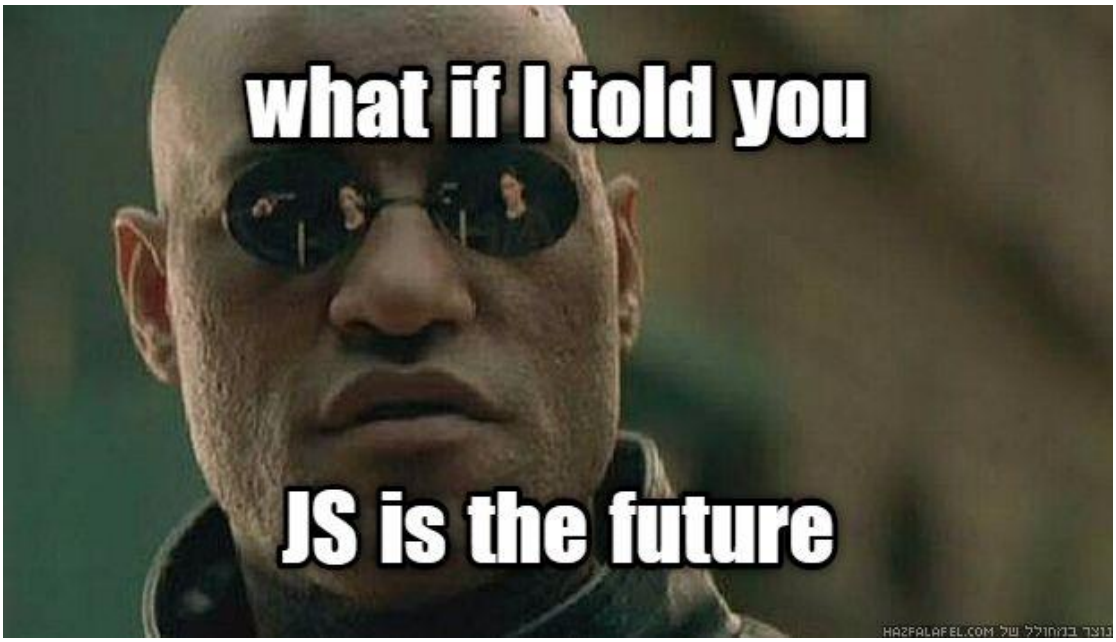
Ultimate Meme Creator

Sprint 2 Challenge

Your task is to create a Meme Generator that looks good and works well on both desktop and mobile.



Delivery Instructions

There are 3 folders to submit your code – Friday 16:00, Sunday 21:00 and Monday 21:00.

Publish and test your app on Github Pages.

Remember to reserve enough time for QA and UI improvements. Try to complete all functionality at least a few hours before the final delivery time and then concentrate on UI finalizations.

Meme Generator - App References

Play with and get to know some of the following apps on both desktop and mobile. Please learn what is it about, what to do and what not to do.

(On your app – take the freedom to create a better UI)

Main reference:

http://g.hazfalafel.com/index.php (line draggable, side-line editing)

Other references:

http://memebetter.com/generator (very basic. no line move)

https://imgflip.com/memegenerator (line draggble, side-line editing)

https://www.iloveimg.com/meme-generator (line draggable, inline editing)

https://play.google.com/store/apps/details?id=com.zombodroid.MemeGenerator&hl=en (Android. Look for more)

