

Федеральное государственное автономное образовательное учреждение высшего
образования

«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий

Кафедра «Информатика и вычислительная техника»

Направление подготовки/ специальность: Системная и программная инженерия

ОТЧЕТ

по проектной практике

Студент: Кузнецов Никита Владимирович Группа: 241 – 326

Место прохождения практики: Московский Политех,
кафедра «СМАРТ-технологии»

Отчет принят с оценкой _____ Дата _____

Руководитель практики: _____

Москва 2025

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

1. Общая информация о проекте:

- Название проекта
- Цели и задачи проекта

2. Общая характеристика деятельности организации (*заказчика проекта*)

- Наименование заказчика
- Организационная структура
- Описание деятельности

3. Описание задания по проектной практике

4. Описание достигнутых результатов по проектной практике

ЗАКЛЮЧЕНИЕ (*выводы о проделанной работе и оценка ценности выполненных задач для заказчика*)

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

ПРИЛОЖЕНИЯ (*при необходимости*)

ВВЕДЕНИЕ

1. Общая информация о проекте

Название проекта:

Разработка платформы жестовой коммуникации ЦНИИ РЖЯ

Цели проекта:

Цель проекта заключается в создании комплексной электронной платформы, предназначенной для систематизации знаний о русском жестовом языке (РЖЯ), повышения доступности и эффективности его изучения, а также обеспечения коммуникации между носителями РЖЯ и широкой аудиторией. Основным элементом — электронный словарь жестов с интерфейсом как на русском, так и на жестовом языке, дополненный функциональными модулями.

Задачи проекта:

- Создание базы данных жестов и их лингвистических характеристик;
- Разработка и внедрение интерфейса словаря, доступного как на русском, так и на жестовом языках;
- Интеграция 3D-анимированного аватара для отображения жестов;
- Внедрение системы морфологической разметки для связи жестов с грамматическими формами;
- Модернизация технологического стека (переход на Go, React, PostgreSQL);

Отдельным подмодулем проекта является платформа аттестации переводчиков РЖЯ, реализующая подачу заявлений, организацию экзаменов и генерацию итоговой документации. Это необходимо для автоматизации и значительного ускорения непростого процесса аттестации переводчиков, которые впоследствии будут развивать электронный словарь.

2. Общая характеристика деятельности организации (заказчика проекта)

Наименование заказчика:

Центральный научно-исследовательский институт русского жестового языка (ЦНИИ РЖЯ)

Организационная структура: ЦНИИ РЖЯ представляет собой автономную некоммерческую научно-исследовательскую организацию. В структуру института входят лингвистические, технические и образовательные подразделения, включая лабораторию лексикографии жестового языка, отдел разработки цифровых инструментов и группу по обучению и сертификации переводчиков.

Описание деятельности: ЦНИИ РЖЯ осуществляет фундаментальные и прикладные исследования в области русского жестового языка, занимается разработкой программного обеспечения, обучением переводчиков и созданием цифровых инструментов — в частности, словарей, систем морфологической разметки и платформ для сертификации. Институт ведёт открытую публикацию лингвистических данных и активно сотрудничает с вузами, включая Московский Политех. Ключевым направлением является создание электронной справочно-аналитической системы "Толковый лексикографический словарь русского жестового языка" и связанных с ней модулей, включая образовательные и технические решения.

3. Описание задания по проектной практике

В рамках учебной практики мне было поручено реализовать серверную часть подпроекта — платформу аттестации переводчиков русского жестового языка (РЖЯ), входящую в состав более крупного проекта «Платформа жестовой коммуникации ЦНИИ РЖЯ». Этот модуль играет ключевую роль в автоматизации процесса сертификации специалистов, которые в дальнейшем будут работать над наполнением словаря РЖЯ.

Главной целью задания было — создание полнофункционального веб-сервиса, который позволяет:

- подавать заявления на участие в аттестации;
- загружать и проверять необходимые документы;
- формировать расписание и управлять экзаменационной сессией;
- проводить онлайн-оценивание через WebSocket в режиме реального времени;
- генерировать выходные документы (например, сертификаты) на основе шаблонов.

Задача была разбита на следующие ключевые этапы:

1. Проектирование архитектуры платформы
 - Разработка структуры базы данных PostgreSQL
 - Определение связей между таблицами (пользователи, документы, экзамены, роли и т.д.)
2. Реализация API на языке Go с использованием Fiber
 - Аутентификация и авторизация с использованием JWT (access/refresh)
 - Обработка multipart-запросов с сохранением данных и файлов
 - Реализация защищённых маршрутов для администратора, переводчика, председателя и экзаменаторов
3. Поддержка реального времени через WebSocket
 - Организация экзаменационных сессий
 - Передача статуса участников и баллов оценки между членами комиссии
4. Генерация ODT-документов

- Использование шаблонов с плейсхолдерами и автозамена на данные из БД
- Подстановка изображений (например, дипломов, паспортов) в отчётные документы

5. Интеграция с основным сайтом проекта

- Совместимость с дизайном и стилем интерфейса
- Подключение административной панели к интерфейсу модераторов и экспертов

4. Описание достигнутых результатов по проектной практике

В результате выполнения задания была спроектирована (см. Приложение 1), реализована и протестирована платформа аттестации переводчиков русского жестового языка (РЖЯ). Основной стек: Go (Golang) + Fiber, GORM, PostgreSQL. Система пока функционирует автономно, но в скором времени планируется сделать ее частью общей платформы жестовой коммуникации ЦНИИ РЖЯ.

Архитектура базы данных (см. Приложение 2)

Проектная база данных построена вокруг следующих ключевых сущностей:

1. Пользователи (User) — модель содержит информацию о пользователях, включая:

- Личные данные в нескольких падежах (необходимость вызвана автозаполнением документов),
- Контактные данные,
- Роль в системе (examiner, student, admin),
- Технические поля (JestID, StoragePath и др.).

```
type User struct {  
    JestID string `gorm:"unique"`  
    Email   string `gorm:"not null;unique"`  
    Role    string `gorm:"not null"`  
    Password string `gorm:"not null"`  
    // ...  
}
```

2. Заявления (Application) — фиксируют информацию о подаче заявки на аттестацию:

- Категория, организация, опыт, обучение и прочее,
- Согласие на обработку персональных данных,
- Статус обработки и причины отклонения (через связную модель ApplicationDecline).

```
type Application struct {  
    UserID      uint  
    ApplicationType string  
    BasisForAttestation string  
    Consent     bool  
    Status      string}
```

3. Документы (UserDocument) — система поддерживает загрузку и классификацию документов по типам:

- DocumentType (паспорт, диплом, СНИЛС, характеристика и др.),
- FilePath — путь к файлу,
- Автоматическое удаление при повторной загрузке.

```
type UserDocument struct {  
    DocumentName string  
    DocumentType string  
    FilePath      string  
}
```

4. Паспорт / диплом (Passport, EducationDocument) — отдельные модели для хранения личных документов с привязкой к UserID.

