

Введение



Вы закончили базовый и продвинутый уровень курса по JavaScript и современным веб-технологиям. Вы начинали с самых азов, а затем погрузились в современные технологии разработки сайтов и обрели глубокое понимание технологии выполнения кода JavaScript в браузере, изучили последние стандарты ECMAScript, знаете, как правильно построить сборку и тестирование приложения. Ваших знаний достаточно для того, чтобы не только создавать полноценные приложения в браузере, которые могут решать реальные проблемы бизнеса, но и разрабатывать их так, чтобы они были быстрыми и удобными в дальнейшем использовании и просты в разработке нового функционала.

Вы уже почти готовы своим кодом менять мир к лучшему! Или, быть может, найти работу своей мечты... Однако остался последний штрих в вашем образовании — **курсовой проект**. Он будет хорошим подспорьем в самопрезентации, так как это ещё один проект в портфолио ваших работ. Мы сделали задание курсовой работы максимально актуальным и приближенным к реальности и считаем, что у нас это получилось хорошо.

До сих пор вы выполняли задания на отработку тех знаний, которые получали в рамках определённого модуля. Теперь же пришло время **объединить весь пройденный материал в едином большом задании**, где вы сможете понять, как они применяются совместно. Этим заданием как раз и является данная курсовая работа.

Некоторые считают курсовую формальностью: «Я получил знания, а бумажка мне не важна». Но диплом — это не просто свидетельство.

Выполненная работа многое даст вам как специалисту:

- ◆ Опыт разработки приложения, максимально приближенного к тому, что бы вы могли делать, устроившись на работу.
- ◆ Уверенность в своих силах, так как вы будете иметь на руках рабочее приложение, которое не стыдно будет добавить в портфолио.
- ◆ Обратную связь от преподавателей, которая поможет вам не просто выполнить задание, а сделать это правильно. Ведь существует множество способов реализовать одно и то же, и не всегда очевидно, как сделать оптимальный выбор.



