Ризики проєкту сервісу підбору розваг

Перед тим як запустити щось на ринок чи почати роботу над великим проектом треба провести аналіз ризиків, яким він може підлягати. Мета такого аналізу не просто поставити галочку, а заздалегідь подумати, що може піти не так і дізнатися про способи контролю даних ризиків.

Проєкт, який розглядається, це сервіс під назвою умовно FunPlan. Це мобільний застосунок з веб версією, який допомагає людям планувати відпочинок. Користувач може задати свій бюджет, настрій, локацію, сказати які речі у нього вже є для дозвілля, і отримати список ідей що можна зробити вільним вечором або у вихідний. Також сервіс має підтримувати спільне планування для компанії друзів або пари, щоб можна було узгодити варіанти між кількома людьми.

Технічно це досить складний проєкт. Є серверна частина з базою даних, є клієнтські застосунки, є інтеграція з картами і зовнішніми джерелами подій. Окрема важлива частина це рекомендаційна система, яка намагається передбачити що саме буде цікаво конкретному користувачу. Через це в проєкті з’являється багато ризиків, які пов’язані і з технологіями, і з людьми, і з ринком.

Один з головних ризиків це неточні рекомендації ШІ системи. Якщо модель навчена на невеликій або погано підібраній вибірці даних, або якщо не враховані реальні вподобання користувачів, сервіс буде пропонувати або занадто банальні, або просто невдалі варіанти. Користувач кілька разів натисне на такі пропозиції, розчарується і просто видалить застосунок. Тут ризик досить ймовірний, особливо на старті, коли ще мало користувачів і мало історії взаємодії. Вплив такого ризику також високий, бо фактично саме рекомендації створюють основну цінність сервісу. Зменшити його можна поступовим запуском, тестуванням на невеликих групах людей, ручною перевіркою частини рекомендацій та введенням простіших сценаріїв, коли система ще мало знає про людину.

Другий важливий ризик це не актуальні або неповні дані про події та розваги. FunPlan сильно залежить від зовнішніх джерел інформації, наприклад афіш, агрегаторів подій, сайтів окремих закладів. Якщо партнери не оновлюють дані або змінюють формат, сервіс може показувати заходи, які вже скасовані, або ціни, що давно виросли. Користувач приїде на подію, а її просто немає, логічно що після цього довіра до застосунку падає майже до нуля. Ймовірність такого ризику досить велика, бо реальний світ дуже динамічний. Тут бажано мати автоматичні перевірки актуальності, домовленості з партнерами щодо оновлення інформації та можливість позначати дані як неперевірені, щоб користувач бачив де система не впевнена. Також хорошим варіантом вирішення проблеми з оновленням даних буде створити окремий застосунок для бізнесів, де вони зможуть швидко і легко оновлювати дані. Такий підхід дозволить не лише забезпечити зменшення неточних даних, а і спростить їх збір бо тоді не треба буде писати велику кількість різних сервісів збору даних через різноманітну природу джерел інформації.

Є також ризик пов’язаний зі зміною політики або вартості зовнішніх API, наприклад сервісу карт. На початку команда може розраховувати на певні безкоштовні ліміти, а потім кількість користувачів виросте, і обмеження раптово стануть критичними. Або постачальник змінить тарифи, і використання карт стане занадто дорогим. Це може сильно вдарити по бюджету або змусити терміново переробляти значну частину функціоналу, що вплине на строки і якість. Щоб знизити цей ризик, треба стежити за лімітами, продумати кешування запитів, а також завчасно передбачити можливість переходу на інший картографічний сервіс.

Окрема група ризиків пов’язана з продуктивністю та масштабованістю системи. Уявімо що після якоїсь рекламної кампанії кількість активних користувачів різко зростає, а серверна частина і база даних не розраховані на таке навантаження. Застосунок починає гальмувати, запити виконуються дуже довго або взагалі повертають помилку. Для користувача це виглядає як повністю неробочий сервіс, навіть якщо проблема лише в технічних деталях. Частота таких ситуацій

залежить від того чи команда проводила навантажувальне тестування і закладала запас по ресурсах. Якщо ні, то ймовірність стає досить великою. Тут потрібна продумана архітектура, використання хмарних рішень і регулярні тести роботи під навантаженням.

Ще один дуже серйозний ризик це порушення безпеки та конфіденційності персональних даних. FunPlan працює з логінами, паролями, контактами, геолокацією та історією переміщень користувача. Якщо в системі будуть слабкі місця, можливий витік цих даних. Навіть якщо ймовірність такої події не надто велика, наслідки будуть критичними. Це і юридичні проблеми, і штрафи, і повна втрата довіри до бренду. Для зниження цього ризику треба від самого початку закладати безпечну архітектуру, використовувати шифрування, обмежувати доступ до бази даних, оновлювати компоненти і проводити перевірки безпеки.