5. Экзамены (Exam, ExamStudent, ExamExaminer) — реализована полноценная модель:

- Exam включает дату, статус (planned, scheduled, completed), коды комиссии (JestID),
- Связанные таблицы exam_students, exam_examiners отражают участников.

```
type Exam struct {  
    Status      string // planned, scheduled, completed  
    Quorum      int  
    JestID      string  
}
```

6. Оценки (ExamGrade, ExamGradeCriterion) — поддерживается раздельная фиксация оценок по критериям:

- Поддержка флага Abstained,
- Система переоценки,
- Возможность не выставить балл (Score *int).

```
type ExamGradeCriterion struct {  
    CriterionID int  
    Score       *int  
}
```


7. JWT-Аутентификация

Реализована безопасная авторизация с поддержкой:

- access и refresh токенов,
- сохранение RefreshToken в базе,
- обновление access-токена через /refresh.

```
type Token struct {  
    User      string  
    RefreshToken string  
}
```

8. Загрузка и просмотр документов

Загрузка документов реализована через multipart/form-data. Особенности:

- Поддержка разных типов документов: "паспорт_разворот", "паспорт_прописка", "диплом", "снилс" и др.
- Файлы хранятся в папках пользователей, формируемых на основе ФИО и JestID.

```
func SaveFile(file *multipart.FileHeader, path string) error {  
    return c.SaveFile(file, path)  
}
```

9. Проведение экзаменов

Экзаменационная сессия реализована через WebSocket:

- Студенты подключаются к конкретному экзамену,
- Экзаменаторы выставляют оценки в реальном времени,
- Председатель может следить за ходом голосования.

Сервер на Go обрабатывает JSON-сообщения типа init_user, grade_update, discussion_start и др.

10. Генерация документов

После завершения экзамена:

- Автоматически генерируется ODT-файл на основе шаблона,
- Подставляются текстовые поля и изображения из StoragePath,
- Поддержка вставки сканов в плейсхолдеры с alt="{{passport_1}}" и др.

// Пример замены текста

```
node.TextContent = strings.ReplaceAll(node.TextContent, "{{fullname}}",  
user.FullName)
```

API-эндпоинты

1. Общее количество маршрутов

Сервер реализует 50+ HTTP-эндпоинтов, распределённых по 4 основным модулям:

- registration.go — регистрация и подтверждение почты
 - user.go — взаимодействие студента (заявки, документы, экзамены)
 - admin.go — маршруты администратора (работа с экзаменами, пользователями)
 - pages.go — отрисовка HTML-страниц (универсальные маршруты)
-

registration.go:

```
app.Get("/registration", controllers.RegisterPage)  
app.Post("/registration", controllers.Register)  
app.Get("/verify", controllers.Verify)  
app.Post("/confirm", controllers.Confirm)
```

Назначение: регистрация по email, подтверждение почты, установка пароля.

user.go:

Маршруты находятся в RegisterUserRoutes, группа /user и защищены UniversalAuthMiddleware.

Работа с профилем:

```
/user/profile  
/user/maindata  
/user/change/photo
```

Работа с документами:

```
/user/document  
/user/documents/send  
/user/data/correct  
/user/data/aprove  
/user/documents/reason
```

Работа с заявками:

`/user/application`
`/user/create-application` // GET — отображение формы
`/user/create-application` // POST — отправка формы

Работа с экзаменами:

`/user/exams`
`/user/exam/:id`
`/user/exam/check-chairman`
`/user/exam/waiting/:exam_id`
`/user/exam/student/:exam_id/:student_id`
`/user/exam/start/:exam_id`
`/user/exam/start-page/:exam_id`

admin.go:

Админские маршруты, доступные только пользователям с ролью admin.

Студенты:

`/admin/applications`
`/admin/api/student`
`/admin/student/profile`
`/admin/student/documents`
`/admin/application/approve`
`/admin/application/decline`

Экзамены:

`/admin/exam/create`
`/admin/exam/show`
`/admin/exam/save`
`/admin/exam/delete`
`/admin/exam/finalize`
`/admin/api/exam/waiting`

WebSocket:

`/exam/ws`

Поддерживает соединение для реального времени в ходе экзамена:

- пересылка оценок
- контроль подключения экзаменаторов
- координация между председателем и участниками

Пример контроллера: подача заявления

```
func SaveUserApplication(c *fiber.Ctx) error {  
    var app models.Application  
    if err := c.BodyParser(&app); err != nil {  
        return c.Status(400).SendString("Invalid request")  
    }  
    app.CreatedAt = time.Now()  
    database.DB.Create(&app)  
    return c.JSON(app)  
}
```

Назначение: принимает JSON с данными заявления, записывает в таблицу applications.

Пример запроса с клиента (AJAX):

```
fetch("/user/create-application", {  
    method: "POST",  
    headers: { "Content-Type": "application/json" },  
    body: JSON.stringify({  
        UserID: 42,  
        ApplicationType: "Первичная",  
        NativeLanguage: "Русский",  
        // ...  
    }),  
});
```

Примечание

Маршруты покрывают полный цикл: от регистрации до экзамена, включая:

- редактирование профиля;
- загрузку, проверку и отображение документов;
- участие в онлайн-экзамене через WebSocket;
- автоматическую генерацию документов;
- административную проверку и управление.

