

# ЗАДАНИЕ 3.1 — REST API

HTTP метод и URL:

POST /api/v1/users

---

Параметр	Тип	Обязательный	Ограничения
firstName	string	да	1–50 символов
lastName	string	да	1–50 символов
username	string	да	уникальный, 3–30 символов
password	string	да	минимум 8 символов, минимум 1 буква в верхнем регистре, 1 цифра, 1 спецсимвол
captchaToken	string	да	токен captcha

## Входные параметры

Пример запроса:

```
{
  "firstName": "Ivan",
  "lastName": "Ivanov",
  "username": "ivan",
  "password": "StrongPass1!",
  "captchaToken": "03AFcWeA6..."
}
```

---

# Успешный ответ

## HTTP статус:

201 Created

## Тело ответа:

Параметр	Тип	Обязательный	Описание
userId	string (uuid)	да	ID пользователя
username	string	да	username
message	string	да	результат операции

## Пример успешного ответа:

```
{
  "userId": "9b1deb4d-3b7d-4bad-9bdd-2b0d7b3dcb6d",
  "username": "ivan",
  "message": "User Register Successfully"
}
```

# Ответы с ошибками

## Формат ошибки:

Параметр	Тип	Обязательный
errorCode	string	да
message	string	да

## 400 Bad Request — ошибка пароля:

```
{
  "errorCode": "INVALID_PASSWORD",
  "message": "Password must have at least one non alphanumeric character, one digit ('0'-'9'), one uppercase ('A'-'Z'), one lowercase ('a'-'z') and be at least 8 characters long."
}
```

---

### 409 Conflict — пользователь уже существует:

```
{
  "errorCode": "USER_ALREADY_EXISTS",
  "message": "User exists!"
}
```

---

### 400 Bad Request — не пройдена captcha:

```
{
  "errorCode": "CAPTCHA_REQUIRED",
  "message": "Please verify captcha to register!"
}
```

---

### 500 Internal Server Error:

```
{
  "errorCode": "INTERNAL_ERROR",
  "message": "Unexpected server error"
}
```

---

### Коды ответов:

Код	Тип
201	Успешная регистрация
400	Клиентская ошибка
409	Клиентская ошибка
500	Серверная ошибка

---

### OPENAPI YAML:

```
openapi: 3.0.3
info:
  title: Book Store User Registration API
  version: 1.0.0
```

```
paths:
  /api/v1/users:
    post:
      summary: Register new user
      requestBody:
        required: true
        content:
          application/json:
            schema:
              type: object
              required:
                - firstName
                - lastName
                - username
                - password
                - captchaToken
              properties:
                firstName:
                  type: string
                lastName:
                  type: string
                username:
                  type: string
                password:
                  type: string
                  minLength: 8
                captchaToken:
                  type: string
      responses:
        "201":
          description: User registered successfully
        "400":
          description: Validation error
        "409":
          description: User already exists
        "500":
          description: Internal server error
```

## **ЗАДАНИЕ 3.2 — АЛГОРИТМ СОЗДАНИЯ ПОЛЬЗОВАТЕЛЯ**

## Пошаговый алгоритм:

1. Backend принимает POST `/api/v1/users`
2. Backend проверяет наличие тела запроса
3. Backend проверяет обязательные поля:
  - firstName
  - lastName
  - username
  - password
  - captchaToken
4. Backend проверяет, что captchaToken валиден (через captcha-сервис)
  - если невалиден → вернуть `400 CAPTCHA_REQUIRED`
5. Backend проверяет длину и формат пароля:
  - минимум 8 символов
  - 1 заглавная буква
  - 1 строчная буква
  - 1 цифра
  - 1 спецсимвол
6. Backend проверяет, существует ли пользователь с таким username
  - если существует → `409 USER_ALREADY_EXISTS`
7. Backend хэширует пароль
8. Backend создаёт запись пользователя в БД
9. Backend присваивает пользователю статус `ACTIVE`
10. Backend возвращает `201 Created` и сообщение об успехе