может уточняться преподавателем с учетом обновления веб-стандартов и инструментов.

Таблица 87 – Рекомендуемые учебные и справочные ресурсы

№	Ресурс	Назначение
1	MDN Web Docs	справочник по HTML, CSS, JavaScript, Web API и доступности
2	React Documentation	официальное руководство по React, компонентам, хукам и архитектурным подходам
3	TypeScript Handbook	официальное руководство по типам, интерфейсам, generics и конфигурации
4	Vite Documentation	настройка сборки, dev-сервера и production build
5	React Router Documentation	маршрутизация SPA-приложений
6	Redux Toolkit Documentation	управление состоянием в приложениях при необходимости
7	Testing Library Documentation	подходы к компонентному тестированию через пользовательское поведение

Продолжение на следующей странице

Продолжение таблицы 88

№	Ресурс	Назначение
8	Playwright Documentation	end-to-end тестирование пользовательских сценариев
9	Web.dev	производительность, доступность, качество и практики веб-разработки
10	WCAG Overview	ориентиры доступности интерфейсов и пользовательских сценариев

Приложение 1. Матрица формирования компетенций

Приложение 2. Матрица компетенций и модулей.

Таблица 89 – Матрица компетенций и модулей

Код	Компетенция	Модули	Форма подтверждения
ПК-1	Проектирование структуры пользовательского интерфейса	1, 2, 9, 10, 11	паспорт проекта, декомпозиция страниц, карта компонентов
ПК-2	Разработка семантической и адаптивной верстки	2, 6, 8, 10, 11	адаптивная страница, UI-секции, чек-лист доступности
ПК-3	Разработка клиентской логики на JavaScript	3, 7, 10, 11	мини-приложение, обработка событий, API, ошибки
ПК-4	Применение TypeScript	3, 6, 7, 8, 10, 11	типизированные модели, props, ответы API, проверки build
ПК-5	Командная разработка и Git-процесс	4, 5, 10, 11	репозиторий, ветки, PR, README, история коммитов
ПК-6	Настройка окружения и сборки	5, 8, 10, 11	Vite, npm scripts, lint, build, env, деплой
ПК-7	Разработка приложений на React	6, 7, 9, 10, 11	компоненты, хуки, формы, страницы, UI-kit
ПК-8	Маршрутизация, состояние и API	7, 8, 10, 11	SPA, protected routes, state, service layer, server state
ПК-9	Качество, доступность и производительность	2, 5, 8, 9, 10, 11	тесты, WCAG-чек-лист, Lighthouse, исправления

Продолжение на следующей странице

Продолжение таблицы 90

Код	Компетенция	Модули	Форма подтверждения
ПК-10	Выполнение и защита проекта	10, 11	итоговый проект, защита, ответы на вопросы, оценочный лист

Приложение 2. Оценочные материалы промежуточной аттестации

Приложение 4. Примерные задания промежуточной аттестации.

Таблица 91 – Примерные задания промежуточной аттестации

Модуль	Пример задания
HTML, CSS и адаптивная верстка	сверстать адаптивную страницу по макету, реализовать форму, состояния интерактивных элементов, мобильную версию и чек-лист доступности
JavaScript и TypeScript	создать мини-приложение с фильтрацией и сортировкой данных, подключением API, обработкой загрузки, ошибки и пустого состояния, типизировать данные
Git и командная разработка	оформить репозиторий, создать ветку, выполнить pull request, решить конфликт, написать README и пройти code review
Инструменты и сборка	настроить Vite-проект, ESLint, Prettier, env-файлы, build, preview и опубликовать статическую сборку
React	разработать React-приложение с компонентами, формами, хуками, списками, состояниями и документацией структуры
Маршрутизация и API	реализовать SPA с несколькими маршрутами, параметрами URL, сервисным слоем, API-запросами, фильтрами и обработкой ошибок
Тестирование и качество	написать модульные, компонентные и e2e-тесты, провести аудит доступности и производительности, исправить найденные дефекты
UI-kit	создать набор компонентов с состояниями, дизайн-токенами, документацией и демонстрационными примерами

Приложение 3. Оценочные материалы итоговой аттестации

Приложение 3. Тематика проектных и выпускных работ.

