00:00:03,920 - 00:00:16,520 - Вот. «Кем ты работал до этого?» Вот. «Про свой опыт системного аналитика. Какие решал интересные задачи?» Вот. Ну и в целом, всё, что хочешь интересного про работу, про свою текучку рассказать, можешь рассказать.

00:00:17,700 - 00:00:22,960 - Да, хорошо. Ну, первоначально это мой антикорр…

00:00:23,760 - 00:00:43,400 - Опыт работы системным аналитиком, он как бы мой единственный, ну, несмотря на мелкие подработки, вот. Работаю системным аналитиком я уже около трёх лет. Соответственно, после окончания института окончил я по направлению прикладная математика. Я хотел пойти работать на

00:00:43,400 - 00:00:45,700 - Big Data Science Analytica

00:00:45,760 - 00:00:50,960 - Вот. И, соответственно, искал подходящие вакансии.

00:00:51,100 - 00:01:08,840 - Нашёл одну вакансию, которая меня заинтересовала. Ну, она называлась «Системный аналитик». Более того, была сфера блокчейна. О блокчейне, как и о системном анализе, я ничего не знал. И, соответственно, ну, решил попробовать. И…

00:01:09,540 - 00:01:26,780 - Так получилось мое начало карьеры. Соответственно, поначалу я работал пресейлом, делал оценки. Далее развивался как системный аналитик. Соответственно, работал с...

00:01:27,460 - 00:01:34,420 - На аутсорсе. У нас была компания на аутсорсе. Работал с многими клиентами.

00:01:34,920 - 00:01:48,360 - Так же с известными в сфере блокчейна. Phantom Blockchain, там Solana, вот. Мы выполняли для них задачи. Ну, соответственно, там был мой первоначальный опыт. После этого я

00:01:48,660 - 00:01:58,300 - перешёл в другую компанию, начал работать с системой библиотечной деятельности. Эта вакансия, ну и компания, она была

00:02:00,140 - 00:02:07,680 - такая, ну, мы работали по некоему такому государственному регламенту, где использовался ГОСТ, где были

00:02:08,100 - 00:02:09,900 - Технописатели

00:02:10,020 - 00:02:27,940 - Соответственно, там было несколько этапов согласования, требований, которые проходили через несколько ступенек. Вот. И за весь вот этот опыт работы я работал больше с задачами на фронт. Далее я

00:02:28,000 - 00:02:38,180 - перешёл в компанию по разработке социальной сети, но этот проект тоже был, ну, как проектная работа, и, соответственно,

00:02:38,560 - 00:02:57,820 - На ней я тоже выполнял такие более фронтовые задачи. На текущем месте работы я уже работаю около года, и, соответственно, я являюсь, можно сказать, старшим системным аналитиком в компании. И мы работаем над продуктом. Здесь в основном задачи на бэк.

00:02:59,360 - 00:03:15,900 - Описание интеграций, ну и все основы утекающей задачи. И, соответственно, у меня в штате, помогая новичкам, аналитикам развиваться. Вот. Круто, спасибо.

00:03:17,160 - 00:03:25,360 - Уточню ещё один момент. Расскажи немножко про состав команды, то есть как ты привык работать, кто обычно в команде присутствует.

00:03:25,980 - 00:03:37,020 - Ну, вообще, на всех местах работы у нас присутствует project-менеджер, который управляет проектом. Также есть дизайнер, соответственно, системной аналитик.

00:03:37,280 - 00:03:53,400 - Особо выделения на бизнес-аналитика нигде нету, поэтому роль бизнес- и системного аналитика выполняет один человек. Также есть разработчики Backfront, соответственно, DevOps.

00:03:54,140 - 00:03:59,420 - Ну и, наверное, всех, которых успел вспомнить, всех называл.

00:04:01,520 - 00:04:03,340 - Окей, давай тогда…

00:04:04,080 - 00:04:14,600 - Пойдём по темам уже, да, по собеседованию. Сегодня поговорим про требования в целом, вот. Спрашиваю про архитектуру немножко.

00:04:15,260 - 00:04:31,140 - Обсудим нотации моделирования процессов. Конечно же, мы поговорим про интеграции, про технологии. И в конце немножко затронем SQL-чик. Может быть, даже в онлайне решишь нам пример.

00:04:31,460 - 00:04:36,360 - Вот. Ну, тогда давай, первый вопрос по требованиям.

00:04:36,740 - 00:04:41,300 - Начнём с простого. Какую ты себе представляешь классификацию?

