

# ES6 & HTML5

Project name: place-keeper

## General

Remember to use the <section, nav, main, aside, header, footer> [semantic elements](#)

Use [ES6](#) throughout your code.

Use the [MVC](#) pattern to shape your app, you should have the following services:

- *utilService*
- *storageService*
- *userPrefService*
- *placeService*

Build a webapp with 3 pages:

## index.html

This is a simple home page with a some graphics and a welcome message, something like: *Find your way back to your best places*

Add navigation links to the other pages

## user-prefs.html

Here we will use a <form> to get the user preferences and save them to localStorage.

The application should use those CSS preferences and show the pages accordingly.

## Step 1 - Colors

Use HTML5 color <input> to let the user set its background and text color of the home page, keep in localStorage and respect his choices in next visits.

## Step 2 – Date and Time

Use HTML5 date and time <input>s to let the user set his exact birth time, when he does show a random astrological forecast in the home page (from an array of 3 ready-made forecasts)

### Step 3 – Wrap in a form

Put those inputs in a `<form>`, and when submit, use a service to keep them in a localStorage object: `userData`

TIP: you will need `event.preventDefault` in the `onsubmit` event handler.

### Step 4 – Add some more inputs

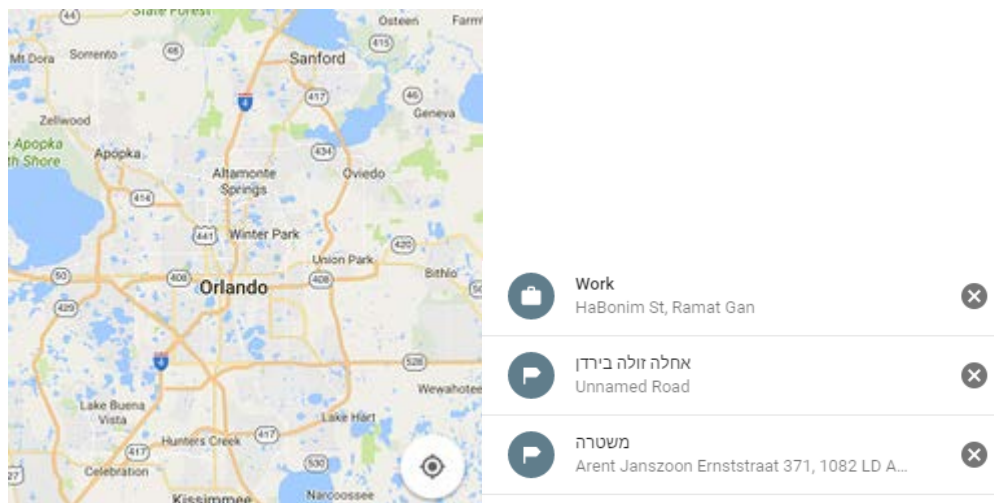
- Add a *required* email `<input>`
- Add a *range* `<input>` to let the user select his age: 18->120
  - Bonus: validate it matches his birth year

### map.html


Here we will show a **map** and allow users to manage their places.

Tips:

- Use the 'create Google Api' doc to create an API key and secure it.
- This ex involves self learning and handling new documentation that we haven't met yet.
- Self learning new technologies is a big part of being a pro programmer!



### Steps

- Show a map centered at **Eilat**, when user clicks the  button (*search for "my location png" in google*), get his current location and center the map accordingly.
  - Something like:  
`map.setCenter(new google.maps.LatLng( 45, 19 )`
- When a user clicks on the map, the user is prompted to enter a name and the clicked location is saved to a places array in the localStorage.
  - BONUS: use inputs
- Show the list and allow the user to remove a place.
- Use a place-service that manages the place entity, a place object looks like that:  
`{id: 123, lat: 32.1416, lng: 34.831213, name: 'Pukis house'}`
- Add navigation links to all pages.
- Bonus: Create more pages and try out some HTML5 features we have covered