JWT-авторизация

Внедрена полная схема авторизации и обновления сессии:

- access токен — на 15 минут;
- refresh токен — на 7 дней, хранящийся в HTTP-only cookie;
- Поддержка middleware на Fiber:

```
func AuthMiddleware(c *fiber.Ctx) error {
    token := c.Cookies("access_token")
    claims, err := VerifyToken(token)
    if err != nil {
        return c.Status(fiber.StatusUnauthorized).SendString("Unauthorized")
    }
    c.Locals("user_id", claims.UserID)
    return c.Next()
}
```

Работа с файлами и изображениями

Платформа поддерживает:

- загрузку изображений (JPEG, PNG);
- предварительный просмотр;
- сортировку по типу и названию.

```
func UploadDocuments(c *fiber.Ctx) error {
    form, err := c.MultipartForm()
    files := form.File["passport"]
    for _, file := range files {
        path := fmt.Sprintf("storage/%s", file.Filename)
        c.SaveFile(file, path)
        db.Create(&models.UserDocument{UserID: uid, Path: path, Type:
"passport"})
    }
    return c.SendString("Uploaded")
}
```

Онлайн-экзамены через WebSocket

Был реализован модуль реального времени:

- Подключение экзаменаторов и председателя;
- Передача информации о студентах;
- Ввод и пересылка оценок;
- Завершение экзамена по команде председателя.

```
func HandleWebSocket(c *websocket.Conn) {
    for {
        msg := Message{}
        if err := c.ReadJSON(&msg); err != nil {
            break
        }
        broadcast <- msg // пересылаем другим участникам
    }
}
```

Генерация документов ODT

Автоматизирована генерация заявлений через шаблоны .odt:

- Данные пользователя подставляются через zip + xml парсинг;
- Изображения вставляются по alt text плейсхолдерам ({{passport}}, {{snils}});

Пример вставки текста:

```
func ReplacePlaceholder(content string, placeholder string, value string) string {
    return strings.ReplaceAll(content, "{{"+placeholder+"}}", value)
}
```

Развёртывание и публикация (см. Приложение 3)

- Платформа развернута на сервере с nginx;
- И доступна по адресу: <https://att.cnii-jest.ru>

ЗАКЛЮЧЕНИЕ

В ходе проектной практики была практически полностью реализована платформа аттестации переводчиков русского жестового языка (РЖЯ). Работа включала проектирование и реализацию архитектуры базы данных, построение REST API, интеграцию с системой WebSocket для проведения онлайн-экзаменов, а также генерацию документов на основе шаблонов. Результатом стало создание backend-модуля, совместимого в будущем с основной платформой «Платформа жестовой коммуникации ЦНИИ РЖЯ». Разработанная система решает актуальные задачи автоматизации аттестации переводчиков, включая:

- Подачу и верификацию заявлений;
- Загрузку и классификацию документов;
- Проведение онлайн-экзамена с синхронной оценкой;
- Генерацию отчётных документов.

Проект имеет важную практическую и социальную ценность: он обеспечивает цифровизацию процессов, ранее выполнявшихся вручную, и значительно повышает доступность и быстроту проведения процедуры аттестации. Для заказчика — ЦНИИ РЖЯ — данный инструмент облегчает администрирование, повышает прозрачность и снижает издержки.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Документация Go Fiber — <https://docs.gofiber.io>
2. GORM ORM для Golang — <https://gorm.io/docs>
3. WebSocket API — MDN:
<https://developer.mozilla.org/ru/docs/Web/API/WebSocket>
4. OpenDocument Format (ODF) XML спецификация — <https://docs.oasis-open.org>
5. PostgreSQL Documentation — <https://postgrespro.ru/docs>
6. Внутренние материалы ЦНИИ РЖЯ (необходимая для корректной логики процедуры информация о порядке проведения аттестации) — <https://cniijest.ru>

ПРИЛОЖЕНИЕ

Рисунок 1 Архитектура системы на Go + PostgreSQL	18
Рисунок 2 Общая структура	19
Рисунок 3 Первые 3 таблицы подробно.....	20
Рисунок 4 Следующие 3 таблицы подробно	21
Рисунок 5 Следующие 4 таблицы подробно	22
Рисунок 6 Следующие 3 таблицы подробно	23
Рисунок 7 Последние 2 таблицы подробно	24
Рисунок 8 index.....	25
Рисунок 9 registration	25
Рисунок 10 user/profile	26
Рисунок 11 user/document	27
Рисунок 12 user/application	28
Рисунок 13 user/exams.....	28
Рисунок 14 user/exam/:id.....	29
Рисунок 15 admin.....	29
Рисунок 16 admin/user/list.....	30
Рисунок 17 admin/student/profile	30
Рисунок 18 admin/exam/planning.....	31
Рисунок 19 admin/exam/create	31
Рисунок 20 admin/exam/scheduled.....	32
Рисунок 21 user/exam/start-page/:id.....	32
Рисунок 22 user/exam/start-page/:id + modal	33
Рисунок 23 user/exam/start/:id.....	33
Рисунок 24 user/exam/student/:id/:id.....	34

