

Профориентационное тестирование в веб-приложении на основе профиля интересов и матрицы соответствия профессий

Андриенко Иван Сергеевич

Приамурский государственный университет имени Шолом-Алейхема

Студент

Аннотация

В работе описана реализация профориентационного тестирования в веб-приложении, где результаты пользователя формируются в виде профиля интересов по нескольким направлениям деятельности. Вопросы теста содержат варианты ответов с коэффициентами вклада, которые увеличивают соответствующие показатели профиля. Профессии представлены в формате JSON как профили соответствия, используемые для расчёта интегрального балла совпадения с профилем пользователя. На основе вычисленных баллов выполняется ранжирование профессий и формируется список рекомендаций (топ-5) с отображением относительных процентов.

Ключевые слова: профориентация; тестирование; профиль интересов; рекомендации; матрица соответствия; ранжирование; скоринг; JavaScript; JSON; Express; REST API.

Career guidance testing in a web application based on a profile of interests and a matrix of matching professions

Andrienko Ivan Sergeevich

Sholom-Aleichem Priamursky State University

Student

Abstract

The paper describes the implementation of career guidance testing in a web application, where the user's results are formed as a profile of interests in several areas of activity. The test questions contain answer options with contribution coefficients that increase the corresponding profile indicators. Professions are presented in JSON format as matching profiles used to calculate the integral match score with the user's profile. Based on the calculated scores, a ranking of professions is performed and a list of recommendations (top 5) is generated with relative percentages displayed.

Keywords: career guidance; testing; interest profile; recommendations; compliance matrix; ranking; scoring; JavaScript; JSON; Express; REST API.

1 Введение

1.1 Актуальность

Профориентационные сервисы востребованы для первичного выбора образовательной и профессиональной траектории. Практически значимы веб-решения, которые обеспечивают понятную интерпретацию результатов (профиль интересов), легко обновляют перечень профессий и не требуют сложной инфраструктуры. Подход на основе профиля интересов и матрицы соответствия позволяет формализовать выдачу рекомендаций и обеспечить воспроизводимость результатов.

1.2 Обзор исследований

Перед анализом собственной реализации важно рассмотреть научные работы, посвящённые использованию веб-приложений. В работе М. С. Берьянова рассматриваются подходы к интеграции слайдера в интерфейс React-приложения, включая организацию структуры компонентов и обработку пользовательских событий [1]. Исследование В. В. Бронской и А. Р. Ибатуллиной демонстрирует практическое применение React при создании пользовательских интерфейсов, где используются динамические элементы, включая визуальные карусели и интерактивные блоки [2]. В статье Д.А. Хайруллина и М.Р. Богданова рассматриваются особенности применения статической типизации в React-приложениях, включая методики типизации компонентов, хуков и состояния, а также влияние строгой типизации на надёжность пользовательских интерфейсов [3]. К.Р. Удальцов исследует вопросы обеспечения качества кода в командной разработке, предлагая формализацию шаблонов безопасных pull request-ов для проектов на React и TypeScript в рамках Git Flow [4]. Особое внимание созданию библиотек компонентов на основе TypeScript уделяется в работе Т.М. Воронко, где описываются принципы модульности, повторного использования кода и проектирования архитектуры компонентов для масштабируемых React-решений [5].

1.3 Цель исследования

Цель исследования – разработать и описать веб-реализацию профориентационного тестирования, формирующую профиль интересов пользователя по ответам, загружающую профили профессий через API и выполняющую расчёт соответствия с последующим ранжированием рекомендаций.

2 Материалы и методы

В качестве основы использовано веб-приложение на базе JavaScript и Node.js, где серверная часть реализована с применением фреймворка Express и предоставляет эндпоинт GET /api/professions для передачи справочника профессий на клиент. Данные о профессиях хранятся в файле professions.json и содержат идентификатор, название и профиль соответствия по направлениям интересов.