00:04:41,900 - 00:04:47,280 - Ну, функциональные, нефункциональные требования. Также есть бизнес-требования.

00:04:50,160 - 00:05:03,160 - А нефункциональные требования — это что такое? Это требования, которые в целом описывают некоторые ограничения на систему. На систему, на приложение в целом. То есть…

00:05:03,920 - 00:05:20,300 - Ну и не только. Они также могут описывать и какое-то руководство по пользованию, какую-то мануалку, да, для конечных пользователей, которые будут, соответственно, взаимодействовать как с системой, так и с каким-то конкретным решением.

00:05:20,580 - 00:05:38,840 - А вот требования к UX и к UI — это к каким можно отнести? Требования к… Ну, не функциональные требования. Угу. С точки зрения формализации, да, функциональных требований, выделяют там такие, ну, понятия, как «user story» и «use case»?

00:05:39,280 - 00:05:47,140 - Можешь рассказать, в чём между ними разница? Или, может быть, пример привести одного и другого? Да, ну, S-кейс — это

00:05:47,340 - 00:06:00,260 - Ну, вот есть, например, в UML с кейс-диаграмм. Это некая сводка сценариев, которая описывает верхнеуровневую работу определённой функциональности в системе. Вот.

00:06:00,300 - 00:06:14,040 - То есть, есть какая-то функциональность, соответственно, есть некие сценарии, как конечный пользователь взаимодействует с этими сценариями, и она описывает верхнего уровня. Вот. А User Story — это больше как

00:06:14,340 - 00:06:17,960 - Детальное описание самого юзкейса.

00:06:18,240 - 00:06:21,220 - Ты… Это, получается, идет в формате

00:06:21,280 - 00:06:40,800 - Ну, вообще, есть много фреймворков по работе с User Story. Там, Геркин, Кюкомба и ещё всякие, вот. Она описывает некую мотивацию определённого участника, ну, определённой роли, почему именно необходимо реализовать ту или иную фичу в системе, в решении.

00:06:40,840 - 00:06:44,980 - Можешь, к примеру, вот юзер-старик какой-нибудь привести?

00:06:45,680 - 00:06:47,980 - Как пользователя я хочу

00:06:50,480 - 00:06:58,820 - Я хочу иметь функциональную возможность выгружать отчет в Excel, чтобы сэкономить себе время.

00:07:02,660 - 00:07:06,720 - Ты сказал, что «Изерстаря» — это такое более

00:07:07,300 - 00:07:14,540 - Низкоуровневый, да. Ну, скажем так, инструмент, да, для описания требований, нежели юзкейс.

00:07:15,000 - 00:07:27,720 - А вот если нам нужно, например, описать какие-то альтернативные поведения, да, системы на какие-то действия пользователя.

00:07:27,760 - 00:07:31,530 - То что нам лучше, использовать юзкейсы или юзерстари?

00:07:31,530 - 00:07:40,720 - Ну вот, в частности, я говорил про диаграмму из кейсов, потому что она, ну, для такого поверхностного понимания очень хорошо как раз иллюстрирует функцию.

00:07:40,720 - 00:07:53,580 - Конечно же, можно use case, ну, дополнить, точнее, вот эти частные сценарии, да, которые мы изобразили на диаграмме, можно расписать в виде сценария, где как раз мы будем указывать

00:07:53,880 - 00:08:05,640 - Описание этого сценария, где будем указывать акторов, которые участвуют в процессе, где мы будем указывать, там, какие-то некие полномочия, некоторые ограничения, условия, триггер.

00:08:05,820 - 00:08:21,520 - Основной процесс, как раз альтернативные сценарии, ну и какой-то конечный результат данного сценария, вот. Соответственно, для откажания use case. А вот на практике в своих командах вы обычно юзер-старей используете или нет?

00:08:22,220 - 00:08:33,900 - На данном… Ну, вообще, в течение времени я использовал юзер-стори описание. Соответственно, к юзер-стори шли критерии приёмки. Критерии приёмки — это всё

00:08:34,260 - 00:08:46,500 - трассируется с User Story, вот. Но как такового описания сейчас на текущем месте работы не требуется, вот. Use Case тоже описывали.

00:08:46,800 - 00:08:49,320 - Ну и, в частности, я описываю

00:08:49,360 - 00:09:08,320 - в текстовом формате, также с указанием всех атрибутов вышеперечисленных. И, соответственно, это служит основой как не только для разработчиков, которые будут, соответственно, разрабатывать функциональность, но также и для тестировщиков, которые будут писать тест-кейсы, PMI и так далее.