Таблица 93 – Тематика итоговых frontend-проектов

№	Тема проекта	Содержание проекта
1	Адаптивный сайт-портфолио frontend-разработчика	разработка многостраничного или SPA-портфолио с проектами, контактной формой, адаптивной сеткой, темной темой и базовой аналитикой пользовательских действий
2	Лендинг цифрового продукта с формой заявки	создание промо-страницы с секциями преимуществ, тарифами, отзывами, формой, валидацией, модальным окном и адаптацией под мобильные устройства
3	Интерактивный каталог фильмов или книг	разработка каталога с поиском, фильтрами, карточками, избранным, пагинацией, запросами к API или mock-сервису
4	Task manager с авторизационным сценарием	создание приложения для задач с досками, статусами, фильтрацией, локальным или серверным хранением, формами и защищенными маршрутами
5	Погодный dashboard	разработка интерфейса прогноза погоды с выбором города, историей поиска, графиком, обработкой ошибок API и адаптивной визуализацией

Продолжение на следующей странице

Продолжение таблицы 94

№	Тема проекта	Содержание проекта
6	Интернет-магазин с каталогом и корзиной	создание клиентской части магазина с карточками товаров, фильтрами, корзиной, расчетом стоимости, оформлением заказа и сохранением состояния
7	Сервис бронирования услуг	разработка интерфейса выбора услуги, специалиста, даты и времени с формой клиента, проверкой доступности и подтверждением записи
8	Административная панель управления заявками	создание dashboard для списка заявок, статусов, карточки клиента, поиска, фильтров, таблиц, модальных окон и ролей пользователя
9	Образовательная платформа с личным кабинетом	разработка интерфейса курсов, уроков, прогресса, домашних заданий, уведомлений и пользовательского профиля
10	Мини CRM для работы с клиентами	создание SPA-приложения с карточками клиентов, задачами, комментариями, фильтрацией, статусами и историей действий
11	UI-kit и демонстрационное приложение	разработка набора компонентов с документацией, состояниями, дизайн-токенами и демонстрационными страницами

Продолжение на следующей странице

Продолжение таблицы 94

№	Тема проекта	Содержание проекта
12	Чат-интерфейс для учебного сервиса	создание интерфейса чатов, списка диалогов, сообщений, статусов отправки, поиска, пустых состояний и адаптивной мобильной версии
13	Дашборд аналитики продукта	разработка интерфейса метрик, графиков, фильтров периода, таблиц, карточек показателей и состояния загрузки данных
14	PWA-приложение для личных привычек	создание приложения с трекером привычек, офлайн-режимом, локальным хранением, календарем и уведомительными состояниями
15	Продакшн-ориентированное React-приложение	реализация законченного проекта с TypeScript, маршрутизацией, API, состоянием, тестами, доступностью, сборкой, деплоем и защитой архитектуры

Приложение 5. Чек-лист проверки итогового frontend-проекта.

Таблица 95 – Чек-лист итогового проекта

Направление проверки	Контрольные вопросы
Функциональность	реализованы заявленные пользовательские сценарии, формы, списки, фильтры, навигация и обработка ошибок
Верстка	интерфейс адаптивен, семантичен, кроссбраузерен, имеет корректные состояния и не ломается на целевых разрешениях
React	компоненты декомпозированы, props типизированы, state расположен корректно, хуки используются осмысленно
TypeScript	данные, функции, API-ответы, props и состояния типизированы без необоснованного использования any
API	запросы вынесены в сервисный слой, загрузка и ошибки обработаны, пустые состояния предусмотрены
Тесты	есть проверки ключевых функций, компонентов и основного пользовательского сценария
Доступность	поддержаны фокус, клавиатура, подписи, контраст, альтернативные тексты и понятные ошибки
Производительность	сборка проходит, ресурсы оптимизированы, крупные части загружаются рационально
Безопасность	нет секретов в репозитории, пользовательский ввод обрабатывается, зависимости проверены
Документация	README содержит назначение, стек, запуск, сценарии, тесты, деплой и ограничения

Продолжение на следующей странице

Продолжение таблицы 96

Направление проверки	Контрольные вопросы
Защита	слушатель демонстрирует проект, объясняет архитектуру, отвечает на вопросы и показывает самостоятельность

Приложение 6. Форма паспорта итогового проекта. Паспорт итогового проекта заполняется слушателем до предзащиты и используется при итоговой аттестации. Паспорт помогает проверить полноту постановки задачи, границы проекта, технологический стек и готовность к демонстрации.

Таблица 97 – Паспорт итогового проекта

Раздел паспорта	Содержание
Название проекта	краткое и понятное название frontend-приложения
Цель проекта	какую пользовательскую или бизнес-задачу решает приложение
Целевая аудитория	кто является основным пользователем приложения
Основные сценарии	перечень ключевых действий пользователя
Функциональные требования	что приложение должно выполнять
Нефункциональные требования	адаптивность, доступность, производительность, безопасность, качество кода
Технологический стек	HTML, CSS, TypeScript, React, маршрутизация, состояние, API, тестирование, сборка
Структура данных	основные сущности, поля, источники данных, формат API или mock-данных
Архитектура frontend	страницы, компоненты, сервисы, хуки, состояние, стили, тесты
Сценарии проверки	как проверяется основной пользовательский путь
Ограничения	что входит в текущую версию проекта и что планируется развивать позже
Дальнейшее развитие	какие функции можно добавить после защиты

Приложение 7. Оценочный лист итоговой защиты.