**Так что предлагаем вам собраться и сделать это!
Будет очень полезно и интересно!**

Описание задачи

Представьте, что вы присоединились
к стартапу 

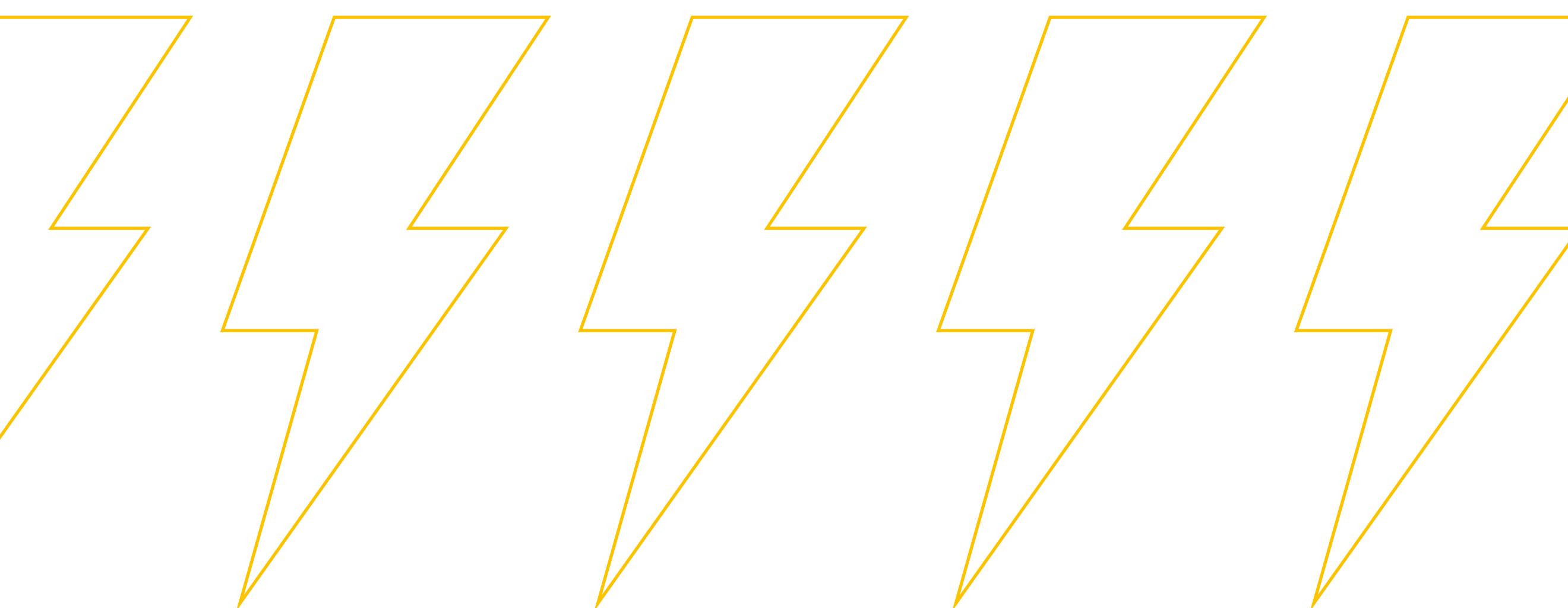


Его миссия — сделать доступными массовые банковские операции с криптовалютами.

Сейчас стартап получил свои первые инвестиции. И, согласно классическому пути своего развития (упрощенно), идея → инвестиции → MVP → продукт, на данный момент стартап находится на этапе MVP. Основатели стартапа уже проработали идею, сделали наброски примерного дизайна, разработали бэкенд (правда, в дальнейшем его, конечно, ещё будут дорабатывать и улучшать, как и сам продукт, но для тестовой среды разработки он уже достаточно пригоден), и теперь они ищут разработчиков, в частности с вашей специализацией — JavaScript frontend-разработчика.

Этот стартап — инновационный проект, призванный объединить воедино традиционный банкинг и криптовалюты. Это будет первый банк на рынке, который открыто и свободно позволит пользователям обмениваться и распоряжаться криптовалютами наравне с другими традиционными валютами.

Возможность поработать в проекте с криптовалютами довольно привлекательна и перспективна для молодого разработчика. Также основатели стартапа выдадут вам часть акций в проекте, помимо неплохой ежемесячной зарплаты, так как вы успели принять приглашение одним из первых. На данный момент в команде вы первый frontend-разработчик, и есть большая вероятность, что стартап вырастет, а вы станете главой отдела frontend-разработки. Итак, вы стали частью молодого проекта и готовы разрабатывать его frontend-часть.



Техническое задание

Необходимо разработать банковскую систему хранения и операций над криптовалютными средствами.

Эта часть техзадания касается frontend-части системы. Frontend будет представлять собой веб-приложение.

Требуется следующий основной функционал приложения:

- Авторизация
- Управление счетами пользователя: создание нового счёта, отображение списка счетов, отображение баланса, просмотр истории транзакций
- Переводы на счета или карты других пользователей
- Возможность производить валютные обмены
- Отображение банкоматов на карте

Необходимо, чтобы веб-приложение имело следующие разделы:

- Форма входа пользователя
- Список счетов пользователя
- Просмотр информации о существующей карте
- Форма для перевода средств
- Подробная история баланса по карте
- Мониторинг курса валют и валютные переводы
- Страница отображения точек банкоматов на карте

Дизайн доступен в виде макетов в Figma:

<https://www.figma.com/file/JUJVDoP27x18v4Eqt66SdK/Bank-Diploma?node-id=1%3A3>

Backend, инструкция по его использованию, документация его методов API и дополнительные материалы находятся в репозитории

https://gitlab.skillbox.ru/learning_materials/js-advanced-diploma

Форма входа пользователя

Coin.

Вход в аккаунт

Логин Placeholder

Пароль Placeholder

Войти

На экране входа пользователь вводит логин и пароль, после чего авторизуется в системе. Регистрация пользователей через портал осуществляться не будет, их регистрируют в другом месте, по приглашениям, так как продукт пока что MVP.

Необходима валидация на заполненность поля «Логин и пароль». На данный момент не поддерживаются пароли и логины длиной менее 6 символов и с пробелами. Поэтому это надо тоже отразить в валидации.

После авторизации пользователь попадает в свой личный кабинет. По умолчанию в раздел просмотра список своих счетов.



Список счетов пользователя

Coin.

Банкоматы Счета Валюта Выйти

Ваши счета Сортировка + Создать новый счёт

123456788932021 3 523 ₽ Последняя транзакция: 21 августа 2021	123456788932021 3 523 ₽ Последняя транзакция: 21 августа 2021	123456788932021 3 523 ₽ Последняя транзакция: 21 августа 2021
123456788932021 3 523 ₽ Последняя транзакция: 21 августа 2021	123456788932021 3 523 ₽ Последняя транзакция: 21 августа 2021	123456788932021 3 523 ₽ Последняя транзакция: 21 августа 2021
123456788932021 3 523 ₽ Последняя транзакция: 21 августа 2021	123456788932021 3 523 ₽ Последняя транзакция: 21 августа 2021	123456788932021 3 523 ₽ Последняя транзакция: 21 августа 2021

Это страница, на которую попадает пользователь после того, как вошёл в систему. Здесь отображаем список счетов, которыми владеет пользователь.

