Project specification for StreetARTists

General

Project Goal

To create web space for every street artist that will allow them to be able to track their income, and to have a place on the web where they can showcase their masterpieces.

What problem does this project solve?

Currently many artists track their expenses in the old traditional way by writing the income amount in notebooks, and they have no idea how much they earned per performance, or how much time has been spent creating the art. By creating a platform that will have the ability to track their income, also will be able to see statistics that will tell them the average price per piece, the amount that they spent to make that masterpiece, even to be able to display to them the average price per hour of their work, etc.

Also, many artists nowadays rely all their online presence on their social networks like Instagram, Facebook to showcase their masterpiece, which in some cases looks unprofessional. By creating profile pages as part of the platform, artists will be able to upload their performances and showcase their masterpieces that will allow them to be discovered easier.

What is the vision?

To help young artists to express themselves through modern technology and to give them the opportunity to be reachable worldwide.

Pages

- Landing Page (index.html)
- Artists Home Page (index.html#artists)
- Artists Items (index.html#artists/items)
- Live Auction (index.html#auction)
- Visitor Home Page (index.html#visitor)
- Visitor Listing (index.html#visitor/listing)

You can add more or subtract

Resources

Icons: Font-Awesome or any other
Chartjs: https://www.chartjs.org/docs/latest/getting-started/
Users (Artists): https://jsonplaceholder.typicode.com/users
Fonts: Reenie Beanie https://fonts.google.com/specimen/Reenie+Beanie,
Roboto https://fonts.google.com/specimen/Roboto
Design: https://www.figma.com/file/4Ds1F0I8h5N76XWw6nUK9P/Figma-Basics?node-id=103%3A40
BiddingAPI: https://blooming-sierra-28258.herokuapp.com/bid
BiddinAPI description: Send POST request with { amount: number }

Landing Page

Join as Artist

Clickable div with select to choose from multiple artists. When choosing an Artist there is an additional dropdown to select the artist. All the artist options are names taken from this endpoint https://jsonplaceholder.typicode.com/users. Depending which artist is chosen, his name will be displayed in the header in next pages.

Join as Visitor

If the visitor option is chosen you will be redirected to the visitors home page.

Visitor - Home Page

After choosing visitors from the Landing Page, users should be redirected to this page.

Header is replaced according to design.

Header

Contains the Logo, static text “Street ARTists”, and auction icon button. When clicked on the auction icon, the user should be redirected to the Auctions Page.

Banner

Contains call to action button, when clicked it should redirect to Visitor Listing Page

Sliders

Contains 2 rows of images that should animate, First row slowly from right to left, Second row from left to right. On any image click user should be redirected to Visitor Listing Page

Carousel

Custom designed carousel where user should be able to infinite scroll between 3 items. Use any custom images of your choice, and Lorem ipsum text as placeholders. But match the design.

Visitor - Listing Page

Here the user is able to list all the available items from different artists. There are filters available for easier findings.

Listing

Cards according to design should be listed in Even / Odd order, meaning every even indexed card should have different design than the odd indexed card (use Figma for details)

Cards are rendered from array “items” found in data.js, filter the array to display only cards that have isPublished: true

Filters

When clicking the floating filter icon button (bottom right corner), the Filters panel should slide from right to left covering the whole screen and displaying following filter values.

- By Item Name (Input type text)
  This value, if any, it should be used to match through items names

- By Artist (Input type select)
  No default options should be preselected, if anything is chosen this value should be used to match through the item artist name. Options are taken from this endpoint (https://jsonplaceholder.typicode.com/users) as name

- By price
  Contains 2 inputs of type number, one for min value, another for max value. If any value is present, these values should be used to filter out Items that have price higher than min value and lower than max value.
- By Type
  No default options should be preselected, if anything is chosen this value should be used to match through item type. Options are taken from the data.js file as ItemTypes.

- Apply Icon Button
  When clicked, Users should be redirected to Visitors Listing but only list items that meet the chosen filter values.

- Close Icon Button (top right corner)
  When clicked it should only close the filters, the rendered items should be same as before opening the filters

Artist - Home Page

Here you can see the current status of the current logged in (chosen) artist items, containing 3 different widgets each of them displaying different data

1. ### Total Items Sold

   Example 12/35 meaning 12 items are sold out of 35 in total published. One item is considered as sold only if there is dateSold property available, in the list of Items.

2. ### Total Income

   Accumulate priceSold property from all Items.

3. ### Live Auctioning Item

   This widget should be available only if there is a live auction ongoing. (meaning track the state of auctioning in global a variable to be aware of in all sections/pages. The amount displayed is the current bid of the item auctioned. On click on this widget redirect to Auction Page (index.html#auction) should happen.

4. ### Chart

   There are quick picker options, meaning that the chart should display a series regarding the option chosen. If the user picks the last 7 days, there should be bars for each date representing the amount of items sold that day. This value should be taken from “Items” where first we filter by current logged in artist, then we filter by dateSold where min date is today - 7 days, and max date is today, and then we accumulate the priceSold amount.

Last 14 days, and last month apply the same logic.

Last year should display only 12 bars representing the last 12 months. Meaning you should group the found amounts in date range from start to end of month

For drawing the charts you should use this library (chartjs - https://www.chartjs.org/docs/latest/getting-started/). Chart should be styled according to the design provided. We are using the ”Horizontal Bar Chart” under the Type Bar Chart. The key for drawing this chart will be to gather the required data in a way that chartjs wants to be provided.
Hint: Spend time reading the chartjs documentation.