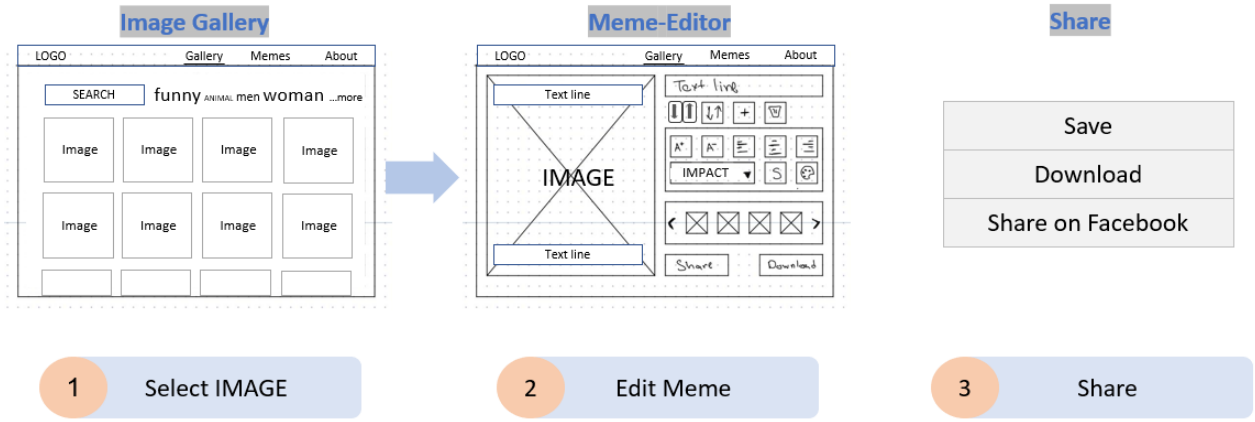
https://itunes.apple.com/us/app/meme-generator-by-zombodroid/id645831841?mt=8 (iOS. Look for more)

Product Definition - flow of the app

Flow is simple: User selects an image, adds some text and downloads the picture to his device.

The app has two main UI areas: Image-Gallery and Meme-Editor (both are implemented on the same web-page)

Once the user selected an image on the Image-Gallery, the image is presented on the Meme-Editor then the user may edit that meme and once ready - download it.



The UX (User Experience) definition of the app is given in MemeGenUX.PDF

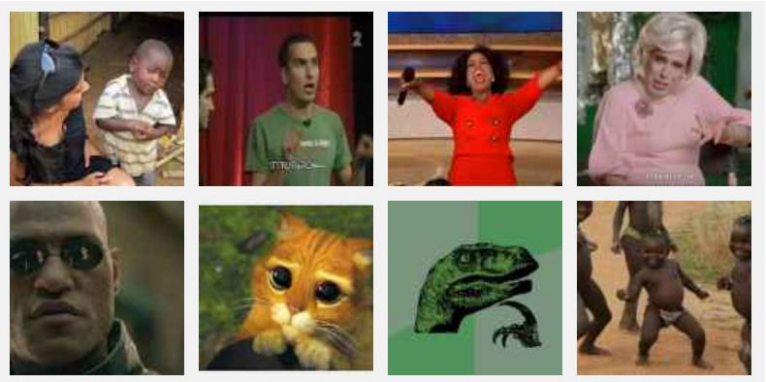
The UI (design of the app) definition is given in Appendix2 below.

MVP guidelines

Use the MVP (Minimal Viable Product) principle when you plan what to implement first. A description for this principle is given at Appendix1 below.

Functionality: Image-Gallery

1. Gallery: Show a Gallery of images



Use the square images from "meme-imgs(square)" folder
Supporting non-square images is a **Bonus** task. In case you do it – use the images from "meme-imgs(various aspect ratios)" folder

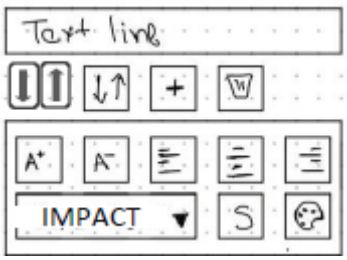
2. Search/Filter: The user should be able to search/filter the images by typing in a SEARCH-BOX letters. Relevant images shall be presented/filtered while typing. Focusing and typing in the SEARCH-BOX shall also cause presentation of an initial list of keywords (use <datalist>)
Note: This is a **Bonus** task (not a must)

3. The user may also filter the list by selecting a keyword from a list where each word size is determined by the popularity of the **keyword search** (make an initial "random" setup so it will look good from the start):

Note: this is a **Bonus** task (not a must)

Functionality: Meme-Editor

1. It is preferred that a single set of control-boxes will handle all the different lines ("control-box" is the section containing the text line and buttons).