3 Результаты и обсуждения

Тест построен вокруг идеи «профиля интересов» пользователя: заранее задан набор направлений деятельности, по которым в процессе прохождения накапливаются баллы. Эти направления объявлены в `app.js` и используются как единая система координат для вопросов, ответов и профессиональных профилей. Вопросы также описаны в `app.js` как массив объектов: каждый вопрос содержит текст и варианты ответов, а у каждого варианта задан «вклад» в профиль интересов — то есть, какие направления и на сколько усиливаются при выборе данного ответа. Такой формат обеспечивает прямую связь между содержанием вопроса и вычислением результата, а также упрощает контроль корректности данных (рис. 1).

```
public > JS app.js > [state]
1  const TRAITS = [
2    "manual",
3    "technical",
4    "creative",
5    "analytical",
6    "social",
7    "leadership",
8    "outdoor",
9    "precision",
10 ];
11
12 const QUESTIONS = [
13   {
14     text: "Что тебе ближе в работе?",
15     options: [
16       { text: "Работать руками [ ] инструментами", scores: { manual: 2, technical: 1 } },
17       { text: "Анализировать данные и находить закономерности", scores: { analytical: 2, pre
18       { text: "Общаться и помогать людям", scores: { social: 2 } },
19       { text: "Придумывать и создавать новое", scores: { creative: 2 } },
20     ] },
21   },
22   {
23     text: "[ ] команде ты чаще...",
24     options: [
25       { text: "Берешь координацию и задаешь темп", scores: { leadership: 2, social: 1 } },
26       { text: "Делаешь сложную техническую часть", scores: { technical: 2, analytical: 1 } },
27       { text: "Отвечаешь за аккуратность и качество", scores: { precision: 2 } },
28       { text: "Придумываешь идеи и оформление", scores: { creative: 2 } },
29     ] },
30   },
31   {
32     text: "Какой формат работы комфортнее?",
33     options: [
34       { text: "[ ] в офисе или в разъездах", scores: { outdoor: 2 } },
```

Рис. 1. Структура направлений интересов и вопроса теста

Справочник профессий вынесен в отдельный файл `professions.json`. Каждая профессия хранит идентификатор, отображаемое название и профиль соответствия по тем же направлениям интересов. По смыслу профиль профессии фиксирует «акценты» данной деятельности: какие направления важнее, а какие вторичны. Благодаря этому профессии можно добавлять и корректировать независимо от клиентской логики теста, сохраняя общий формат сопоставления (рис. 2).