Приложение 1. Архитектура серверной платформы

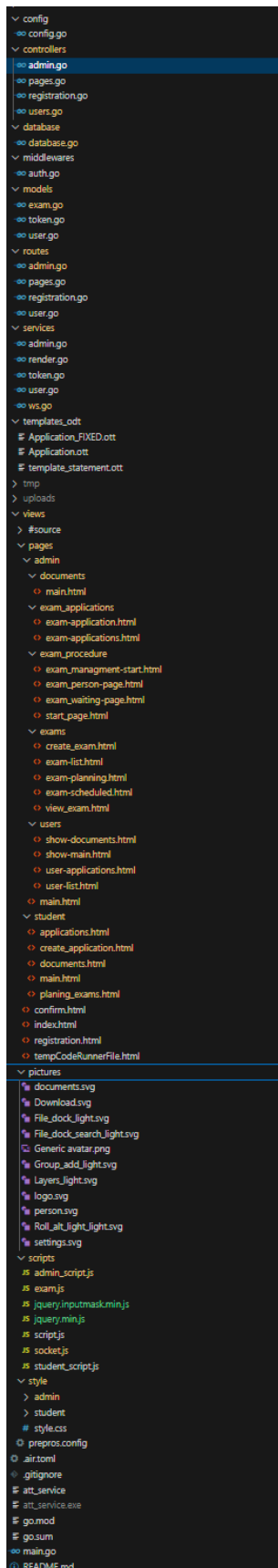


Рисунок 1 Архитектура системы на Go + PostgreSQL

Приложение 2. Структура базы данных

Список отношений			
Схема	Имя	Тип	Владелец
public	application_declines	таблица	postgres
public	application_declines_id_seq	последовательность	postgres
public	applications	таблица	postgres
public	applications_id_seq	последовательность	postgres
public	concluding_speeches	таблица	postgres
public	concluding_speeches_id_seq	последовательность	postgres
public	education_documents	таблица	postgres
public	education_documents_id_seq	последовательность	postgres
public	email_verifications	таблица	postgres
public	email_verifications_id_seq	последовательность	postgres
public	exam_examiners	таблица	postgres
public	exam_examiners_id_seq	последовательность	postgres
public	exam_grade_criteria	таблица	postgres
public	exam_grade_criteria_id_seq	последовательность	postgres
public	exam_grades	таблица	postgres
public	exam_grades_id_seq	последовательность	postgres
public	exam_students	таблица	postgres
public	exam_students_id_seq	последовательность	postgres
public	exams	таблица	postgres
public	exams_id_seq	последовательность	postgres
public	introductory_speeches	таблица	postgres
public	introductory_speeches_id_seq	последовательность	postgres
public	passports	таблица	postgres
public	passports_id_seq	последовательность	postgres
public	tokens	таблица	att_user
public	tokens_id_seq	последовательность	att_user
public	user_documents	таблица	postgres
public	user_documents_id_seq	последовательность	postgres
public	users	таблица	postgres
public	users_id_seq	последовательность	postgres

(30 строк)

Рисунок 2 Общая структура

att_service_db=# \d application_declines				
Таблица "public.application_declines"				
Столбец	Тип	Правило сортировки	Допустимость NULL	По умолчанию
id	bigint		not null	nextval('application_declines_id_seq'::regclass)
application_id	bigint			
reasons	text			
explanation	text			
created_at	timestamp with time zone			
Индексы:				
"application_declines_pkey" PRIMARY KEY, btree (id)				
"idx_application_declines_application_id" UNIQUE, btree (application_id)				
Ограничения внешнего ключа:				
"fk_applications_decline" FOREIGN KEY (application_id) REFERENCES applications(id)				
att_service_db=# \d applications				
Таблица "public.applications"				
Столбец	Тип	Правило сортировки	Допустимость NULL	По умолчанию
id	integer		not null	nextval('applications_id_seq'::regclass)
user_id	bigint		not null	
application_type	character varying(255)			
application_number	character varying(255)			
native_language	character varying(255)			
citizenship	character varying(255)			
marital_status	character varying(255)			
organization	character varying(255)			
job_position	character varying(255)			
requested_category	character varying(255)			
basis_for_attestation	character varying(255)			
existing_category	character varying(255)			
existing_category_term	character varying(255)			
work_experience	character varying(255)			
current_position_experience	character varying(255)			
awards_info	character varying(255)			
training_info	character varying(255)			
memberships	character varying(255)			
consent	boolean			
created_at	timestamp with time zone			
updated_at	timestamp with time zone			
is_latest	boolean			true
status	character varying(255)			
decline_reason	character varying(255)			
decline_explanation	character varying(255)			
Индексы:				
"applications_pkey" PRIMARY KEY, btree (id)				
Ограничения внешнего ключа:				
"applications_user_id_fkey" FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE CASCADE				
Ссылки извне:				
TABLE "application_declines" CONSTRAINT "fk_applications_decline" FOREIGN KEY (application_id) REFERENCES applications(id)				
att_service_db=# \d concluding_speeches				
Таблица "public.concluding_speeches"				
Столбец	Тип	Правило сортировки	Допустимость NULL	По умолчанию
id	integer		not null	nextval('concluding_speeches_id_seq'::regclass)
city	character varying(100)		not null	
day	integer		not null	
month	character varying(20)		not null	
year	integer		not null	
attestation_number	integer		not null	
Индексы:				
"concluding_speeches_pkey" PRIMARY KEY, btree (id)				

Рисунок 3 Первые 3 таблицы подробно

```
att_service_db=# \d education_documents
```

Столбец	Тип	Правило сортировки	Допустимость NULL	По умолчанию
id	integer		not null	nextval('education_documents_id_seq'::regclass)
institution_name	text			
city_name	text			
diploma_series_number	character varying(50)			
diploma_reg_number	text			
issue_date	timestamp with time zone			
specialty_code_name	text			
qualification_level	text			
user_id	bigint		not null	
diploma_series	text			
specialty_code	text			
created_at	timestamp with time zone			
updated_at	timestamp with time zone			
deleted_at	timestamp with time zone			

Индексы:

```
"education_documents_pkey" PRIMARY KEY, btree (id)
"idx_education_documents_deleted_at" btree (deleted_at)
```



```
att_service_db=# \d email_verifications
```

Столбец	Тип	Правило сортировки	Допустимость NULL	По умолчанию
id	bigint		not null	nextval('email_verifications_id_seq'::regclass)
email	text		not null	
link	text		not null	
expires_at	timestamp with time zone		not null	
created_at	timestamp with time zone			
updated_at	timestamp with time zone			

Индексы:

```
"email_verifications_pkey" PRIMARY KEY, btree (id)
```



```
att_service_db=# \d exam_examiners
```

Столбец	Тип	Правило сортировки	Допустимость NULL	По умолчанию
exam_id	bigint		not null	
user_id	bigint		not null	
id	bigint		not null	nextval('exam_examiners_id_seq'::regclass)
jest_id	text			

Индексы:

```
"exam_examiners_pkey" PRIMARY KEY, btree (exam_id, user_id)
```

Ограничения внешнего ключа:

```
"fk_exam_examiner_exam" FOREIGN KEY (exam_id) REFERENCES exams(id) ON DELETE CASCADE
"fk_exam_examiners_exam" FOREIGN KEY (exam_id) REFERENCES exams(id)
"fk_exam_examiners_user" FOREIGN KEY (user_id) REFERENCES users(id)
"fk_exams_examiners" FOREIGN KEY (exam_id) REFERENCES exams(id)
```

Рисунок 4 Следующие 3 таблицы подробно

att_service_db=# \d exam_grade_criteria

Таблица "public.exam_grade_criteria"				
Столбец	Тип	Правило сортировки	Допустимость NULL	По умолчанию
id	bigint		not null	nextval('exam_grade_criteria_id_seq'::regclass)
created_at	timestamp with time zone			
updated_at	timestamp with time zone			
deleted_at	timestamp with time zone			
grade_id	bigint			
criterion_id	bigint			
score	bigint			