Artist - Menu

Clicking the menu button in the header when artist is logged in, should slide down panel with 3 list items. Each one should lead to different pages, Home, Items, Auction

Home

Redirects to Artist - Home Page (index.html#artists)

Items

Redirects to Artist - Items Page (index.html#artists/items)

Auction

Redirects to Auction Page (index.html#auction).

Artist - Items Page

All Items from the artist that is logged in are displayed as cards, similar to the cars in the visitor listing page but these cards contain additional control buttons. Send to Auction, Publish/Unpublish, Remove, Edit.

Send to Auction

This button should send this Item to live auction. It is only enabled if there is no other ongoing auction already!

Publish/Unpublish

This button toggles the state of the item, when clicked it toggles the property of the item (isPublished). This property is later used into Visitors Listing Page where only items with isPublished: true should be listed

Remove

Clicking will show a confirmation popup with any custom design that contains “confirm” and “cancel” options. Even a regular confirmation popup is enough. After confirming that card should be removed from HTML as well as from the data (the Items array)

Edit

Opens the Artist Add New Item Page but in EDIT mode with information about the current card. All the fields are prefilled and when confirming the update, the HTML and the data should be updated.

Add New Item Card

This Card should be present at the bottom of the cards, always. Clicking on it opens Artist Add New Item Page in ADD mode

Hint: *When matching items from the array in order to update/remove, the best approach would be using the item.id. First take care of the list of objects, than call the render functions to go through data and update the html*

Artist - New/Edit Item

New

Opens page for creating new item. Each item object should have all of these properties.

- image (required)
    - Only when on mobile view, “Take a snapshot” icon button is visible, clicking on it should display a popup for taking images from your device. This popup is described in the next section.
- title (required)
    - String
- description (optional)
    - Textarea
- type (required)
    - Select from options taken from data.js itemTypes
- isPublished (required)
    - Checkbox with checked as default value. This value should be taken into consideration when rendering artists items for visitors. (only if isPublished is true, than show the card)
- price (required)
    - Input type number

- artist (required invisible)
    - String - auto filled with current logged in user name
- dateCreated (required invisible)
    - Will be auto filled with today's date
- dateSold (optional invisible)
    - Will be auto filled when auction finish about this item
- priceSold (optional invisible)
    - Will be auto filled when auction finish with the last bid amount
- isAuctioning (optional invisible)
    - Will be set to true when Artist clicks on “Send to Auction” button

On Add, HTML and Data (ListItems) should be pushed with the new item and it’s values.

On Cancel, everything should be discarded, HTML and Data should not be changed.

Edit

Same page, but with prefilled inputs from the item that is clicked. After confirming the update, HTML and Data (ListItems) should be updated with the edited item values

Same hint applies here: *When matching items from the array in order to update/remove, the best approach would be using the item.id. First take care of the list of objects, than call the render functions to go through data and update the html*

Artist - Capture Image Popup

This popup should handle taking images directly from your device and storing them as image property into your list of items

Take a Snapshot (icon button)

On click the snapshot preview should be captured, popup should close and preview of image should be displayed in Artist - Add/Edit Item Page

Retake Snapshot

If user wants to retake the snapshot, he can click on the image, and the same image capturing popup should appear

Auctioning

Here you are required to show your UX/UI skill set. This is a page where you need to create (simulate) Live auctioning of some Item. The Main idea is to have only one item that is on auction. This item is set only by the artists in (Artists Items Page) when clicking on the “Send to Auction” button. Access to this page has artists and Visitors, but only Visitors can bid, and

artists can only observe (the bid button should be disabled for artists).

Once an item is sent to auction, the timer should start counting (sec by sec) from 2:00 minutes to 0. For bonus points (5) this timer value should be stored in localStorage so if the user refreshes the page, the timer should continue counting from where it stopped.

Users should be able to see all the bids that are happening.

Initial price of item is half from the current bidding item object property “price” (item.price / 2)

Only Visitors can bid by applying the amount of the next price and clicking on some button to confirm the bid.

When the Bid button is clicked, a request is sent to this endpoint(ask your mentor for URL), the button becomes disabled. And from the response property isBidding will decide if another person is bidding for the same item. If isBidding is true, then additional property “upBid” will contain the extra amount that needs to be added to the current active bid.

Example
If the user bids 500, the request response will return { isBidding: true, upBid: 50} meaning his bid price will be 500 + 50 = 550. Everytime response returns isBidding: true, in timer time should be increased for 1 minute. If the response return {isBidding: false} than user should wait until the timer ends and when timer becomes 0, the item is updated with priceSold: [lastBidAmount], dateSold: new Date(), isAuctioning: false

Scoring

All the application should be dynamic and data driven, not single value should be hard coded

Think as you have back-end APIs, always change data via separate functions,
Suggestion, create separate functions for:

- Creating new Item,
- Editing existing Item,
- Getting all Items,
- Delete single Item

This way will be much easier to achieve the bonus 20 points and later to add the back-end communication.

Landing Page	10
Artists Home Page	10
Artists Menu	5
Artists Items Page	5
Artists Add New Item	10

Artists Capture Image Popup	15
Artists Edit Item	5
Visitors Home Page	10
Visitors Filters	15
Auctioning	15
BONUS: Local Storage support, for the items list	15
BONUS: Local Storage support, auctioning timer	5