Все счета на данной странице «условно-рублевые», на самом деле банк за кадром обслуживает их через виртуальную валюту с использованием блокчейн-технологии, но для пользователей эти счета просто представляются как обычные удобные рублёвые счета. Выводим баланс данных счетов с постфиксом валюты рубля.

Кнопка **«Создать новый счёт»** добавляет новый счёт данному пользователю, никаких дополнительных настроек не требуется, счёт создаётся сразу в автоматическом режиме, достаточно лишь вызвать метод на бэкенде и перезапросить список счетов, туда сразу добавится новый счёт с произвольно присвоенным ему номером.

Кнопка **«Открыть»** на каждом счёте ведёт на страницу подробного просмотра информации о счёте. (Описание ниже, в следующем разделе).

Необходимо предусмотреть возможность сортировки счетов. Так как предполагается, что пользователю не потребуется большое количество счетов, то все сортировки выводим в порядке возрастания. Необходима сортировка по признакам: **«Номер счёта», «Баланс», «Время последней транзакции»**.

Ваши счета

Сортировка ▾

+ Создать новый счёт

По номеру

По балансу

По последней транзакции

123456788932021

3 523 ₽

Последняя транзакция:
21 августа 2021

Открыть

123456788932021

3 523 ₽

Последняя транзакция:
21 августа 2021

Открыть

123456788932021

3 523 ₽

Последняя транзакция:
21 августа 2021

Открыть

123456788932021

3 523 ₽

Последняя транзакция:
21 августа 2021

Открыть

123456788932021

3 523 ₽

Последняя транзакция:
21 августа 2021

Открыть

123456788932021

3 523 ₽

Последняя транзакция:
21 августа 2021

Открыть

123456788932021

3 523 ₽

Последняя транзакция:
21 августа 2021

Открыть

123456788932021

3 523 ₽

Последняя транзакция:
21 августа 2021

Открыть

123456788932021

3 523 ₽

Последняя транзакция:
21 августа 2021

Открыть

Просмотр информации о существующей карте
и форма перевода средств

Просмотр счёта

← Вернуться назад

№ 12455242373623463

Баланс 31 235 ₽

Новый перевод

Номер счёта получателя

Placeholder

Сумма перевода

Placeholder

Отправить

Динамика баланса



История переводов

Счёт отправителя Счёт получателя Сумма Дата

123545123541 123545123541 - 1 234 ₽ 31.12.2021

123545123541 123545123541 + 1 234 ₽ 31.12.2021

← Кнопка «Вернуться назад» возвращает пользователя на страницу со списком карт.

На странице имеются следующие элементы:

- Форма переводов
- График истории баланса
- Список прошлых транзакций

Форма переводов имеет поле для ввода счёта получателя (также там необходимо автодополнение по уже использовавшимся счетам получателя, берём данные из localStorage). Поле ввода суммы переводов. И кнопка, которая производит отправку данного перевода. Предусмотреть валидацию на заполненность полей, валидацию на ввод положительной суммы (отрицательную сумму отправить должно быть невозможно).

После успешной отправки перевода запоминаем использованный счёт получателя в localStorage для дальнейшего участия в автодополнении.

График истории баланса выводит bar chart истории значений баланса данного счёта за последние 6 месяцев (или меньше, если данных недостаточно), самый актуальный месяц — справа. График вертикально масштабируется автоматически, чтобы самое максимальное значение соответствовало верхней границе отображаемого графика. Цифры справа от графика соответствуют минимальным и максимальным значениям баланса за отображаемый период.

История переводов выводит таблицу из 10 (или менее) записей последних транзакций с участием текущего просматриваемого счёта. Отрицательные суммы (исходящие переводы) выделяем красным цветом, положительные суммы (входящие переводы) выделяем зеленым цветом. Дату форматируем в формате День.Месяц.Год.

График баланса и история переводов кликабельны, при клике на них пользователь переходит в раздел подробного просмотра истории баланса



Подробный просмотр истории баланса

Coin.