Крім технічних речей, є й організаційні ризики. Один з них це нестача або невдале поєднання кадрових ресурсів. В реальності проєкт зазвичай робить невелика команда, де одна людина може бути і розробником, і тестувальником, і частково виконувати функції менеджера. Якщо при цьому немає фахівця з машинного навчання чи UX дизайну, якість окремих частин продукту неминуче впаде. Перевантажені працівники роблять більше помилок, строки зсуваються, а частина важливого функціоналу так і не доводиться до потрібного рівня. Щоб хоча б частково зменшити цей ризик, важливо чітко розподілити ролі, не змішувати критично несумісні задачі в одній позиції та за можливості залучати зовнішніх консультантів на вузькі теми.

До ризиків, що пов’язані з користувачами, можна віднести невідповідність UX очікуванням цільової аудиторії. Сервіс орієнтований на молодих людей, студентів, туристів, невеликі компанії друзів. Якщо інтерфейс буде перевантажений, незрозумілий або просто нудний, користувачам буде важко розібратися як додавати власні ідеї, як ділитися планами з друзями, як фільтрувати результати. Часто це трапляється тоді, коли дизайн робиться без реальних інтерв’ю з

майбутніми користувачами і без тестування прототипів. Простий спосіб знизити ризик тут це кілька циклів прототипування, покази реальним людям і швидке внесення змін за їхніми відгуками.

Нарешті, треба сказати і про ринковий та бізнес ризик. Навіть якщо технічно застосунок працює добре, ніхто не гарантує, що люди захочуть ним активно користуватися або платити за додаткові можливості. На ринку вже є карти, афіші, різні агрегатори закладів. Великі гравці можуть швидко скопіювати основну ідею і вбудувати її у свій екосистему. Тоді проєкту буде складно знайти своє місце і заробити достатньо, щоб окупити розробку. Тут важливо правильно сформулювати унікальну цінність сервісу, наприклад зробити акцент на поєднанні персональних рекомендацій з колективним плануванням та підтримкою особистого інвентарю користувача. Також можна поступово тестувати різні моделі монетизації і не вкладати всі ресурси тільки в один варіант.

Як висновок, можна сказати що проєкт FunPlan має багато різних ризиків, які стосуються технологій, людей, зовнішніх сервісів і самого ринку. Частина з них досить ймовірна, частина менш, але наслідки в багатьох випадках будуть відчутними. Важливо оцінити вплив кожної проблеми для подальшого керування ризиками.

Трудові ресурси проєкту сервісу підбору розваг

Готовий продукт не означає успіх. Навколо нього є замовник або власник продукту, інвестор, реальні користувачі, партнери заклади та онлайн сервіси, які дають дані про розваги. Усередині самого проєкту головну роль відіграє керівник проєкту, команда розробників, фахівці з аналітики, безпеки, дизайну та підтримки користувачів. Всі ці люди складають оточення та учасників проєкту, і від того наскільки грамотно розподілені їхні обов’язки залежить успіх сервісу.

Ключова фігура всередині проєкту - це керівник проєкту. Йому потрібно вміти нормально спілкуватися з людьми, планувати строки, рахувати бюджет, домовлятися з партнерами і командою, розв'язувати конфлікти. По суті він є зв'язком між бізнесом і технічною частиною. У випадку FunPlan менеджер проєкту тримає на контролі одразу кілька напрямків: розробку застосунку, роботу з даними для рекомендацій, співпрацю з партнерами та запуск маркетингу. Це вимагає не тільки технічного розуміння, а й м'яких навичок, бо команда може бути зібрана з людей різного досвіду і навіть з різних компаній.

Якщо дивитися на склад команди, то для цього проєкту мінімально потрібні такі ролі. По перше, бекенд-розробник, який відповідає за серверну частину, базу даних, інтеграції з зовнішніми API. По друге, фронтенд або мобільний розробник, який робить інтерфейс для користувача, бажано з досвідом роботи з геолокацією і картами. По третє, фахівець з машинного навчання або хоча б аналітик даних, який налаштовує рекомендаційний алгоритм. Також важливі UX/UI-дизайнер, тестувальник, адміністратор інфраструктури (DevOps) і людина, що працює з партнерами та контентом подій.

