Вариант 15: Платформа для менторства

Описание: Система соединяет менторов (наставников) и менти (учеников) по различным навыкам (программирование, дизайн, карьера). Пользователи создают профили с навыками, находят пары и проводят сессии, которые можно оценивать.

Анализ предметной области и ключевые сущности

Предметная область — платформа, которая связывает людей в роли менторов и менти по конкретным навыкам и поддерживает процесс менторства от поиска пары до проведения и оценки сессий. Здесь важно, что «менторство» — это не одна таблица, а цепочка связанных событий: человек описывает себя и свои навыки; ученик формулирует запрос на обучение; менторы откликаются; по результату создаётся пара; пара проводит одну или несколько встреч; по встречам оставляют оценки. Если модель не фиксирует хотя бы одно звено этой цепочки, часть сценария становится «невидимой» для базы данных.

Отсюда выделяются ключевые сущности:

1. Пользователь — центральный объект. Он может быть ментором, учеником или совмещать роли. У пользователя есть профиль, и он участвует в дальнейших процессах (создает запросы, откликается, проводит и оценивает сессии).
2. Навык — справочная сущность, потому что навыки повторяются у множества людей и используются как основание для подбора пар. Навыки должны быть единым словарём, иначе невозможно корректно искать совпадения (например, «Python» vs «python»).
3. Навык пользователя — самостоятельная сущность-связь. В предметной области недостаточно просто знать, что пользователь «умеет Python»: нужно знать уровень владения и стаж. Эти свойства относятся не к навыку вообще и не к пользователю вообще, а к их сочетанию. Поэтому связь users—skills становится отдельной таблицей.
4. Запрос на менторство — сущность намерения ученика. Описание говорит «пользователи находят пары», значит система должна хранить факт поиска: кто ищет, по какому навыку, с какой целью. Без этой сущности мы не поймём, откуда взялась пара и какие запросы остались без ответа.

5. Отклик ментора (заявка) — сущность взаимодействия. Подбор пары предполагает, что менторы видят запросы и предлагают помощь. Отклик имеет собственный статус (ожидает, принят, отклонен) и содержание (сообщение), поэтому это отдельная таблица, а не просто атрибут матча.
6. Матч (пара ментор–ученик) — сущность устойчивых отношений. Это результат подбора, который живет во времени, имеет статус (активна, завершена, отменена) и может порождать цепочку сессий. Пара обязательно привязана к навыку, иначе нельзя понять, по какому направлению идёт наставничество.
7. Сессия — сущность события внутри пары. Встреч может быть много, они планируются на дату/время, имеют длительность, статус и содержательные поля (повестка, заметки). Это классическая связь «1 матч → N сессий».
8. Оценка сессии — сущность обратной связи. В описании прямо сказано, что сессии можно оценивать. Оценка зависит от конкретной сессии и конкретного пользователя, то есть это отдельная таблица с уникальностью «один пользователь — одна оценка одной сессии».

Атрибуты сущностей и связи между ними

Пользователи (users)

У пользователя есть идентификатор id, имя и email как основные поля профиля. Email уникален — это естественный ключ для входа и коммуникаций. Биография и аватар — необязательные элементы профиля, поэтому допускают NULL. Ролевые флаги is_mentor и is_mentee отражают, как пользователь участвует в системе: наставник, ученик или оба сразу. Временные поля created_at и updated_at фиксируют момент регистрации и обновления профиля — это важно для аналитики и поддержки жизненного цикла учетной записи.

Связи пользователей проявляются в других таблицах: пользователь задаёт свои навыки (user_skills), создает запросы (mentorship_requests), может быть тем, кто откликается (mentorship_applications), входит в пары (mentorship_matches), участвует в сессиях и оценивает их (session_ratings).

Навыки (skills)

Навык описан через id и уникальное название name. Уникальность нужна, чтобы один и тот же навык существовал в системе в единственном

экземпляре. Дополнительное описание (description) помогает расширить справочник, но не обязательно для каждого навыка. created_at позволяет понимать, когда навык добавлен в каталог.

Навык связан с пользователями через user_skills, а также используется как критерий в запросах (mentorship_requests) и матчах (mentorship_matches).

Навыки пользователей (user_skills)

Таблица реализует связь многие-ко-многим: один пользователь может иметь много навыков, и один навык может быть у многих пользователей. Поэтому ключ составной (user_id, skill_id). Атрибуты proficiency, years_experience и is_primary описывают уровень и опыт именно конкретного пользователя в этом навыке. added_at фиксирует момент добавления навыка в профиль.

Связи: user_skills.user_id → users.id, user_skills.skill_id → skills.id.

Запросы на менторство (mentorship_requests)

Это сущность, принадлежащая менти. Основные атрибуты: кто ищет (mentee_id), по какому навыку (skill_id) и зачем (goal). Возможный желаемый уровень ментора (desired_proficiency) — не строго обязательный, поэтому может быть NULL. Запрос имеет статус: он может быть открытым, на рассмотрении, закрытым или уже приведшим к матчу. Временные поля нужны для отслеживания актуальности запросов.