Банкоматы

Счета

Валюта

Выход

История баланса

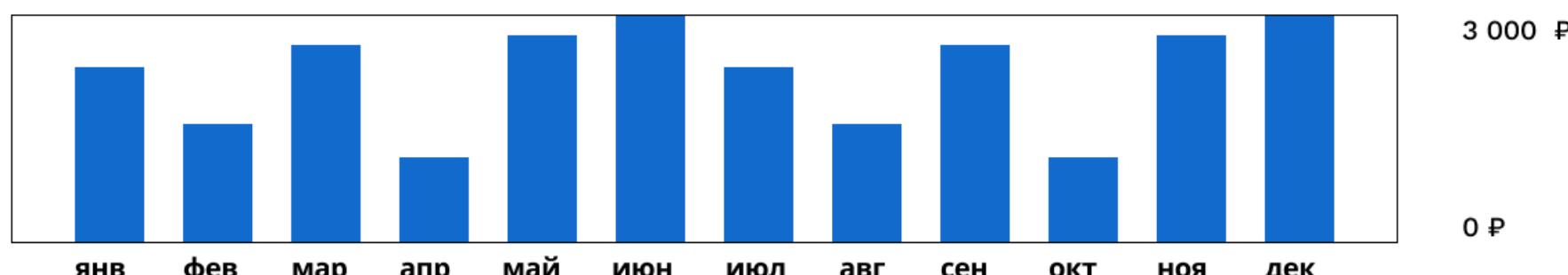
← Вернуться назад

№ 12455242373623463

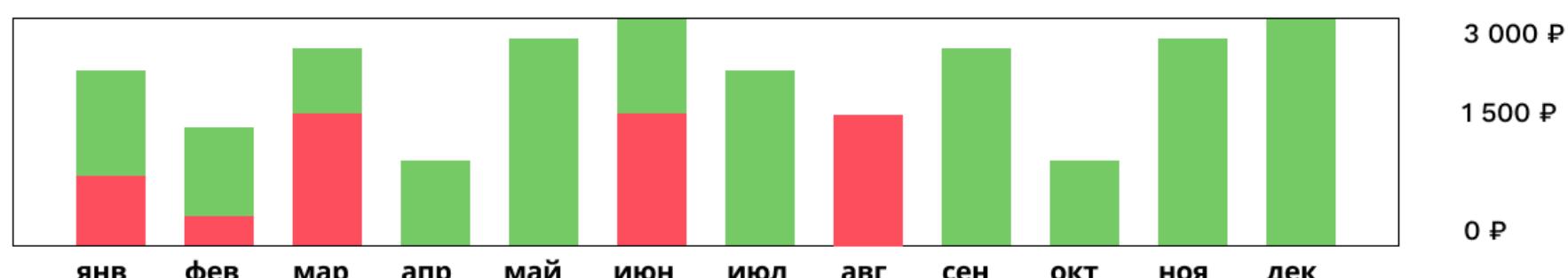
Баланс

31 235 ₽

Динамика баланса



Соотношение входящих исходящих транзакций



История переводов

Счёт отправителя	Счёт получателя	Сумма	Дата
------------------	-----------------	-------	------

123545123541 123545123541 -1 234 ₽ 31.12.2021

123545123541 123545123541 +1 234 ₽ 31.12.2021

← Кнопка «Вернуться назад» возвращает на экран просмотра счёта

График динамики баланса выводит bar chart истории значений баланса данного счёта за последние 12 месяцев (или меньше, если данных недостаточно), самый актуальный месяц — справа. График вертикально масштабируется автоматически, чтобы самое максимальное значение соответствовало верхней границе отображаемого графика. Цифры справа от графика соответствуют минимальным и максимальным значениям баланса за отображаемый период.

График «Соотношение входящих и исходящих транзакций» аналогичен по своему функционалу графику динамики баланса, описанному выше, однако с некоторыми отличиями. Отличие в том, что мы теперь выводим раскрашенные частями полоски, красная часть полоски отражает процент расходных (негативных) транзакций в этом столбике, зелёная часть полоски отражает процент доходных (положительных) транзакций в этом столбике. Справа вводится ещё одна цифра между максимальным

и минимальным значением баланса, это цифра, показывающая максимальное значение либо отрицательного, либо положительного соотношения транзакций, смотря какое меньше.