Индексы:

"exam_grade_criteria_pkey" PRIMARY KEY, btree (id)

"idx_exam_grade_criteria_deleted_at" btree (deleted_at)

Ограничения внешнего ключа:

"fk_exam_grades_criteria" FOREIGN KEY (grade_id) REFERENCES exam_grades(id)

att_service_db=# \d exam_grades

Таблица "public.exam_grades"				
Столбец	Тип	Правило сортировки	Допустимость NULL	По умолчанию
id	bigint		not null	nextval('exam_grades_id_seq'::regclass)
created_at	timestamp with time zone			
updated_at	timestamp with time zone			
deleted_at	timestamp with time zone			
exam_id	bigint			
examiner_id	bigint			
student_id	bigint			
qualification	text			
specialization	text			
recommendation	text			
abstained	boolean			

Индексы:

"exam_grades_pkey" PRIMARY KEY, btree (id)

"idx_exam_grades_deleted_at" btree (deleted_at)

Ссылки извне:

TABLE "exam_grade_criteria" CONSTRAINT "fk_exam_grades_criteria" FOREIGN KEY (grade_id) REFERENCES exam_grades(id)

att_service_db=# \d exam_students

Таблица "public.exam_students"				
Столбец	Тип	Правило сортировки	Допустимость NULL	По умолчанию
exam_id	bigint		not null	
user_id	bigint		not null	
id	bigint		not null	nextval('exam_students_id_seq'::regclass)
jest_id	text			
completed	boolean			
discussion	boolean			

Индексы:

"exam_students_pkey" PRIMARY KEY, btree (exam_id, user_id)

Ограничения внешнего ключа:

"fk_exam_student_exam" FOREIGN KEY (exam_id) REFERENCES exams(id) ON DELETE CASCADE

"fk_exam_students_exam" FOREIGN KEY (exam_id) REFERENCES exams(id)

"fk_exam_students_user" FOREIGN KEY (user_id) REFERENCES users(id)

"fk_exams_students" FOREIGN KEY (exam_id) REFERENCES exams(id)

att_service_db=# \d exams

Таблица "public.exams"				
Столбец	Тип	Правило сортировки	Допустимость NULL	По умолчанию
id	bigint		not null	nextval('exams_id_seq'::regclass)
created_at	timestamp with time zone			
updated_at	timestamp with time zone			
deleted_at	timestamp with time zone			
date	timestamp with time zone			
commission_start	timestamp with time zone			
commission_end	timestamp with time zone			
status	character varying(50)		not null	'planned'::character varying
chairman_id	bigint			
quorum	bigint			
c_chairman_id	bigint			
secretary_id	bigint			
jest_id	character varying(20)			

Индексы:

"exams_pkey" PRIMARY KEY, btree (id)

"idx_exams_deleted_at" btree (deleted_at)

Ссылки извне:

TABLE "exam_examiners" CONSTRAINT "fk_exam_examiner_exam" FOREIGN KEY (exam_id) REFERENCES exams(id) ON DELETE CASCADE

TABLE "exam_examiners" CONSTRAINT "fk_exam_examiners_exam" FOREIGN KEY (exam_id) REFERENCES exams(id)

TABLE "exam_students" CONSTRAINT "fk_exam_student_exam" FOREIGN KEY (exam_id) REFERENCES exams(id) ON DELETE CASCADE

TABLE "exam_students" CONSTRAINT "fk_exam_students_exam" FOREIGN KEY (exam_id) REFERENCES exams(id)

TABLE "exam_examiners" CONSTRAINT "fk_exams_examiners" FOREIGN KEY (exam_id) REFERENCES exams(id)

TABLE "exam_students" CONSTRAINT "fk_exams_students" FOREIGN KEY (exam_id) REFERENCES exams(id)

Рисунок 5 Следующие 4 таблицы подробно

att_service_db=# \d introductory_speeches

Столбец		Тип	Таблица "public.introductory_speeches"		По умолчанию
			Правило сортировки	Допустимость NULL	
id		integer		not null	nextval('introductory_speeches_id_seq'::regclass)
city		character varying(100)		not null	
day		integer		not null	
month		character varying(20)		not null	
year		integer		not null	
attestation_number		integer		not null	
commission_member1_lastname		character varying(100)			
commission_member1_firstname		character varying(100)			
commission_member1_patronymic		character varying(100)			
commission_member1_info		text			
commission_member2_lastname		character varying(100)			
commission_member2_firstname		character varying(100)			
commission_member2_patronymic		character varying(100)			
commission_member2_info		text			
commission_member3_lastname		character varying(100)			
commission_member3_firstname		character varying(100)			
commission_member3_patronymic		character varying(100)			
commission_member3_info		text			
commission_member4_lastname		character varying(100)			
commission_member4_firstname		character varying(100)			
commission_member4_patronymic		character varying(100)			
commission_member4_info		text			
commission_member5_lastname		character varying(100)			
commission_member5_firstname		character varying(100)			
commission_member5_patronymic		character varying(100)			
commission_member5_info		text			
commission_member6_lastname		character varying(100)			
commission_member6_firstname		character varying(100)			
commission_member6_patronymic		character varying(100)			
commission_member6_info		text			
commission_member7_lastname		character varying(100)			
commission_member7_firstname		character varying(100)			
commission_member7_patronymic		character varying(100)			
commission_member7_info		text			
commission_member8_lastname		character varying(100)			
commission_member8_firstname		character varying(100)			
commission_member8_patronymic		character varying(100)			
commission_member8_info		text			
commission_member9_lastname		character varying(100)			
commission_member9_firstname		character varying(100)			
commission_member9_patronymic		character varying(100)			
commission_member9_info		text			
commission_member10_lastname		character varying(100)			
commission_member10_firstname		character varying(100)			
commission_member10_patronymic		character varying(100)			
commission_member10_info		text			

Индексы:

"introductory_speeches_pkey" PRIMARY KEY, btree (id)

att_service_db=# \d passports

Столбец		Тип	Таблица "public.passports"		По умолчанию
			Правило сортировки	Допустимость NULL	
id		bigint		not null	nextval('passports_id_seq'::regclass)
user_id		bigint		not null	
passport_series		character varying(20)			
passport_number		character varying(20)			
passport_issued_by		character varying(200)			
passport_issue_date		timestamp with time zone			
passport_division_code		character varying(20)			
birth_date		timestamp with time zone			
birth_place		character varying(200)			
registration_address		character varying(200)			
created_at		timestamp with time zone			
updated_at		timestamp with time zone			
deleted_at		timestamp with time zone			