00:09:09,040 - 00:09:10,340 - Окей.

00:09:11,600 - 00:09:26,680 - Следующий вопрос такой. Допустим, ты пришёл на проект в качестве аналитика. Для тебя, как для аналитика, какие бы ты выделил источники требований? То есть куда бы ты ходил, что бы смотрел, чтобы эти требования собирать?

00:09:27,340 - 00:09:47,660 - Ну, изначально я бы пообщался с проект-менеджером для того, чтобы утвердить некий круг стейкхолдеров, к которым мне необходимо как раз искать каналы связи для того, чтобы собирать эти требования, вот. Ну и, соответственно, там в зависимости от стейкхолдеров уже и

00:09:48,640 - 00:09:55,120 - Как говорится, ну, подбирается канал связи для коммуникации, для сбора требований, и по ним дальше ты взаимодействуешь.

00:09:56,800 - 00:10:12,740 - Окей. Ну а вот помимо стейкхолдеров… Ну, условно говоря, стейкхолдеры — это люди, да? Вот человек может быть источником требований тебе что-то сказать, там, объяснить и так далее. А вот помимо людей, да, и стейкхолдеров, что ещё может быть источником для тебя? Ну, а также текущее, базовое.

00:10:13,300 - 00:10:33,300 - Текущая база данных, ну, условно, там, вики какая-нибудь, документация проекта, которую ведёт, там, текущий аналитик, вот, или проект-менеджер. Также это какие-то видеозаписи, там, всяких митингов, грумингов, ретро и так далее. Ну, здесь на самом деле источником требования выступает множество.

00:10:33,840 - 00:10:47,820 - Сущности, можно так сказать. А вот на практике чем чаще всего работал? И помимо того, что там общался со стейкхолдерами, какие-то ещё источники использовал в работе? Ну да, на самом деле в процессе работы

00:10:48,520 - 00:11:01,320 - Как говорится, очень много, ну очень много источников, к которым приходится обращаться. Это не только какие-то носители информации, в том числе люди, да.

00:11:02,120 - 00:11:12,600 - команды, которые со мной. Ну, а также дополнительное — это какой-то бенчмаркинг, какое-то изучение, анализ рынка. Безусловно, работаю с

00:11:12,980 - 00:11:27,040 - документаций с вики куда как раз вся история знаний сохраняется вот где описаны все требования описаны система изыск вот ну соответственно также

00:11:27,640 - 00:11:37,080 - Ну, мы ведем записи всех видеоконференций, ну, и текстовый протокол, вот, и, соответственно, по текстовым протоколам также.

00:11:38,020 - 00:11:50,780 - Окей, давай маленький кейс. Допустим, ну, я вот заказчик, да, и я хочу, чтобы в нашей системе была доработана форма регистрации пользователей.

00:11:51,360 - 00:12:06,640 - Допустим, тебе нужно написать постановку чисто вот на доработку фронтовой части, доработку формы. Я тебе говорю, я хочу, чтобы эта форма регистрации пользователей содержала фио, ну, электронную почту, чтобы пользователь указывал, и пароль.

00:12:07,480 - 00:12:17,530 - Всё. Достаточно ли тебе информации, чтобы написать постановку? Вот если недостаточно, то вот какие бы ты вопросы уточнил и моменты по поводу этой задачи?

00:12:17,530 - 00:12:35,880 - Ну, я бы первостепенно полез в базу данных, посмотреть, какие данные мы вообще запрашиваем в методе регистрации для того, чтобы определить некий список изменений, да, которые планируются, если функциональность, ну, в дальнейшем будет реализована. Вот. Далее я бы

00:12:35,880 - 00:12:44,560 - дополнительно зафиксировал все изменения, которые планируются и которые были высказаны стейкхолдерам. Соответственно,

00:12:45,420 - 00:12:56,180 - Согласовал бы их с стейкхолдером, который вынес такое пожелание. А далее, соответственно, обращаемся к дизайнеру для того, чтобы внести изменения

00:12:56,520 - 00:13:13,900 - В интерфейсе, и, соответственно, передаем в разработку. Ну, я, допустим, стрейкхолдер. Вот давай мы имитируем, да, реальную задачу. Вот я тебе могу ответить на твои вопросы и согласовать, что ты хочешь. То есть…

00:13:14,220 - 00:13:20,900 - да, там, дизайнеру на прорисовку макета, это окей, это понял. Вот всё-таки по самой форме.