История переводов выводит таблицу из 25 (или менее) записей последних транзакций с участием текущего просматриваемого счёта. Если в истории переводов больше записей, чем 25, тогда вводим механизм постраничного отображения. Отрицательные суммы (исходящие переводы) выделяем красным цветом, положительные суммы (входящие переводы) выделяем зеленым цветом. Дату форматируем в формате День.Месяц.Год.

Валютные инструменты



Coin.

Банкоматы

Счета

Валюта

Выход

Валютный обмен

Ваши валюты

ETH	6.3123545131
BTC	1.4543231123
JPY	32 151 225 122
USD	1 151 225 122
RUB	5 151 225 122
BYR	1 225 122

Обмен валюты

Из в Обменять

Сумма

Изменение курсов в реальном времени

BTC/ETH	6.3123545131	▲
BTC/ETH	6.3123545130	▼
BTC/ETH	6.3123525131	▼
BTC/ETH	6.3123545131	▲

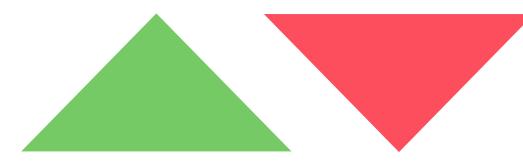
Здесь пользователи могут получить информацию о состоянии своих валютных счетов, быть в курсе последних колебаний курса и обменять одну валюту на другую.

Список всех известных валют получаем из API. Условно считаем, что у пользователя уже открыты счета во всех известных валютах. Не отображаем валютные счета с нулевым балансом. При переводе из одной валюты в другую бэкенд может выдать ошибки о недостатке средств валюты списания, эту ошибку надо обработать.

Предусмотреть валидацию на запрет перевода отрицательной суммы.

В части страницы «**Ваши валюты**» выводятся коды валютных балансов пользователя и текущее значение баланса.

Справа находится вывод изменения курса обмена валютных пар. Валютные пары выводятся через слэш в качестве разделителя. Около цифры значения изменения курса отображается стрелка направленная вверх или вниз в зависимости от текущего движения курса. Положительное и отрицательное движение подсвечивается зелёным и красным цветом стрелки соответственно.



Карта банкоматов

Coin.

Банкоматы Счета Валюта Выйти

Карта банкоматов

Адрес или объект Найти Пробки Слои ▾

Москва

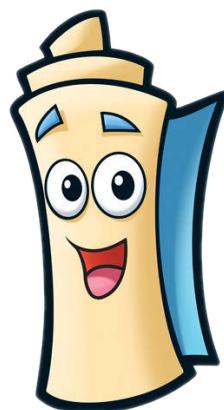
6 км

© Яндекс Условия использо

Открыть в Яндекс.Картах Создать свою карту

В этом разделе выводим карту с точками, где располагаются банкоматы, где может обслужить наш банк.

На данный момент простой метки на карте достаточно, если же необходимо отобразить какой-то заголовок или подобную текстовую информацию на метке, то предлагается использовать название компании — Coin.



Несмотря на то, что в дизайне вывод карты выглядит визуально похожим на Яндекс.Карты, вы вольны использовать любой другой провайдер карт, сможет удовлетворить требуемому функционалу выше (вывод своих меток). Однако предпочтение желательно отдать Яндекс.Картам или Google Картам, если есть возможность.

Технические требования

Выполненную работу необходимо предоставить в виде отдельного репозитория

Репозиторий должен включать в себя необходимые ресурсы для приложения и файл **README** с инструкциями по запуску сборки приложения. Результатом сборки будет папка **dist** (должна создаваться или очищаться автоматически в начале процесса выполнения сборки), в которой находится страница **index.html** в качестве точки входа в приложение, а также все ресурсы, необходимые приложению для работы. Бэкенд на данный момент разработан, однако ещё немного сыроват и иногда может возвращать какие-то неполные данные или данные заглушки. Однако структуру данных и их формат (контракт API) бэкенд соблюдает корректно. Поэтому необходимо обрабатывать какие-либо неполные данные, чтобы не происходило сбоев в работы программы, уместно выводить ошибку каким-либо образом в случае получения неполных данных. **Ошибки в работе бэкенда не должны влиять на работоспособность приложения в целом, только на отображение отдельных конкретных частей.**

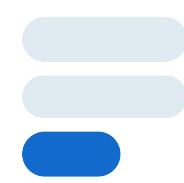
При долгой загрузке каких-либо данных нужно отображать какую-либо визуальную индикацию загрузки данных. Например **spinner**, **loader** или **skeleton**.



