ТЕХНИЧЕСКОЕ ЗАДАНИЕ

на создание автоматизированной информационной системы поиска и подбора репетиторов (АИС «Репетитор»)

1. ОБЩИЕ СВЕДЕНИЯ

1.1. Полное наименование системы: Автоматизированная информационная система взаимодействия преподавателей и учеников «Репетитор» (далее — Система).
1.2. Заказчик: Галкин И.Ю.
1.3. Исполнитель: Галкин И.Ю.
1.4. Основание для проведения работ: Договор №
1.5. Плановые сроки начала и окончания работ: с 12.12.2025 по 22.12.2025

2. НАЗНАЧЕНИЕ И ЦЕЛИ СОЗДАНИЯ СИСТЕМЫ

2.1. Назначение системы

Система предназначена для автоматизации процессов поиска репетиторов, планирования занятий, проведения взаиморасчетов и ведения рейтинга преподавателей.

2.2. Цели создания системы

- Создание единой базы данных репетиторов и учеников.
- Сокращение времени на поиск преподавателя по заданным критериям (предмет, цена, рейтинг).
- Обеспечение безопасной сделки между участниками (холдирование средств).
- Автоматизация расписания (бронирование слотов времени).

3. ХАРАКТЕРИСТИКА ОБЪЕКТОВ АВТОМАТИЗАЦИИ

Объектом автоматизации является процесс взаимодействия между:

1. **Репетитором** (физическое лицо, оказывающее образовательные услуги).
2. **Учеником** (физическое лицо, потребляющее услуги).

3. **Администратором** (сотрудник сервиса, осуществляющий модерацию и поддержку).

Текущее состояние: поиск осуществляется вручную через доски объявлений, расписание ведется в разрозненных календарях, оплата производится бесконтрольно (P2P переводы).

4. ТРЕБОВАНИЯ К СИСТЕМЕ

4.1. Требования к структуре и функционированию системы

Система должна быть реализована в архитектуре «Клиент-Сервер» и включать следующие подсистемы:

1. **Серверная часть (Backend):** Реализована на языке Java.
2. **Клиентская часть (Frontend/Mobile):** Веб-интерфейс и/или мобильное приложение.
3. **Подсистема хранения данных (Database).**

4.2. Требования к функциям (функциональные требования)

4.2.1. Подсистема регистрации и авторизации

- Регистрация пользователей по Email/телефону.
- Ролевая модель: STUDENT, TUTOR, ADMIN.
- Использование JWT (JSON Web Token) для сессий.

4.2.2. Личный кабинет Репетитора

- Заполнение профиля: ФИО, фото, предметы, стеки технологий (для IT), описание опыта.
- Управление расписанием: создание доступных временных слотов (Time Slots).
- Установка тарифов (цена за час).
- Просмотр предстоящих и прошедших занятий.

4.2.3. Личный кабинет Ученика

- Поиск репетиторов с фильтрами:
    - По предмету (Java, Math, English).
    - По цене (диапазон).

    - По рейтингу.
- Бронирование занятия (выбор слота из расписания репетитора).
- Оставление отзыва и оценки (1-5 звезд) после завершения занятия.

4.2.4. Подсистема администрирования

- Модерация новых анкет репетиторов.
- Блокировка пользователей.
- Просмотр статистики системы.

4.3. Требования к видам обеспечения (Технический стек)

4.3.1. Требования к программному обеспечению (Backend)

- **Язык программирования:** Java 17 (LTS) или Java 21 (LTS).
- **Фреймворк:** Spring Boot 3.x.
- **Модули Spring:** Spring Web, Spring Security, Spring Data JPA.
- **Сборка проекта:** Maven или Gradle.
- **ORM:** Hibernate.
- **База данных миграций:** Liquibase или Flyway.
- **API Документация:** Springdoc OpenAPI (Swagger UI).

4.3.2. Требования к СУБД

- **Тип:** Реляционная база данных.
- **ПО:** PostgreSQL версии 14 и выше.

4.3.3. Требования к тестированию

- Покрытие Unit-тестами (JUnit 5, Mockito) не менее 60% бизнес-логики.
- Интеграционные тесты (Testcontainers).

4.4. Требования к надежности и безопасности

- **Безопасность данных:** Пароли должны храниться в хешированном виде (BCrypt).
- **Защита:** Обеспечение защиты от SQL-инъекций и XSS-атак.
- **Персональные данные:** Соответствие ФЗ-152 (хранение данных граждан РФ на территории РФ).
- **Отказоустойчивость:** Система должна восстанавливать работоспособность после сбоя сервера не более чем за 15 минут.

5. СОСТАВ И СОДЕРЖАНИЕ РАБОТ

Перечень этапов выполнения работ:

1. **Техническое проектирование:** Разработка схемы БД (ER-диаграмма), проектирование API (OpenAPI spec).
2. **Разработка прототипа (MVP):** Реализация базовых функций (регистрация, поиск, профиль).
3. **Разработка основной функциональности:** Бронирование, интеграция платежного шлюза (mock или реальный).
4. **Тестирование:** Проведение модульного и интеграционного тестирования.
5. **Развертывание:** Настройка CI/CD, деплой на тестовый сервер.

6. ПОРЯДОК КОНТРОЛЯ И ПРИЕМКИ

6.1. Виды испытаний

- Предварительные испытания (на стороне Разработчика).
- Приемо-сдаточные испытания (с участием Заказчика).

6.2. Критерии приемки

- Успешное прохождение тест-кейсов (Приложение А).
- Отсутствие блокирующих багов (Critical/Blocker).
- Демонстрация полного цикла: Регистрация -> Поиск -> Бронирование -> "Проведение" урока -> Отзыв.

7. ТРЕБОВАНИЯ К ДОКУМЕНТАЦИИ

По результатам работ Исполнитель передает Заказчику:

1. Исходный код системы (репозиторий Git).
2. Инструкция по развертыванию (Deployment Guide / Docker Compose файл).
3. Описание API (Swagger/OpenAPI).
4. Руководство пользователя и Руководство администратора.

Приложение 1. Предварительная схема сущностей (Java Classes Idea)

Для реализации требований предполагается создание следующих сущностей (JPA Entities):

1. User (id, email, passwordHash, role)
2. TutorProfile (userId, description, hourlyRate, List<Skill>)
3. TimeSlot (id, tutorId, startTime, endTime, isBooked)
4. Booking (id, studentId, timeSlotId, status)
5. Review (id, bookingId, rating, comment)