00:13:21,500 - 00:13:23,520 - Чтоб ты еще уточнил.

00:13:24,340 - 00:13:31,080 - То есть, тут нужен список вопросов от меня. Ну не список вопросов, а просто, ну, хочешь, список вопросов, да.

00:13:31,700 - 00:13:47,260 - Ну, как я и сказал, первоначально нужно посмотреть, как сейчас текущая система работает, какие у нас есть обязательные поля на форме регистрации, какие необходимо внести изменения. Вот. Я бы уточнил конечный набор полей, которые нам необходимы.

00:13:48,560 - 00:13:56,720 - Соответственно, с данным конечным набором полей пошло бы вносить изменения уже. Со стороны фронтовой части здесь

00:13:59,340 - 00:14:17,760 - Как я понимаю, у нас уже была реализована некая форма, раз заказчик хочет внести в нее изменения. И, соответственно, уточнил бы дополнительно некие вопросы, которые связаны как раз с интерфейсом. То есть, предполагается ли это изменение

00:14:18,860 - 00:14:33,920 - удаление, да, тех полей, которые были реализованы ранее, или предполагается также изменение не только полей, но и всего интерфейса в целом? Вот. Ага. Давай чуть-чуть тогда уточню.

00:14:34,900 - 00:14:43,100 - Есть, ну, конечный набор полей, да, как я сказал, фива, почта и пароль. Они были до этого, они остаются и сейчас.

00:14:44,640 - 00:14:59,920 - Вопрос такой. Кую бы ты предложил применить на уровне фронта валидацию для этих полей? И предложил бы Ильи вообще. Ну, допустим, вот у нас есть поле «Фио». Я бы, во-первых,

00:15:00,120 - 00:15:15,920 - Ну если можно выявить инициативу, да, высказать, то первоначально я бы уточнил насчет разделения фила на три поля, вот, соответственно. Потом у нас идет почта и пароль. Соответственно,

00:15:16,160 - 00:15:21,140 - Поле ввода почты я бы задал маску, которая, соответственно,

00:15:22,220 - 00:15:41,020 - Валидирует формат как раз имейла, адреса, вот. На уровне… Ну, точнее, не на уровне, а на поле ввода пароля я бы уточнил некие критерии к паролю, которые обычно выставляются, там, не более восьми символов… о, не менее восьми символов, использование специальных символов и так далее.

00:15:42,580 - 00:15:47,460 - Ну, так же, чтобы уточнил, некоторые конечные

00:15:49,320 - 00:16:07,300 - Конечно. Ну, точнее, максимальную длину символов для пароля, вот, дополнительно. Окей. Давай тогда перейдём к следующей теме. Чуть-чуть про архитектуру поспрашиваю. Ну, первый вопрос.

00:16:08,340 - 00:16:14,580 - В своих проектах с какими приложениями сталкивался, на чем они были, на какой архитектуре вы строили?

00:16:14,820 - 00:16:31,140 - Ну, в основном это был монолит. Ну, вот когда мы работали на аутсорсе, там некоторые, конечно, задачи были, точнее, проекты были как монолитные, так и микросервисная архитектура. Вот. Соответственно…

00:16:32,580 - 00:16:41,700 - В остальных проектах тоже такой некий монолит. На текущем месте работаю с микросервисами. Вот.

00:16:42,700 - 00:16:59,660 - Окей. А можешь что-нибудь подробнее про монолиты рассказать? Например, знаешь ли ты, какие они бывают? Ну, во-первых, монолиты бывают распределённые, бывают одномодульные, бывают многомодульные. Вот. Соответственно…

00:17:01,260 - 00:17:14,800 - как говорится, одномодельные и многомодельные, разница между ними как бы лежит на поверхности, что мы реализуем некие такие модули в архитектуре, да, которые отвечают за

00:17:16,000 - 00:17:23,200 - разрабатываемую функциональность за фичи, вот. Ну а, соответственно, основное, как бы, отличие

00:17:23,200 - 00:17:42,220 - Как и от SOA, сервис-ориентированная архитектура в том, что все они развертываются вместе с базой данных, с одной, в отличие от микросервисов. Основное отличие микросервисов — это как раз является отдельное развертывание, то есть каждому микросервису

00:17:42,620 - 00:17:56,980 - дополнительно создается там своя очередь дополнительно поднимается база данных вот и тем самым микросервисе микросервисной архитектуре у нас данные становятся