Spinner



Loader



Skeleton

Обычно циклическая анимация загрузки, например, крутящийся (откуда собственно и название) полукруг.

Индикация загрузки в виде прогресс-бара или полоски. Иногда делают так, чтобы была горизонтальная небольшая полосочка и бегала влево-вправо по всей ширине экрана

Более продвинутая техника, она даёт пользователю понимание не только о том, что контент грузится, но ещё позволяет заранее представить форму будущего контента в виде силуэтов

В проекте должны присутствовать **end-to-end-тесты** на базовый функционал приложения. Базовым функционалом приложения считаем: возможность авторизоваться, возможность просмотреть список счетов, возможность перевести сумму со счёта на счёт, возможность создать новый счёт и перевести с него сумму тоже. Опишите в README инструкцию по их запуску и проверке результатов прохождения этих тестов, чтобы можно было в любое время убедиться в целостности и работоспособности проекта.

Также приветствуется написание **unit-тестов** на компоненты приложения, хорошие кандидаты для покрытия unit-тестами — это вспомогательные функции: утилиты, чистые функции, чистые компоненты.

Улучшения и дополнения

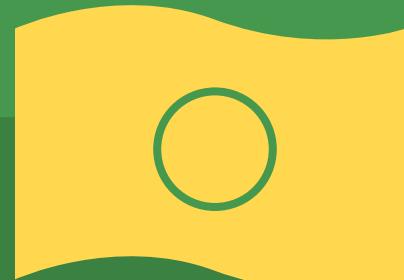
Если вы уже выполнили задание по ТЗ, вы можете попробовать свои силы в дополнительных заданиях. Они расположены по сложности — от более простых к более сложным. 💪😎



Адаптив — адаптировать сайт под мобильные экраны

Так как мы стартап, нам надо двигаться быстро. В связи с этим не все моменты были продуманы на этапе проектирования MVP. Одна из таких вещей, которую не проработали в дизайне, — адаптив. До этого дизайн делал фрилансер, однако в этот раз он не имеет возможности помочь нам, компания пока в поиске дизайнера в штат, поэтому адаптив придётся делать frontend-разработчику на своё усмотрение.

Необходимо, чтобы сайт красиво (или хотя бы доступно) смотрелся на следующих ширинах экрана и промежуточных между ними: 320, 768 и 1200 пикселей.



Отображение типа платёжной системы карты

Теперь ваш стартап может производить переводы не только между криптовалютными счетами, но также принимать и отправлять переводы на обычные банковские карты.

В связи с чем надо отображать иконку платёжной системы (хотя бы для систем VISA, MASTERCARD и MIR, можно и других) в форме отправки перевода и истории транзакций.

Отображение по следующей логике: валидируем введённый номер, если номер невалидный, оставляем текущее поведение без иконки. Если номер валидный, отображаем иконку платёжной системы рядом с номером.

В платежной форме отображаем иконку рядом с полем ввода номера. В истории транзакций иконка рядом с номером счёта отправителя или получателя.



Кэширование данных

Те данные, которые редко меняются в пределах сессии, можно запоминать и не перезапрашивать с сервера так часто. Также это хорошо улучшает опыт пользователя, создавая впечатление того, что сайт загружает данные моментально.

Рассмотрите, к каким данным и разделам сайта вы можете применить технику кэширования, и реализуйте это.

Также развитием этой техники является техника фоновой подгрузки данных. Сначала вы отображаете данные из кэша, тем самым создавая у пользователя иллюзию моментальной загрузки страницы, в то время как фоном делаете запрос на обновление этих данных. Когда обновлённые данные вернулись, вы подменяете отображаемые данные на новые.



Кастомизация выкладки страницы пользователем

Многим пользователям было бы удобно менять местами блоки, отображаемые на сайте. Например, кто-то хотел бы поменять местами форму перевода и блок динамики баланса на странице просмотра счёта. Кому-то хотелось бы в другой последовательности расставить блоки на странице подробной информации о истории изменения баланса.

Реализуйте возможность пользователю самому менять порядок блоков на сайте.

Рассмотрите, с какими блоками будет особо полезен данный приём, и реализуйте для них это. Особенно хорошим станет решение, которое будет работать для практически всех блоков на сайте.

Формат сдачи задания

Исходный код выполненного задания должен быть выложен на **GitLab**.

Инструкции по сборке проекта должны присутствовать в **README** репозитория.

Результатом сборки должно быть создание папки **dist** с необходимыми ресурсами для работы приложения и файлом **index.html**, который является точкой входа в приложение.