Индексы:

"passports_pkey" PRIMARY KEY, btree (id)

"idx_passports_deleted_at" btree (deleted_at)

Ограничения внешнего ключа:

"passports_user_id_fkey" FOREIGN KEY (user_id) REFERENCES users(id)

att_service_db=# \d tokens

Столбец		Тип	Таблица "public.tokens"		По умолчанию
			Правило сортировки	Допустимость NULL	
id		bigint		not null	nextval('tokens_id_seq'::regclass)
created_at		timestamp with time zone			
updated_at		timestamp with time zone			
deleted_at		timestamp with time zone			
user		text		not null	
refresh_token		text		not null	

Индексы:

"tokens_pkey" PRIMARY KEY, btree (id)

"idx_tokens_deleted_at" btree (deleted_at)

Рисунок 6 Следующие 3 таблицы подробно

att_service_db=# \d user_documents				
Таблица "public.user_documents"				
Столбец	Тип	Правило сортировки	Допустимость NULL	По умолчанию
id	integer		not null	nextval('user_documents_id_seq'::regclass)
user_id	bigint		not null	
document_name	text		not null	
document_type	text		not null	
file_path	text		not null	
created_at	timestamp with time zone			
updated_at	timestamp with time zone		not null	
deleted_at	timestamp with time zone			
Индексы:				
"user_documents_pkey" PRIMARY KEY, btree (id)				
"idx_user_documents_deleted_at" btree (deleted_at)				
Ограничения внешнего ключа:				
"user_documents_user_id_fkey" FOREIGN KEY (user_id) REFERENCES users(id) ON UPDATE CASCADE ON DELETE CASCADE				
att_service_db=# \d users				
Таблица "public.users"				
Столбец	Тип	Правило сортировки	Допустимость NULL	По умолчанию
id	bigint		not null	nextval('users_id_seq'::regclass)
created_at	timestamp with time zone		not null	
updated_at	timestamp with time zone		not null	
deleted_at	timestamp with time zone			
email	character varying(100)		not null	
password	character varying(100)		not null	
role	character varying(100)		not null	
refresh_token	text			
sex	character varying(10)			
mobile_phone	character varying(20)			
work_phone	character varying(20)			
snils	character varying(20)			
status	character varying(255)		not null	'pending'::character varying
rejection_reason	text			
approved	boolean		not null	false
surname_in_ip	character varying(100)			
surname_in_rp	character varying(100)			
surname_in_dp	character varying(100)			
name_in_ip	character varying(100)			
name_in_rp	character varying(100)			
name_in_dp	character varying(100)			
lastname_in_ip	character varying(100)			
lastname_in_rp	character varying(100)			
lastname_in_dp	character varying(100)			
mail	character varying(100)			
confirmed	boolean			false
jest_id	character varying(20)			
storage_path	character varying(255)			
application_status	text			
Индексы:				
"users_pkey" PRIMARY KEY, btree (id)				
"idx_users_deleted_at" btree (deleted_at)				
"uni_users_email" UNIQUE CONSTRAINT, btree (email)				
"uni_users_jest_id" UNIQUE CONSTRAINT, btree (jest_id)				
Ссылки извне:				
TABLE "applications" CONSTRAINT "applications_user_id_fkey" FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE CASCADE				
TABLE "exam_examiners" CONSTRAINT "fk_exam_examiners_user" FOREIGN KEY (user_id) REFERENCES users(id)				
TABLE "exam_students" CONSTRAINT "fk_exam_students_user" FOREIGN KEY (user_id) REFERENCES users(id)				
TABLE "passports" CONSTRAINT "passports_user_id_fkey" FOREIGN KEY (user_id) REFERENCES users(id)				
TABLE "user_documents" CONSTRAINT "user_documents_user_id_fkey" FOREIGN KEY (user_id) REFERENCES users(id) ON UPDATE CASCADE ON DELETE CASCADE				

Рисунок 7 Последние 2 таблицы подробно

Приложение 3. Интерфейс платформы



Аттестация переводчиков
русского жестового языка

Авторизация

Войти

Зарегистрироваться

Рисунок 8 index



Аттестация переводчиков
русского жестового языка

Регистрация

Введите Вашу почту. Для подтверждения мы вышлем ссылку на вашу почту.

Отправить

Уже зарегистрированы?

Рисунок 9 registration



Личные данные

Документы

Мои заявления

Назначенные экзамены



	Именит. падеж:	Родит. падеж:	Дател. падеж:
Фамилия:	Кузнецов	Кузнецова	Кузнецова
Имя:	Никита	Никиты	Никиту
Отчество:	Владимирович	Владимировича	Владимировича

Пол: ☒ Мужской

☐ Женский

Мобильный телефон:

+7 (917) 129-46-22

Рабочий телефон: +7 () - - -

Электронная почта:

g@mail

Почтовый адрес:

hvjskjkss

Сохранить

Рисунок 10 user/profile



Личные данные

Документы

Мои заявления

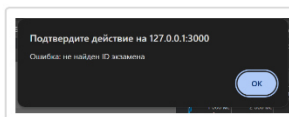
Назначенные экзамены

Паспорт

Серия: **87-65** №: **987654** Дата выдачи: **03.01.0001** Кем выдан: **kjhgfdghj** Код подразделения: **987-659**

Дата рождения: **04.01.0001** Место рождения: **jhgfdgh** Адрес регистрации: **kjhgfdghj**

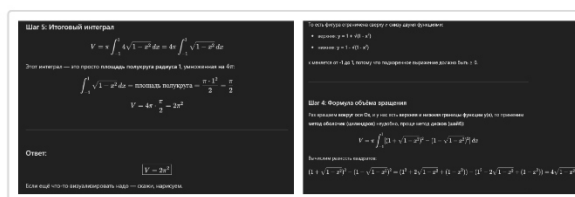
Сканы паспорта:



Снилс

№: **234-567-890 98**

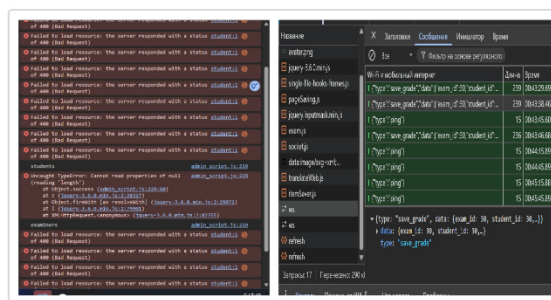
Скан:



Диплом об основном образовании

№: **6543876543**

Сканы:



Сохранить

Рисунок 11 user/document



Личные данные

Документы

Мои заявления

Назначенные экзамены

Создать заявку на аттестацию [Создать](#)

Заявка на аттестацию	Подтверждено
Заявка на аттестацию	Подтверждено
Заявка на аттестацию	Отклонено Причина
Заявка на аттестацию	Подтверждено
Заявка на аттестацию	Отклонено Причина
Заявка на аттестацию	Подтверждено
Заявка на аттестацию	Подтверждено
Заявка на аттестацию	Подтверждено
Заявка на аттестацию	Отклонено Причина
Заявка на аттестацию	Подтверждено
Заявка на аттестацию	Подтверждено
Заявка на аттестацию	Подтверждено

Рисунок 12 user/application



Личные данные

Документы

Мои заявления

Назначенные экзамены

Экзамен от 16.05.2025

[Посмотреть экзамен](#)

Дата экзамена:

16.05.2025

Рисунок 13 user/exams



назад

Дата проведения экзамена: 16.05.2025

Код экзамена: 06-30-1

Начало действия комиссии: 06.05.2025

Конец действия комиссии: 31.05.2025

Экзаменаторы:

	Егоров Влад Андреевич	Председатель
	Шипелов Максим Андреевич	Экзаменатор

Экзаменуемые:

	Иванов Виктор Ильич	
	Кузнецов Никита Владимирович	

Рисунок 14 user/exam/:id



Пользователи



Экзамены



Заявки на экзамены



Шаблоны



Документы

Рисунок 15 admin



Аттестация переводчиков
русского жестового языка

Все пользователи


Заявки

назад

Поиск по ФИО

Поиск

Все пользователи




Кузнецов Никита Владимирович

Администратор

Посмотреть аккаунт

Удалить аккаунт




Иванов Виктор Ильич

Аттестуемый

Посмотреть аккаунт

Удалить аккаунт




Кузнецов Никита Владимирович

Аттестуемый

Посмотреть аккаунт

Удалить аккаунт




Егоров Влад Андреевич

Экзаменатор

Посмотреть аккаунт

Удалить аккаунт



Шипелов Максим Андреевич

Экзаменатор

Посмотреть аккаунт

Удалить аккаунт

Рисунок 16 admin/user/list



Аттестация переводчиков
русского жестового языка

Статус: Подтвержден


Роль: Аттестуемый



Личные данные

Документы

назад



	Именит. падеж:	Родит. падеж:	Дател. падеж:
Фамилия:	Кузнецов	Кузнецова	Кузнецова
Имя:	Никита	Никиты	Никиту
Отчество:	Владимирович	Владимировича	Владимировича

Пол: ☒ Мужской ☐ Женский

Мобильный телефон: 9171294622

Рабочий телефон: +7 () - - -

Электронная почта: g@mail

Почтовый адрес: hvjskjkss

Рисунок 17 admin/student/profile



Аттестация переводчиков
русского жестового языка

Прошедшие экзамены

Запланированные экзамены

Назначенные экзамены

назад

Запланировать экзамен

Экзамен от 09.05.2025

Назначить

Посмотреть экзамен

Экзамен от 16.05.2025

Назначить

Посмотреть экзамен

Рисунок 18 admin/exam/planning



Аттестация переводчиков
русского жестового языка

назад

Дата проведения экзамена: 21.05.2025

Предварительный код экзамена: 06-30-2

Начало действия комиссии: 08.05.2025

Конец действия комиссии: 30.05.2025

Выберите экзаменаторов:

Убрать всех

Егоров Влад Андреевич

Председатель

Шипелов Максим Андреевич

Секретарь

Выберите экзаменуемых:

Убрать всех

Иванов Виктор Ильич

Кузнецов Никита Владимирович

Назначить

Сохранить

Рисунок 19 admin/exam/create



Аттестация переводчиков
русского жестового языка

Прошедшие экзамены

Запланированные экзамены

Назначенные экзамены

назад

Экзамен от 09.05.2025

Отменить

Посмотреть экзамен

Экзамен от 16.05.2025

Отменить

Посмотреть экзамен

Экзамен от 16.05.2025

Отменить

Посмотреть экзамен

Экзамен от 21.05.2025

Отменить

Посмотреть экзамен

Рисунок 20 admin/exam/scheduled

Вступительная речь

Уважаемые участники экзамена!

Мы рады приветствовать вас на экзамене по русскому жестовому языку. Этот экзамен является важным этапом в вашей профессиональной карьере. Мы желаем вам удачи и успехов! Надеемся, что вы продемонстрируете свои знания наилучшим образом.