00:17:57,640 - 00:18:05,280 - Ну, у нас, по сути, нет общих баз данных. Ну, понятное дело, что они могут быть, но это…

00:18:05,580 - 00:18:10,110 - Зачастую это вот отдельные микросервисы с отдельной базой данных, где у нас

00:18:10,110 - 00:18:30,870 - В каждой базе данных хранятся данные для корректной работоспособности вот этого микросервиса, и, соответственно, у нас происходит… Ну, обеспечивает такое некое безопасность, тем самым, что если положится один микросервис, то остальные там будут строить. И, соответственно, у нас

00:18:30,870 - 00:18:33,720 - нету связанных данных, когда у нас

00:18:34,700 - 00:18:55,300 - Между… Ну когда все, например, модули, да, как в многомодульном монолите, обращаются к одной базе данных. У нас там и, соответственно, уменьшается риск конфликтов и так далее. Вот. Окей, да. Давай ты чуть-чуть уже зашёл на территорию другого вопроса, там, начал про какие-то плюсы, да, рассказывать о микросервисах. Сейчас дойдём.

00:18:55,400 - 00:18:58,300 - Давай чуть-чуть ещё вернусь про «Монолит».

00:18:59,960 - 00:19:11,080 - Ну, базовый вопрос, да. В принципе, поняли, что это такое. Давай про его преимущества, его недостатки чуть поговорим. Так, ну, преимущества здесь…

00:19:13,360 - 00:19:31,680 - У нас, получается, есть, ну, реализуются в коде компоненты, которые можно переиспользовать в микросервисах у нас, соответственно. Приходится дублировать эти компоненты, что является минусом, вот. Также в Monolith

00:19:32,780 - 00:19:35,300 - Ну и с преимуществом, если говорить

00:19:36,780 - 00:19:57,400 - Ну, можно сказать, единый источник информации, вот, единая кодовая база, что является, наверное, преимуществом. При масштабировании она… Ну, при увеличении нагрузок она плохо масштабируется, соответственно, это как недостаток. Вот. Также при...

00:19:59,040 - 00:20:16,600 - движи… ну, в течение проекта, соответственно, там в Monolith остается Legacy код, с которым… ну, и, вероятнее всего, разработчики, которые его писали, они могли уйти из компании. И, соответственно, необходимо выделять время для поиска анализа этой

00:20:16,660 - 00:20:20,200 - этого кода, что является тоже минусом.

00:20:20,820 - 00:20:41,720 - Вот. Это, скорее, минус отсутствие документации, чем минус монолита. Ну, как говорится, документация… Какая документация? Если мы говорим про документацию, которая описывает именно код, вот, а если мы говорим про документацию, которую готовят обычно аналитики, как, ну, для конечного пользователя, для…

00:20:42,400 - 00:21:03,080 - написание технического задания разработчикам то соответственно она особо не даст информации другому разработчику как работает тот или иной legacy код хорошо давай тогда теперь про не микросервиса про распределённую архитектуру, в принципе, чуть-чуть поговорим.

00:21:03,580 - 00:21:12,960 - Вот помимо микросервисов какие-то еще, знаешь, паттерны? Ну, P2P, когда у нас есть некие узлы системы,

00:21:13,240 - 00:21:14,920 - Они, соответственно

00:21:14,920 - 00:21:23,190 - географически распределены. У нас нет общего дата-центра. Ну, у нас нет централизованного, можно сказать, сервера.

00:21:23,190 - 00:21:37,020 - Каждый узел содержит, там, полную копию всей сети, и, соответственно, между собой они коммуницируют и валидируют все данные, которые содержат, что позволяет как раз обеспечить

00:21:37,680 - 00:21:45,420 - несколько источников информации при выходе одного узла у нас соответственно не падает со системы

00:21:47,100 - 00:21:53,920 - Глубоко ушёл. Тогда давай поделимся, какие есть ещё, да, варианты распределённой архитектуры.

00:21:54,540 - 00:22:13,180 - Дальше. Какие еще знаешь? Event-driven знаешь? Event-driven, да. Ну, это когда у нас архитектура связана на событиях определенных, которые возникают по некоторым… Ну, когда происходит какое-то событие. Вот. Ну, а как именно она реализуется, не могу сказать.

00:22:13,980 - 00:22:21,820 - Ну давай тогда в архитектуре ещё один вопрос задам. У вас уже много говорили про микросервисы.