Также в текстовый файл **README.md**, помимо инструкций сборки, надо добавить различные пояснения для проверяющих преподавателей, которые вы считаете важным передать. Ещё здесь должен быть добавлен список тех улучшений и дополнений, которые вы реализовали.

Критерии оценки

Преподаватель проверяет работу по следующим критериям:

- Соответствие разработанного интерфейса требованиям в техническом задании.
- Наличие интеграционных тестов и их успешное прохождение.
- Понятность написанного кода. Сюда же можно отнести корректное именование переменных, констант и функций, которое должно соответствовать тому, что в них содержится.
- Единый стиль написания кода, в том числе:
 - нет смешанного именования (camelCase, snake_case);
 - везде используются одинаковые отступы (2 пробела, 4 пробела или 1 табуляция);
 - единый стиль объявления функций (function declaration или function expression);
 - единый стиль работы с массивами (цикл for of или метод forEach).
- Вы можете делать проект в соответствии с кодстайлом Skillbox, а можете практиковать собственный стиль. Главное — его единообразие.
- Правильность использования элементов HTML.
- Правильность использования CSS стилей.

➤ Уместность использования средств языка JavaScript. Сюда могут относиться следующие обязательные моменты (но не только они):

- для переменных используется `let`;
- для констант `const`;
- предпочтение отдаётся `const` всегда, когда это возможно;
- изоляция области видимости;

➤ И рекомендации:

- дублирующиеся участки кода вынесены в функции;
- правильное использование методов массива (`forEach`, `filter`, `find`, `reduce`);
- осознанная работа с объектами как со ссылочным типом; разделение логики и представления (это значит, что в работе вы должны отделять отрисовку интерфейса и работу с сервером).

Что ещё может улучшить работу:

- Уместный и красивый адаптив.
- Использована техника `skeleton` для индикации загрузки, а не `loader` или `spinner`.
- Приложение отображает понятные и уместные ошибки, дающие пользователю понять, почему была ошибка и как выйти из этой ситуации.
- Написаны `unit`-тесты на многие внутренние компоненты программы и проверки выполняются успешно.
- Написаны интеграционные (`end-to-end`) тесты не только на базовый функционал приложения, а, например, ещё и на раздел валютных обменов + карту банкоматов и просмотр истории транзакций.

Работа возвращается на доработку **сразу**, если:

- отсутствует или не работает базовая функциональность, указанная в техническом задании;
- отсутствуют интеграционные (`end-to-end`) тесты на базовый функционал;
- имеются грубые ошибки в организации или структуре проекта;
- преподаватель не может запустить проект локально.

Рекомендации по выполнению

- Внимательно изучите документацию. Убедитесь, что вам всё понятно. Это первый и очень важный шаг в начале работы над любым проектом. Если у вас возникают вопросы, фиксируйте их для себя и обязательно задавайте своему куратору дипломного проекта. В реальных проектах уточнять требования у вашего заказчика — это совершенно нормально. Можно даже сказать, что это часть вашей работы.
- Помните про такие стандартные технологии, как local storage, session storage, fetch, web sockets. Их использование просто необходимо для успешного выполнения разработки в рамках данной курсовой.
- Не пытайтесь делать всё и сразу, разделите работы по вёрстке и по реализации логики программы. Так вы сможете сфокусировать своё внимание на чём-то одном, что значительно упрощает работу. Каждый вид работ тоже можно разделить на подзадачи, вполне вероятно, вам так будет проще и удобнее работать.
- Составьте план работы по датам. Исходя из нашего опыта, продуктивнее выделять на курсовой проект по 2–3 часа несколько дней в неделю, чем делать этот же объём за один подход. Рекомендуем придерживаться этого графика и обязательно выделять время для отдыха.
- Отмечайте свой прогресс по мере выполнения плана. Это полезно по нескольким причинам: во-первых, вы будете держать ритм, во-вторых — контролировать ситуацию. И самое главное: каждый выполненный этап — это ваша маленькая победа. Чем больше таких побед вы будете замечать, тем большее удовольствие от выполненного проекта получите в итоге.
- Прежде чем отправлять каждую часть проекта на проверку преподавателю (а в реальной работе — тестировщику), всегда проверяйте за собой. Мы рекомендуем вам проверить весь проект целиком, когда вы его закончите. Идеально, если между завершением работы и вашей проверкой пройдёт некоторое время — это позволит вам отключиться и в результате посмотреть на работу свежим незамыленным взглядом. И помните, что ошибки при работе — это нормально, их совершают даже самые матёрые разработчики. Но по-настоящему хорошие специалисты отличаются тем, что способны самостоятельно найти свои ошибки и исправить их ещё до того, как их работа будет передана дальше.
- Для сложной структуры HTML вы можете разбить код на несколько вспомогательных функций, чтобы было проще ориентироваться. Например, для таблицы вы могли бы сделать функции для создания заголовочной строки и строки со значением (строки транзакций, например). Также было бы удобно выделить в отдельную функцию повторяющиеся блоки, например, создание блока отображения счёта, и переиспользовать эту функцию в дальнейшем несколько раз с разными входными параметрами, чтобы менять отображение.