Таблица 99 – Оценочный лист итоговой защиты

№	Критерий	Макс.	Баллы	Комментарий
1	Функциональность и пользовательские сценарии	20		
2	HTML, CSS, адаптивность и доступность интерфейса	10		
3	JavaScript, TypeScript и React-реализация	20		
4	Маршрутизация, состояние и API-интеграция	15		
5	Архитектура, качество кода и сопровождаемость	10		
6	Тестирование, производительность и безопасность	10		
7	Репозиторий, README и демонстрационные материалы	10		
8	Защита и ответы на вопросы	5		
	Итого	100		

Приложение 8. Вопросы к итоговой защите и собеседованию.

Таблица 101 – Примерные вопросы к защите

Блок вопросов	Примерные вопросы
HTML и CSS	почему выбрана такая семантическая структура, как работает адаптивность, где реализованы состояния <code>hover</code> , <code>focus</code> , <code>active</code> и <code>disabled</code>
JavaScript	как обрабатываются события, где происходит преобразование данных, как реализована асинхронная логика и обработка ошибок
TypeScript	какие сущности типизированы, почему выбран конкретный тип, где используются интерфейсы и <code>generics</code>
React	как разделены компоненты, где хранится состояние, какие хуки используются и зачем
Маршрутизация	какие маршруты есть в проекте, как передаются параметры, как обрабатываются защищенные страницы
API	как устроен сервисный слой, как обрабатываются загрузка, ошибка, пустой список, повторный запрос
Тестирование	какие сценарии покрыты тестами, почему выбраны такие проверки, какие дефекты были найдены
Доступность	как обеспечена клавиатурная навигация, фокус, подписи полей, контраст и альтернативные тексты
Производительность	как проверялась скорость загрузки, какие ресурсы оптимизированы, что можно улучшить
Безопасность	какие риски учтены при работе с пользовательским вводом, токенами, зависимостями и переменными окружения
Архитектура	как можно расширить проект, какие решения нужно было бы пересмотреть при росте функциональности

Продолжение на следующей странице

Продолжение таблицы 102

Блок вопросов	Примерные вопросы
Самостоятельность	какой фрагмент проекта был самым сложным, как слушатель искал и исправлял ошибки

Приложение 4. Перечень локальных нормативных актов, на которые должна опираться реализация программы

Реализация Программы обеспечивается локальными нормативными актами ООО «Иннопрог», регулирующими прием, зачисление, режим занятий, электронное обучение, дистанционные образовательные технологии, контроль, аттестацию, отчисление, выдачу документов о квалификации, ФИС ФРДО и обработку персональных данных.

Приложение 5. Лист актуализации программы

Приложение 11. Лист актуализации программы. Лист актуализации используется для фиксации изменений Программы «Frontend-разработчик», связанных с обновлением законодательства, профессиональных стандартов, квалификационных требований, учебных материалов, оценочных средств, цифровой образовательной среды и проектной практики.

Таблица 103 – Лист актуализации программы

№	Дата	Основание изменения	Содержание изменения
1	01.05.2026	Первичное утверждение	Программа введена в действие приказом организации.
2			
3			

Приложение 6. Требования к репозиторию, README и демонстрации

Приложение 1. Термины и сокращения.

Таблица 104 – Термины и сокращения

Термин	Пояснение
Frontend	часть веб-приложения, которая выполняется в браузере пользователя и обеспечивает интерфейс взаимодействия с сервисом
SPA	single page application, одностраничное приложение с клиентской маршрутизацией
UI	user interface, пользовательский интерфейс
UX	user experience, пользовательский опыт
API	интерфейс программного взаимодействия, через который frontend получает или отправляет данные
DOM	объектная модель документа, с которой взаимодействует браузерный JavaScript
TypeScript	надмножество JavaScript с системой типов
React	библиотека для разработки пользовательских интерфейсов на основе компонентов
State	состояние приложения или компонента
Props	данные, передаваемые компоненту от родительского компонента
Hook	функция React для работы с состоянием, эффектами и переиспользуемой логикой
CI/CD	автоматизация проверки, сборки и публикации проекта
ФРДО	федеральный реестр сведений о документах об образовании и (или) о квалификации, документах об обучении

Приложение 9. Требования к репозиторию и README.