TIP: you will need to keep the x,y of the lines

Relating the control-box to the selected line might be done by either:
   a. "Switch line" button switching focus between the lines OR by
   b. Clicking the correct line on the canvas

Mark the selected line (so the user can see which line is active)

Although we request to implement a single control-box, in case time doesn't allow, you may use several control boxes as done in some of the reference examples (such as http://g.hazfalafel.com/index.php)

2. As a default, Use the common meme font "**Impact**", white with black **stroke**.
3. Set of controls: font family, font color, font size, L-R-C alignment.
4. Up-Down alignment arrows for positioning the text line (no need if Line-Dragging is implemented)
5. "Delete-Line" and "Add-Line" Buttons.
6. "Download" Button/Link of the created Meme image
7. First two lines shall be at the TOP and BOTTOM of canvas, further lines at the center
8. Line dragging is not a must (it's a **Bonus** task) and shall be implemented only in case you implemented all above functions first.
9. Adding stickers, Saving the Created Meme to "Memes" Tab as well as Sharing to Facebook are **Bonuses**

Design and Responsivity

Both the Image-Gallery and the Meme-Editor shall look good on both Desktop and Mobile.

QA mobile UI on your mobile devices using GITHUB pages

Keep original proportion of images on both Canvas and Gallery

After main flow functionality is implemented and works fine, pay attention also to Canvas responsivity and sizing. We recommend using image aspect-ratio to calculate canvas height from its width.

Using one of the UI designs given in Appendix3 is not a complete must Nevertheless, its highly recommended - following the given designs will improve your CSS skills.

Model and Code - Recommendations

Images: img objects in your model should have the properties:

```
[{id: 1, url: 'img/popo.jpg', keywords: ['happy']}
```

Keywords examples: happy, crazy, sarcastic, sad, animal…

Model: A proposed initial data structure (managed by a meme-service):

```
var gKeywords = {'happy': 12,'funny puk': 1}

var gImgs = [{id: 1, url: 'img/popo.jpg', keywords: ['happy']}];
var gMeme = {
    selectedImgId: 5,
    selectedLineIdx: 0,

    lines: [
            {
```

```
                txt: 'I never eat Falafel',
                size: 20,
                align: 'left',
                color: 'red'
            }
    ]
}
```

Search/Filer: use the <input> list attribute to show some pre-made options

Recommended Order of implementation

Here is a recommended order of the first few development phases

1. Phase1 – main flow with no design (shall take 1-6 hours):
    a. Setup git and make sure you can commit and push to the repository.
    b. Design an initial home page (index.html, main.js, CSS files)
    c. Commit and Push
    d. Create gMeme[] as described above with a single txt line.
    e. Create a Canvas with a single image – the image shall be taken from gMeme (managed by a memeService)
    f. Draw a text line on it with IMPACT font at the top of the image. The text shall be taken from gMeme
    g. Add text input to the HTML and dynamically take the text line value from the input to gMeme and from it to the Canvas
    h. Make a simple image-gallery with 2 images. Click an image to update gMeme and present it onto the Canvas. Note that to start with – make the Editor located above the Image-Gallery.
    i. Make sure you can access your project in gitPages

2. Phase2 – Basic line operations:
    a. Add the button "increase/decrease" font
    Implement text size increase / decrease
    b. Add "up/down" button
    Implement moving lines up/down

3. Phase3 – switch between two lines:
    a. Add the button "switch line"
    b. Add (to gMeme) a second line and implement switching between the lines (focus) using the button