Связи: mentorship_requests.mentee_id → users.id, mentorship_requests.skill_id → skills.id. Запрос далее порождает отклики.

Отклики менторов (mentorship_applications)

Отклик связывает конкретного ментора с конкретным запросом. Поэтому хранит request_id и mentor_id. Сообщение ментора (message) — свободный текст. Статус отклика отражает этап обработки (ожидание, принят, отклонен, отозван). Уникальность (request_id, mentor_id) не даёт одному ментору «заспамить» запрос десятками одинаковых заявок.

Связи: mentorship_applications.request_id → mentorship_requests.id, mentorship_applications.mentor_id → users.id.

Матчи/пары (mentorship_matches)

Пара фиксирует устойчивую связь ментора и менти по конкретному навыку. Для этого нужны mentor_id, mentee_id, skill_id. Поле request_id

делает происхождение пары прозрачным: матч может быть создан из запроса, но теоретически может быть создан и администратором/алгоритмом — поэтому ссылка не обязательна. Статус матча отражает жизненный цикл отношений. Заметки сторон (mentor_note, mentee_note) — рабочие поля, которые относятся к самой паре, а не к отдельной сессии. Даты started_at/ended_at/cancelled_at дают возможность анализировать длительность менторства и причины завершения. Ограничение mentor_id <> mentee_id исключает бессмысленные самоматчи. Уникальность (mentor_id, mentee_id, skill_id) означает: одна пара по одному навыку существует в системе в единственном экземпляре, а ее состояние меняется через status.

Связи: оба участника — ссылки на users, навык — ссылка на skills, запрос — ссылка на mentorship_requests. Матч порождает сессии.

Сессии (sessions)

Сессия принадлежит матчу (match_id) и описывает планируемую/проведённую встречу. Время встречи — scheduled_at, длительность — duration_minutes. Статус сессии нужен, потому что встреча может быть отменена, завершена или не состояться по неявке. meeting_link отражает реальность онлайн-встреч. agenda и notes относятся к содержанию данной сессии. Поля отмены (cancelled_at, cancel_reason) полезны для аналитики. Уникальность (match_id, scheduled_at) не позволяет задвоить один и тот же слот внутри пары.

Связь: sessions.match_id → mentorship_matches.id. Сессия далее может иметь оценки.

Оценки сессий (session_ratings)

Оценка относится к конкретной сессии (session_id) и конкретному пользователю, который ее оставил (rater_id). Рейтинг числовой, ограниченный шкалой 1–5. Комментарий — необязательный текст. Уникальность (session_id, rater_id) гарантирует, что один участник не «переголосует» свою же оценку.

Связи: session_ratings.session_id → sessions.id, session_ratings.rater_id → users.id.

Обоснование типов данных и структуры таблиц

Во всех таблицах используется BIGSERIAL как первичный ключ. Это автоинкрементный тип с большим диапазоном, что подходит для

потенциально крупной платформы. В качестве внешних ключей выбран BIGINT, чтобы типы совпадали с родительскими PK и не было проблем при соединениях и проверках целостности.

Строковые поля профиля, целей, заметок и комментариев сделаны типом TEXT, потому что заранее неизвестны максимальные длины: имя пользователя может быть длинным, биография или цель — тоже. Использование фиксированного VARCHAR(n) в такой задаче не даёт плюсов, но может создать искусственные ограничения, поэтому TEXT логичнее.

Временные поля заданы как TIMESTAMP WITH TIME ZONE. Это важно, потому что сессии и пользователи могут находиться в разных часовых поясах. Без часового пояса время встреч хранится неоднозначно и может интерпретироваться неверно.

Для шкал используются небольшие числовые типы и ограничения:

- уровень навыка proficiency — SMALLINT с проверкой 1–10; диапазон маленький, хранить большие типы нет смысла;

- опыт years_experience — NUMERIC(4,1); дробная часть нужна, чтобы фиксировать, например, 0.5 года, а формат с 1 знаком после запятой достаточно точный для предметной области;

- рейтинг rating — SMALLINT 1–5, потому что оценка дискретна и мала.

Статусы реализованы через TEXT + CHECK. CHECK удерживает значения в допустимом наборе, не выходя за рамки табличных средств.

Структура таблиц подчинена правилу нормализации: никакая таблица не хранит «повторяющиеся группы» или атрибуты, которые относятся к другой сущности. Например, уровни и стаж навыка нельзя поместить ни в users, ни в skills — поэтому есть user_skills. Аналогично, отклики нельзя хранить как массив в запросе, а сессии нельзя хранить внутри матча — это раздувает строки и ломает связь 1:N, поэтому используются отдельные таблицы с внешними ключами.