Таблица 106 – Содержание README итогового проекта

Раздел README	Требования к содержанию
Название и описание	краткое назначение проекта, целевая аудитория и решаемая задача
Стек	перечень технологий: HTML, CSS, TypeScript, React, Vite, библиотеки, тесты, инструменты
Функциональность	список реализованных возможностей и пользовательских сценариев
Установка	команды установки зависимостей и запуска проекта
Переменные окружения	описание <code>.env.example</code> без раскрытия секретов
Сборка	команды <code>build</code> и <code>preview</code> , особенности публикации
Тесты	команды запуска тестов и краткое описание проверяемых сценариев
Структура проекта	пояснение основных папок, компонентов, сервисов, хуков и стилей
Доступность и качество	результаты самопроверки, известные ограничения и исправленные дефекты
Скриншоты или демо	изображения экранов, ссылка на опубликованную версию или запись демонстрации

Приложение 12. Требования к репозиторию, README и демонстрации.

Для подтверждения освоения Программы и выполнения итогового проекта слушатель представляет проектные материалы в форме, позволяющей проверить самостоятельность, воспроизводимость и качество результата.

Минимальный состав проектных материалов:

- репозиторий или архив проекта с исходным кодом и структурой каталогов;
- файл README.md или иной документ с описанием цели, функций, структуры, установки и запуска проекта;
- описание используемых технологий, зависимостей, версии языка/фреймворка/инструментов;
- инструкции по подготовке окружения, настройке переменных, запуску приложения, тестов или демонстрационного сценария;
- демонстрационные данные, скриншоты, ссылка на развернутую версию или видеодемонстрация при наличии такой возможности;
- краткое описание ограничений, известных проблем и возможных направлений развития проекта.

Материалы должны быть достаточны для повторной проверки проекта преподавателем, экспертом или членом аттестационной комиссии без дополнительных устных пояснений, за исключением вопросов по защите.

Приложение 7. Дорожная карта подготовки выпускного проекта

Приложение 10. Индивидуальная траектория слушателя. Индивидуальная траектория используется для фиксации продвижения слушателя от входной диагностики до итоговой защиты. Она помогает планировать повторение тем, отслеживать задолженности, фиксировать проектные решения и готовиться к аттестации.

Таблица 108 – Шаблон индивидуальной траектории

Раздел	Что фиксируется
Входная диагностика	указать уровень владения HTML, CSS, JavaScript, Git, React и TypeScript
Учебные цели	описать ожидаемый результат обучения и желаемую тему проекта
Риски	зафиксировать темы, которые вызывают трудности, и план их повторения
Проектная тема	выбрать тему итогового проекта и согласовать минимальный функциональный объем
Контрольные точки	указать даты сдачи ключевых модулей и предзащиты
Доработки	вести список замечаний наставника и статус исправления
Готовность к защите	проверить запуск, тесты, README, демонстрацию и ответы на вопросы

Приложение 13. Дорожная карта подготовки выпускного проекта. Дорожная карта используется как методический ориентир для поэтапной подготовки итогового аттестационного проекта по квалификации «Frontend-разработчик».

1. Выбор темы и согласование прикладной задачи с преподавателем или куратором.

2. Формулирование цели проекта, целевой аудитории, пользовательских сценариев и критериев приемки.
3. Подготовка структуры проекта, репозитория, базовой документации и плана реализации.
4. Реализация минимально работоспособной версии и фиксация результата в репозитории.
5. Расширение функциональности, обработка ошибок, подготовка данных, интерфейса или демонстрационного сценария в соответствии со спецификой направления.
6. Проверка качества, тестирование, рефакторинг, оформление инструкций по запуску и использованию.
7. Предзащита или консультационная проверка, устранение замечаний, подготовка презентации и демонстрации.
8. Итоговая защита проекта с ответами на вопросы комиссии и фиксацией результата итоговой аттестации.

Приложение 8. Контрольный лист соответствия требованиям программы

Приложение 14. Контрольный лист соответствия требованиям программы. Контрольный лист применяется для внутренней проверки полноты Программы и готовности ее реализации.

Таблица 110 – Контрольный лист соответствия требованиям программы

№	Проверяемый элемент	Статус	Комментарий
1	Указаны вид программы, квалификация, трудоемкость, форма обучения, язык и срок освоения.	выполнено	
2	Представлены учебный план, календарный учебный график и рабочие программы модулей.	выполнено	
3	Описаны планируемые результаты, профессиональные компетенции и связь с квалификационными ориентирами.	выполнено	
4	Определены организационно-педагогические условия, ЭИОС, ЭО и ДОТ, кадровые и методические условия.	выполнено	
5	Описаны текущий контроль, промежуточная аттестация, итоговая аттестация, пересдача и апелляция.	выполнено	

№	Проверяемый элемент	Статус	Комментарий
6	Определены документы по итогам обучения и перечень локальных актов, обеспечивающих реализацию.	выполнено	