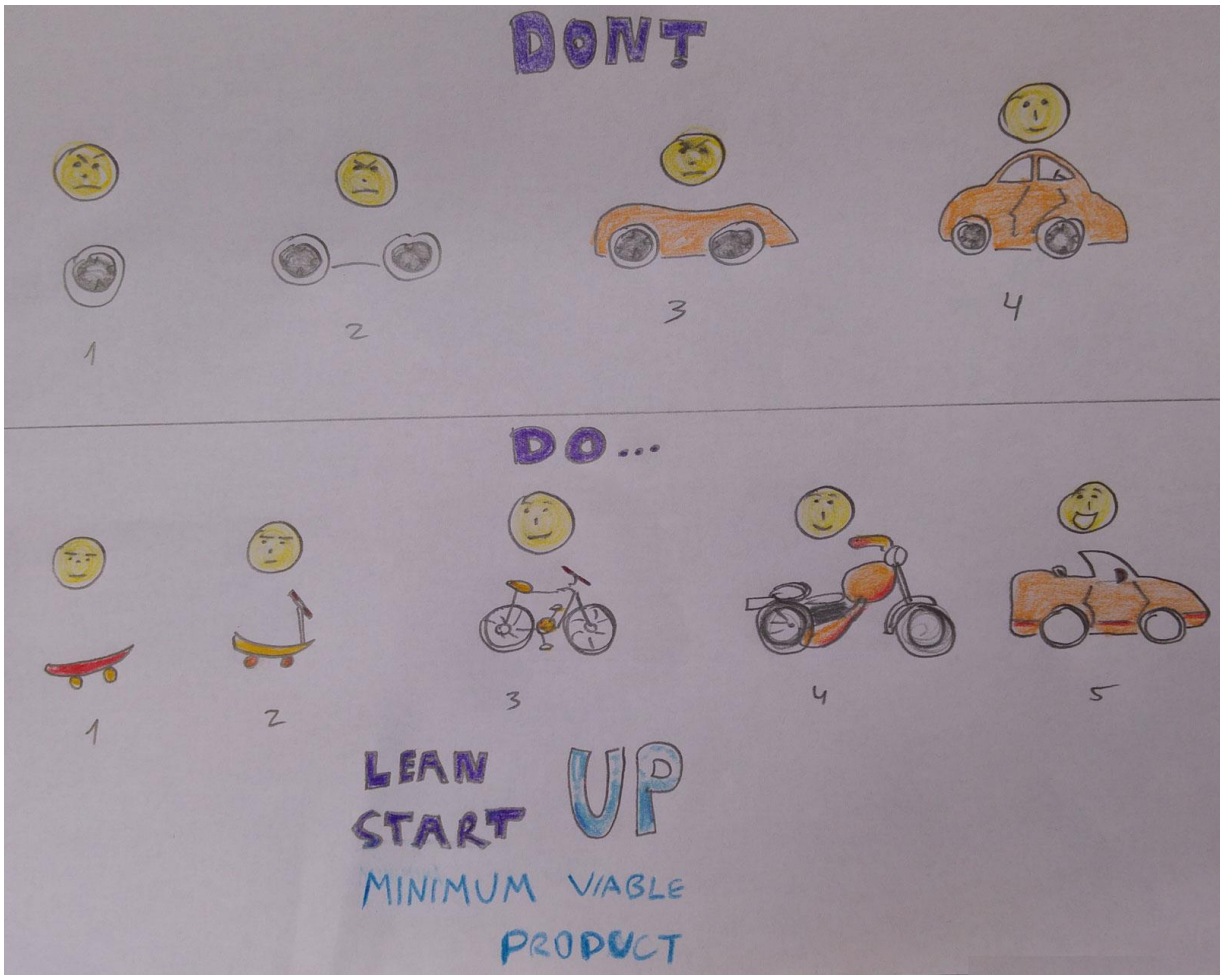
4. Phase4 - Basic CSS:
    a. Build the page layout with header (NavBar), footer and container for the center
    b. Locate the two images at the center of the Image-Gallery page. Make sure both Images are the same width in CSS

c. Give the Meme-Editor the right proportion between the space allocated for the canvas and the line-controls
d. Implement a basic CSS for mobile (both gallery and Editor)
e. Implement that the Editor is hidden to start with and revealed when an image is clicked

Bonuses and Advanced features (appear grey at the sections above)

1. Save created Memes to the "Memes" TAB (using local storage)
2. Support "Drag&Drop" of the text lines
3. Add "search" to the Image-Gallery Page
4. Add stickers (emojies, sunglasses, hats, etc)
5. Drag and drop stickers
6. Share on Facebook (use the sample code provided)
7. Support using various aspect-ratio of images
8. Allow using an image from your computer
9. Add "search by keywords" to Image-Gallery Page
10. Inline (on Canvas) text editing
11. Resize sticker
12. Website theme: celeb-meme, politic-meme, ani-meme, kid-meme, Mondial-meme
13. Use the new Web Share API to share your meme
14. i18n for Hebrew