Полезные ссылки и материалы

Подключение Яндекс.Карты:

<https://yandex.ru/dev/maps/jsapi/doc/2.1/quick-start/index.html?from=techmapsmain>

Пример использования пользовательских меток на картах:

<https://yandex.ru/dev/maps/jsbox/2.1/placemark>

Подключение Google Карты:

<https://developers.google.com/maps/documentation/javascript/overview?hl=ru>

Пример добавления маркера на Google Карты:

<https://developers.google.com/maps/documentation/javascript/adding-a-google-map?hl=ru>

<https://developers.google.com/maps/documentation/javascript/marker-clustering?hl=ru>

Простые JS-фреймворки, упрощающие работу с DOM:

<https://github.com/redom/redom>

Сборщики модулей:

<https://parceljs.org/>

<https://webpack.js.org/>

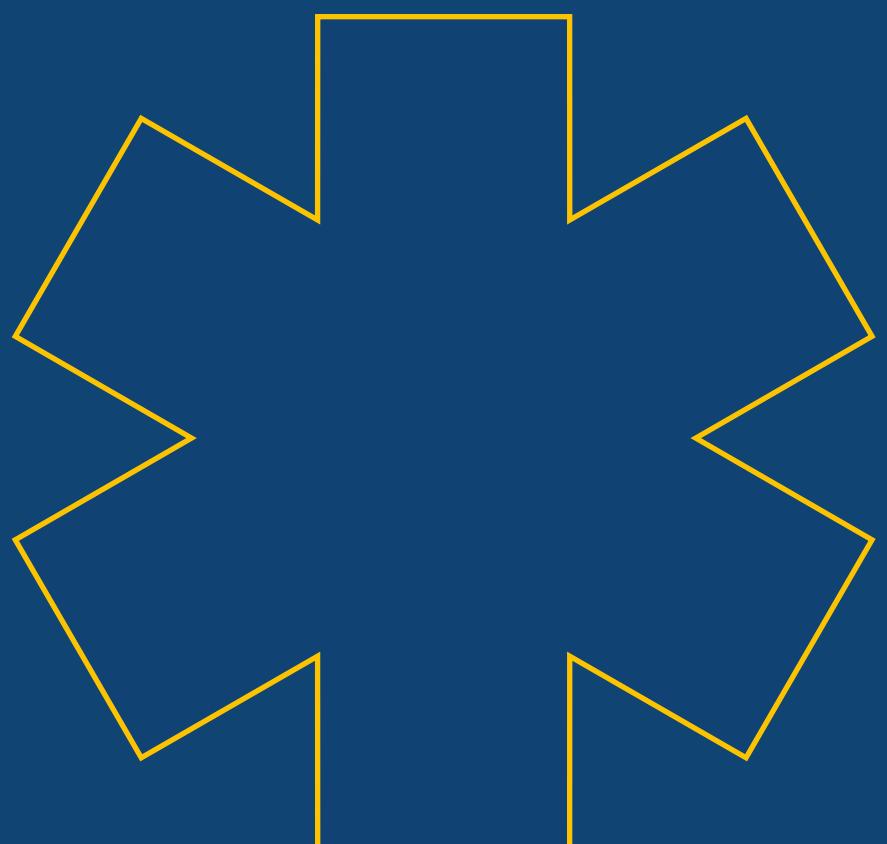
Библиотеки отрисовки графиков:

<https://www.chartjs.org/>

<https://plotly.com/javascript/>

(либо можно самому на DOM или на Canvas)

https://developer.mozilla.org/en-US/docs/Web/API/Canvas_API



Статья про технику индикации загрузки вида skeleton:

<https://uxdesign.cc/what-you-should-know-about-skeleton-screens-a820c45a571a>