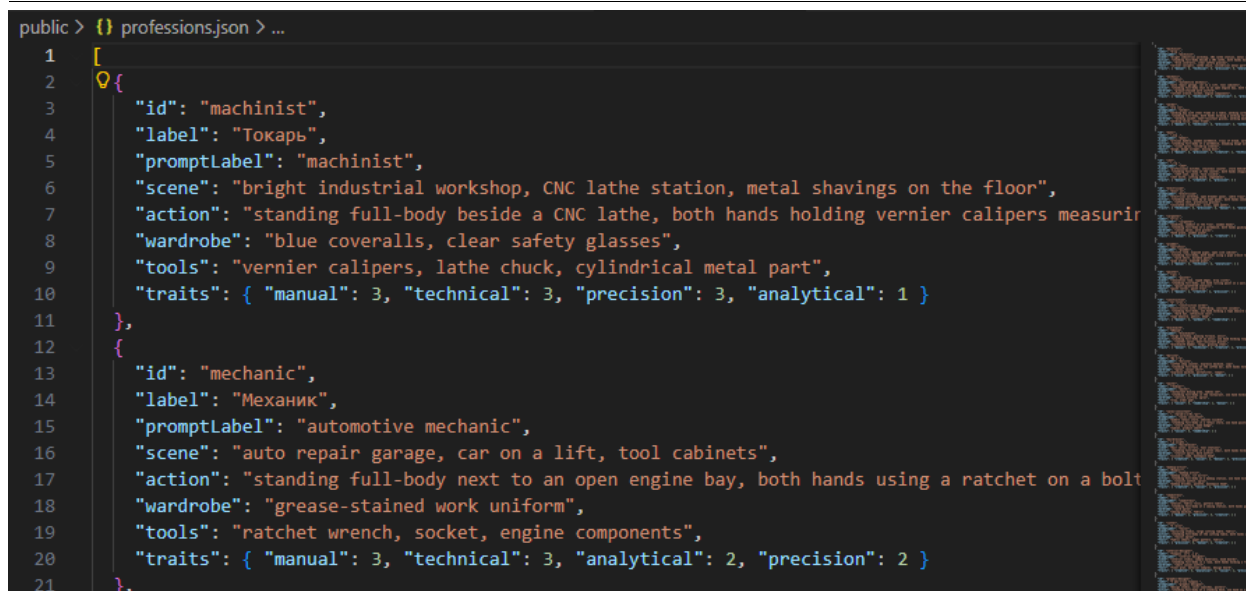


Рис. 2. Пример записи профессии в справочнике

Для передачи данных на фронтенд используется сервер на Express. На старте сервер читает `professions.json`, приводит записи к единому виду (в том числе учитывает отсутствие профиля, подставляя пустой объект) и отдаёт список профессий по API. Такой способ позволяет клиентской части всегда получать предсказуемую структуру данных и не дублировать хранение справочника в браузере.

При выборе варианта выполняется обновление профиля интересов: вклад выбранного ответа последовательно добавляется в соответствующие направления. После этого увеличивается счётчик текущего вопроса и выполняется либо переход к следующему вопросу, либо завершение теста, если вопросы закончились. В результате профиль интересов формируется пошагово и отражает совокупность выбранных предпочтений пользователя. Для формирования рекомендаций используется вычисление интегрального показателя соответствия. Для каждой профессии просматриваются все направления интересов: балл пользователя по направлению умножается на коэффициент значимости этого направления в профиле профессии, после чего произведения суммируются. Полученный показатель интерпретируется как степень совпадения профиля пользователя с профилем профессии. Далее все профессии сортируются по убыванию этого значения, формируя упорядоченный рейтинг (рис. 3).

```

public > JS app.js > finishTest
308 function applyAnswer(option) {
309   if (option?.scores) {
310     Object.entries(option.scores).forEach(([trait, value]) => {
311       state.traitScores[trait] = (state.traitScores[trait] || 0) + value;
312     });
313   }
314
315   state.testIndex += 1;
316   if (state.testIndex >= QUESTIONS.length) {
317     finishTest();
318   } else {
319     renderQuestion();
320   }
321 }
322
323 function finishTest() {
324   const scored = scoreProfessions();
325   state.recommended = scored.slice(0, 5);
326   state.selectedProfessionId = state.recommended[0]?.id || null;
327   renderResults();
328   showScreen("result");
329 }
330
331 function scoreProfessions() {
332   const scored = state.professions.map((profession) => {
333     const traits = profession.traits || {};
334     const score = TRAITS.reduce(
335       (sum, trait) => sum + (state.traitScores[trait] || 0) * (traits[trait] || 0),
336       0
337     );
338     return { ...profession, score };
339   });
340
341   scored.sort((a, b) => b.score - a.score);
342   return scored;
343 }
344
345 function renderResults() {
346   resultListEl.innerHTML = "";
347   if (state.recommended.length === 0) {
348     resultSelectedEl.textContent = "Не удалось подобрать профессии. Попробуй еще раз.";
349     return;
350   }
351
352   const maxScore = Math.max(...state.recommended.map((item) => item.score || 0), 1);
353

```

Рис. 3. Расчёт интегрального соответствия

После завершения теста из ранжированного списка выбираются первые пять профессий. Для повышения наглядности рассчитываются относительные проценты: значение каждой профессии приводится к шкале относительно максимума в топ-5, после чего отображается рядом с названием. Список результатов строится в виде интерактивных элементов, где выбранная рекомендация выделяется визуально, что делает интерфейс понятным при просмотре нескольких близких вариантов (рис. 4).