Посмотреть подключившихся

Рисунок 21 user/exam/start-page/:id

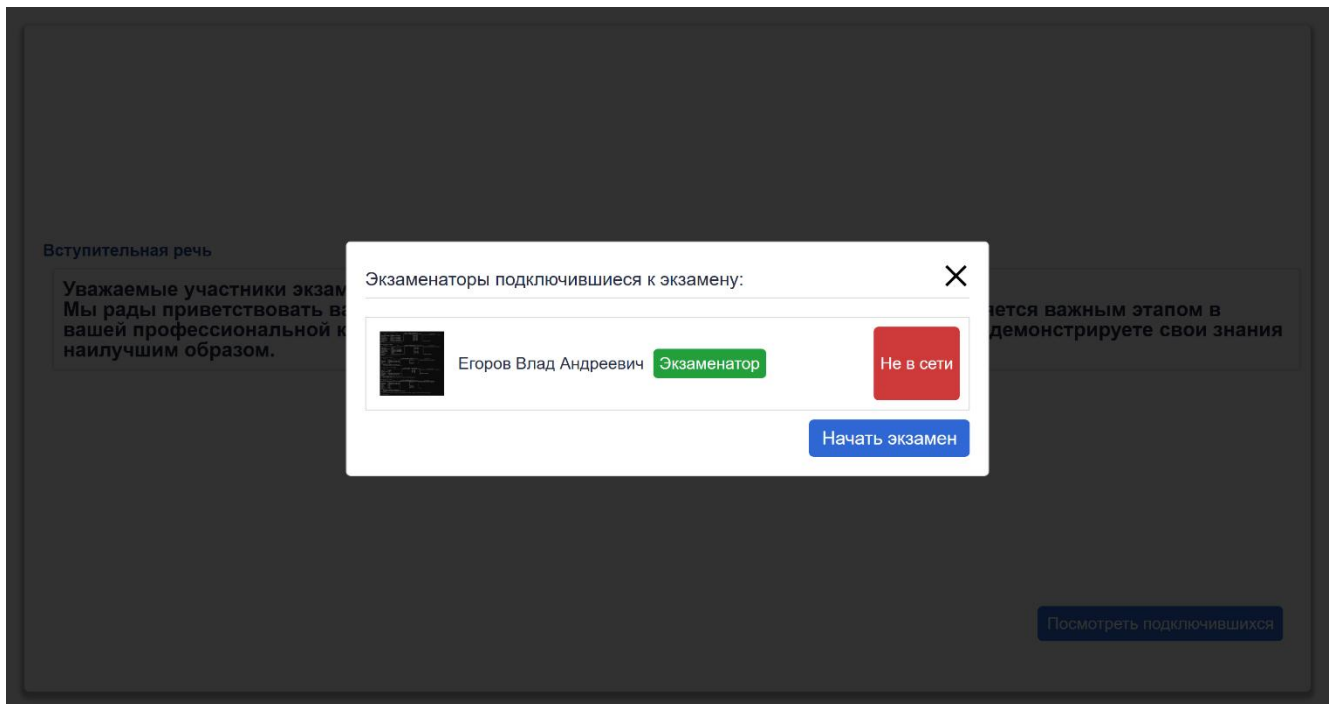


Рисунок 22 user/exam/start-page/:id + modal

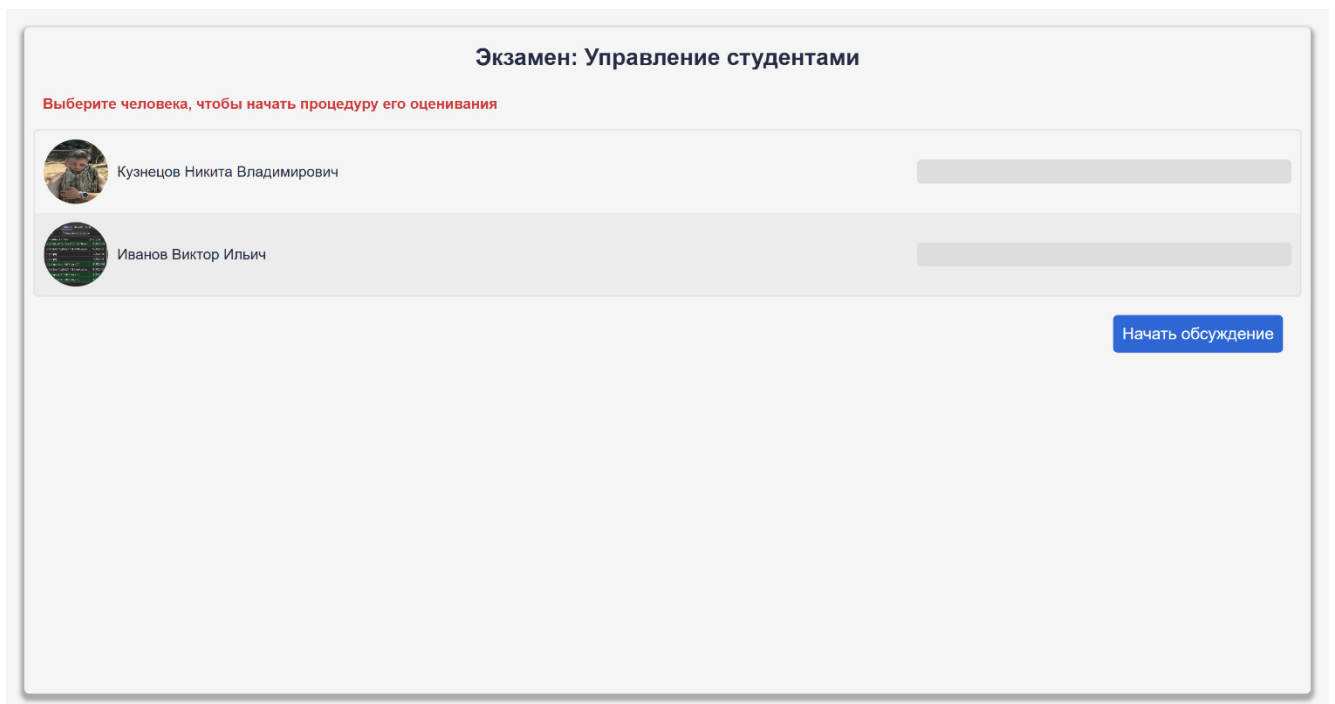


Рисунок 23 user/exam/start/:id

Оценочный лист

К Протоколу № 06-30-2/2 от 19.05.2025

Аттестуемый: Кузнецов Никита Владимирович

Запрашиваемая квалификация: 03.01600.04

Заявленная специализация:

Всего баллов за экзамен: 28



№ п/п	Показатель (критерий оценки)	Баллы					
		0	1	2	3	4	5
1	Задание 1 Ответ по билету	■					
2	Обратный перевод (Последовательный письменный, Жестовый → русский)		■				
3	Прямой перевод (Синхронный, Русский → жестовый)	■					
4	Прямой перевод (Последовательный, Русский → жестовый)				■		
5	Обратный перевод (Синхронный устный, Русский → жестовый)			■			
6	Четкость исполнения жестов					■	
7	Скорость перевода				■		
8	Мимика (Выразительность лица)					■	
9	Артикуляция			■			
10	Лексический запас жестов					■	
11	Внешний вид согласно дресс-коду						■

Рекомендации: пример

Решение о присвоении квалификации:

Присваиваемая квалификация: 03.01600.03 Переводчик русского жестового языка I категории (6 уровень квалификации)

Допуск (Специализация): образование

☐ Воздержаться

Рисунок 24 user/exam/student/:id/:id