00:22:23,440 - 00:22:30,800 - Давай в общем смысле еще раз повторим про их преимущества и недостатки, и потом еще один вопрос задам про

00:22:31,440 - 00:22:34,560 - Связанный с сравнением микросервисов и

00:22:35,120 - 00:22:41,380 - Монолитов. Так, мне сейчас нужно назвать… Микросервис. Да.

00:22:41,820 - 00:23:00,800 - Ну, во-первых, основное преимущество микросервисов — это распределение нагрузки, это масштабирование, отказоустойчивость. Соответственно, у нас есть ограничение функциональ… отграничение функциональности, потому что у нас каждый микросервис, соответственно, за свою функциональность, вот, тем самым…

00:23:01,300 - 00:23:09,320 - у нас не возникает никаких конфликтов, ну, при правильном подходе, соответственно. Вот. Также

00:23:10,900 - 00:23:18,000 - Горизонтальное масштабирование применяется к микросервисам, потому что нам необходимо…

00:23:18,760 - 00:23:35,840 - реализовывать новые микросервисы. Вот. Соответственно, преимущество — у нас отсутствует общая база данных, что, соответственно, повышает уровень безопасности, у нас меньше возникает конфликтов.

00:23:36,460 - 00:23:45,500 - к тому, что каждый микросервис работает со своей базой данных. Вот. Из недостатков у нас нет возможности

00:23:46,080 - 00:23:59,300 - сослаться на реализованные компоненты. Нам необходимо дублировать код, потому что каждый новый микросервис — это каждый новый проект, отдельный новый проект. Вот. Нам необходимо дополнительное

00:23:59,720 - 00:24:09,660 - развертывать всё это дело. Нам необходимо дополнительно всё это поддерживать, что довольно, ну, что не так дёшево со своей стороны.

00:24:10,800 - 00:24:19,800 - С точки зрения производительности, допустим, у нас есть две абсолютно идентичные по функционалу системы.

00:24:20,220 - 00:24:27,920 - Ну они такие, знаешь, в вакууме, допустим, находятся. Вот одна реализована как монолит единый, а вторая точно такая же.

00:24:27,940 - 00:24:37,740 - уу, построена на микросервисах. Как ты считаешь… Ну или неважно, не на микросервисах, а просто на распределённой какой-то архитектуре, на каком-то паттерне. Как ты считаешь,

00:24:37,740 - 00:24:53,740 - Какая из них будет работать быстрее? Ну, тут все зависит от размера проекта вообще в целом, от количества функций, которые предоставляет приложение конечному пользователю. Если говорить… Одинаковое количество, абсолютно. Две идентичные системы. Какая из них будет работать быстрее?

00:24:54,020 - 00:25:00,620 - Ну если говорить о каком-то начальном этапе, то мне кажется, монолит, потому что микросервисы — это

00:25:01,620 - 00:25:05,340 - Непременно явля… Ну, при передаче

00:25:05,360 - 00:25:22,600 - Какой-то информации при взаимодействии у нас появляются посредники. То есть мы передаем информацию какому-то микросервису, он ее обрабатывает, идет другой микросервис, и, соответственно, здесь тратится время на коммуникацию между ними. Дополнительно у нас есть также механизм

00:25:22,600 - 00:25:43,580 - Валидация авторизации микросервиса между друг другом, ну и конечного пользователя, ну клиента с этим микросервисом, что тоже затрудняет процесс. Так что я бы выделил, ну не затрудняет, а я бы сказал увеличит время на обработку. Поэтому я бы выделил, что монолит будет быстрее. Ну, это на первом этапе. Вот. Проект.

00:25:44,360 - 00:25:53,680 - Давай еще один вопрос. Слышал ли ты что-нибудь про оркестрацию и хореографию? Да, слышал, но детально не подскажу про это.

00:25:54,680 - 00:26:00,280 - Окей, давай тогда теперь про нотации пообщаемся.

00:26:01,880 - 00:26:03,520 - Ну, первый вопрос.

00:26:04,220 - 00:26:17,220 - Общий, да. Какие нотации знаешь, в принципе? В UML я слышал, тоже рассказывал. Давайте какие-нибудь другие. — BPMN, RML, UML, C4.

00:26:17,760 - 00:26:28,380 - Вот, такие нотации. А, давай тогда про BPMN. Задаю вопрос. Первый вопрос. Без каких элементов ни одна диаграмма не сможет обойтись?

00:26:29,100 - 00:26:36,420 - Ну, во-первых, без дорожек, без общего пула, без

00:26:37,760 - 00:26:48,700 - событий без связей и без начальной ну точки входа и точки выхода ну когда старт и стоп