Ці ролі напряму пов'язані з ризиками, які були описані в попередньому есе. Наприклад, щоб зменшити ризик неточних рекомендацій, потрібен фахівець, який розуміє як працюють моделі, вміє збирати та чистити дані, тестувати якість рекомендацій. Щоб уникнути неактуальної інформації про події, потрібна людина, яка відповідає за взаємодію з закладами, агрегаторами та перевірку даних. Ризик перевантаження сервера зменшується за рахунок наявності спеціаліста, який розбирається в масштабуванні, хмарній інфраструктурі і вміє планувати навантаження. Ризик витоку даних потребує або досвідченого бекенд-розробника з розумінням безпеки, або залучення окремого консультанта.

Управління людськими ресурсами в цьому проєкті означає не тільки набрати потрібних людей, але й визначити їхню кваліфікацію, скільки часу вони потрібні, хто кому підпорядковується, як вирішувати конфлікти і як мотивувати команду. Спочатку формують список ролей і

базові вимоги до кожної з них. Наприклад, для бекенд-розробника – досвід з веб-фреймворком, роботою з базою даних і авторизацією. Для ML-фахівця – досвід з рекомендаційними моделями, аналітикою подій користувача. Для UX-дизайнера – портфоліо застосунків, в яких є прості і зрозумілі сценарії.

Далі треба вирішити, які ролі можуть поєднуватися однією людиною, а які краще розділити. Через обмежений бюджет у невеликому проєкті часто одна людина виконує дві або навіть три ролі. Наприклад, системний архітектор може паралельно бути старшим розробником, а людина яка відповідає за контент, може частково виконувати функцію підтримки користувачів. Але є поєднання, яких бажано уникати. Було б погано якщо розробник сам собі тестувальник і менеджер з якості, бо тоді контроль стає формальним. Так само не варто робити керівника проєкту одночасно головним програмістом, бо він або буде кодити і не встигати керувати людьми, або навпаки.

Окрема тема – це формування команди в часі. Спочатку є етап, коли людей тільки підбирають і знайомлять з ідеєю, ролі ще не до кінця зрозумілі, усі трохи обережні. Потім настає період “притирання”, коли починаються перші конфлікти через різні підходи до роботи, різні очікування від продукту. На цьому етапі менеджер проєкту має більше говорити з людьми, пояснювати рішення, м’яко гасити напругу. Далі команда входить у фазу нормальної роботи, коли вже є певні правила і кожен знає свою ділянку, зростає довіра. І тільки після цього можна говорити про по-справжньому стабільну, “боєздатну” команду, яка працює майже автономно, а керівник більше координує, ніж жорстко керує.

Щоб команда FunPlan не розсипалась після перших проблем, треба продумати базові речі мотивації. Для студентського або навчального варіанту проєкту це може бути можливість зробити цікавий кейс у портфоліо, участь у конкурсах, відчуття, що робиш щось корисне для реальних людей. Для комерційної версії додається ще фінансова мотивація, бонуси за досягнуті цілі, частка в прибутку або опціони.

Важливо, щоб правила були понятні, а система премій не виглядала випадковою, інакше вона тільки збільшить конфлікти.

Також управління людськими ресурсами включає навчання і розвиток. У цьому проєкті є теми, де команда може не мати достатньої кваліфікації, наприклад машинне навчання або інформаційна безпека. Частину прогалин можна закрити короткими курсами, внутрішніми міні-лекціями або наставництвом більш досвідченої людини. Інколи дешевше і швидше навчити свого розробника новій технології, ніж шукати ще одного спеціаліста на пів ставки, який потім усе одно буде інтегруватися в команду.

Не менш важливим є план вивільнення ресурсів. Деякі ролі потрібні тільки на певних етапах. Наприклад, UX-дизайнер найбільш завантажений на стадії прототипування та перших релізів, потім його участь може бути меншою. Фахівець з налаштування інфраструктури потрібен при запуску і під час масштабування, а постійно на повний день може і не знадобитися. Якщо подумати про це заздалегідь, можна зменшити витрати і уникнути ситуації, коли люди сидять без роботи і втрачають мотивацію.

У підсумку можна сказати, що для проєкту FunPlan трудові ресурси – це не просто список посад. Це продумана система ролей, обов’язків, компетенцій, зв’язків і мотивації. Від того, наскільки розумно буде сформована команда, залежить реалізація майже всіх попередніх ризиків: і точність рекомендацій, і актуальність даних, і продуктивність сервера, і безпека, і якість користувацького досвіду. Якщо вдало поєднати теорію управління людськими ресурсами з реальними потребами такого ІТ-проєкту, то шанси, що сервіс підбору розваг стане живим і корисним продуктом, помітно зростають.