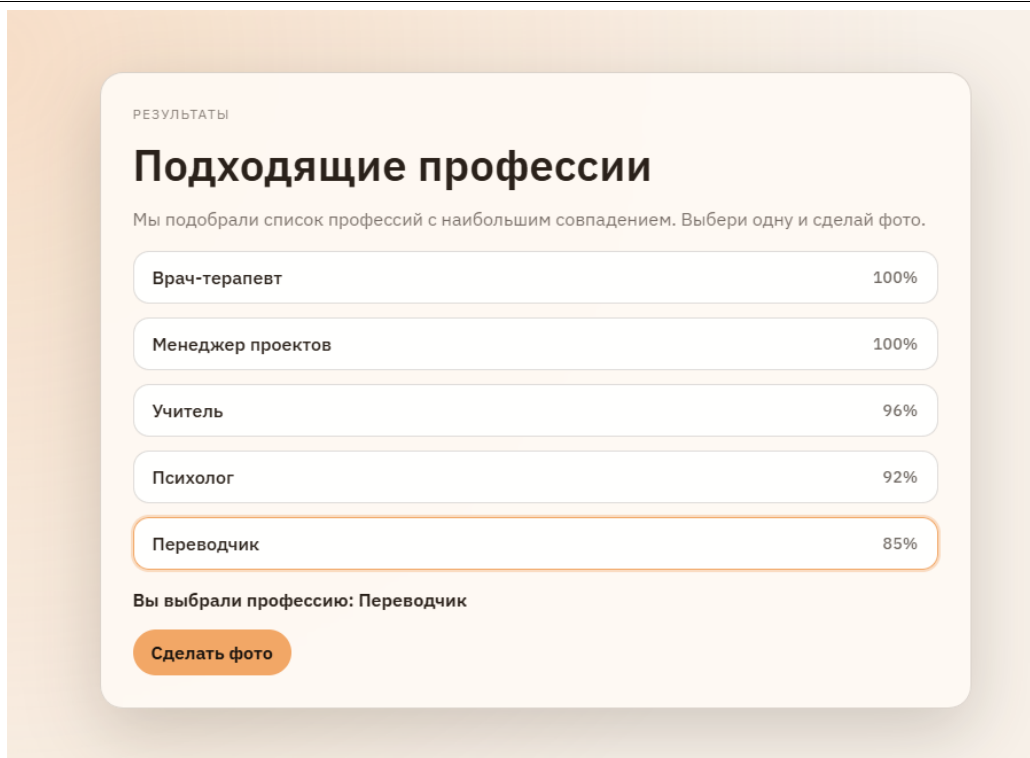


Рис. 4. Экран результата и относительные проценты

4 Выводы

В ходе работы реализовано профориентационное тестирование в формате веб-приложения, формирующее профиль интересов пользователя на основе ответов на вопросы и выполняющее ранжирование профессий по интегральному показателю соответствия. Структура данных теста задана в `app.js`, а справочник профессий вынесен в `professions.json`, что упрощает сопровождение и обновление перечня профессий без изменения основной логики. Получение справочника организовано через серверный API на Express, после чего клиентская часть обеспечивает динамическую отрисовку вопросов, отображение прогресса и формирование списка рекомендаций (топ-5) с относительной оценкой в процентах. Применённый подход обеспечивает прозрачность вычислений и воспроизводимость результатов за счёт единого формата профилей интересов и профилей профессий.

Библиографический список

1. Берьянов М. С. Внедрение компонента-слайдера (карусели) в React-приложение // Студенческий. 2023. № 1-1 (213). С. 32–34.
2. Бронская В. В., Ибатуллина А. Р. Применение JavaScript-библиотеки React для создания приложения интернет-магазина одежды // Известия высших учебных заведений. Технология текстильной промышленности. 2023. № 4 (406). С. 169–175.
3. Хайруллин Д.А., Богданов М.Р. Изучение использования TypeScript в React-приложениях, преимущества статической типизации, подходы к типизации компонентов, хуков и состояния // Международный журнал

- гуманитарных и естественных наук. 2024. № 12-3 (99). С. 233–238.
4. Удальцов К.Р. Формализация шаблонов безопасных pull request-ов для React-приложений с TypeScript в командах с Git Flow // Актуальные проблемы социально-гуманитарного и научно-технического знания. 2025. № 4 (44). С. 35–36.
 5. Воронко Т.М. Создание React-библиотеки с использованием TypeScript // Электронные системы и технологии. Материалы 58-й научной конференции аспирантов, магистрантов и студентов БГУИР. Минск, 2022. С. 200–202.