Appendix1 – MVP – Minimum Viable Product

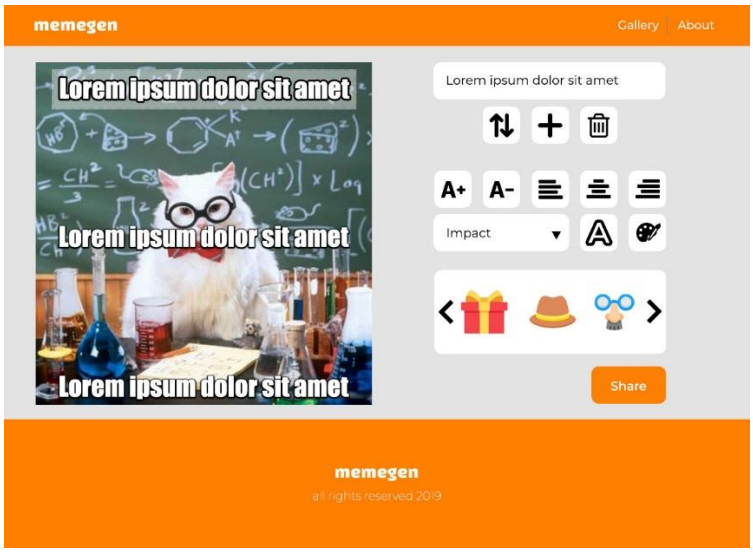


Appendix2: UI

Please note that the designers may have not completed the entire design. They are also not available, so you will need to take some UI decisions, please make good ones.

Select one of the following 3 UI designs:

1. **Meme1 - Sergei**

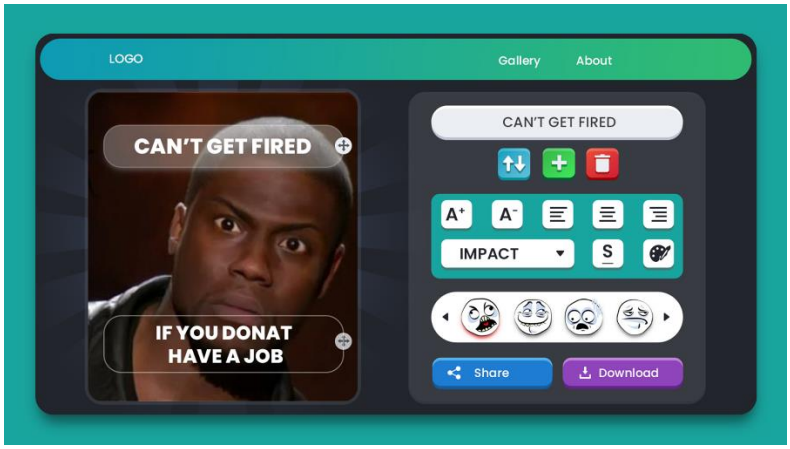


PSD file: Located in the “DESIGN/Meme1 - Sergi/”

Zeplin: https://app.zeplin.io/project/5daf01e024578154edc2cd12/dashboard

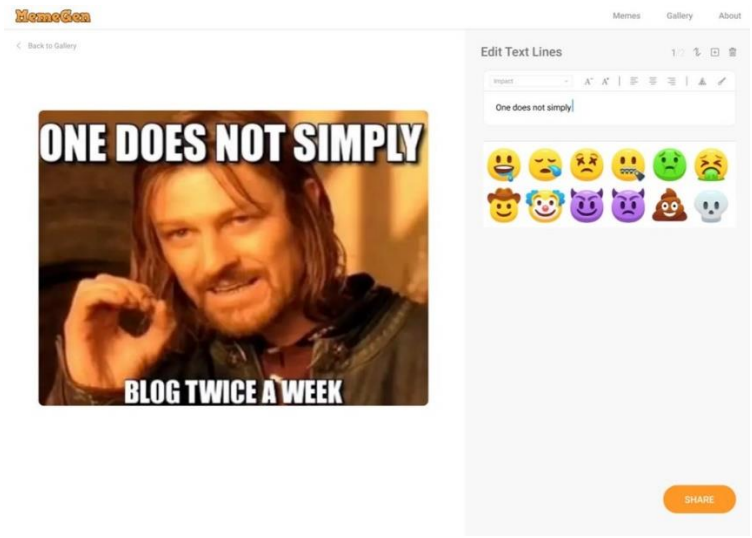
Zeplin intro: https://www.youtube.com/watch?v=agRUrGCTuFs

2. **Meme2 – Game Style**



PSD file: Located in “DESIGN/Meme2 – Game Style/” folder

3. **Meme3 - Alex**:



Figma: https://www.figma.com/file/LTqroiQHhQwDMU8VD5bjI6/MemeGen?node-id=0%3A1