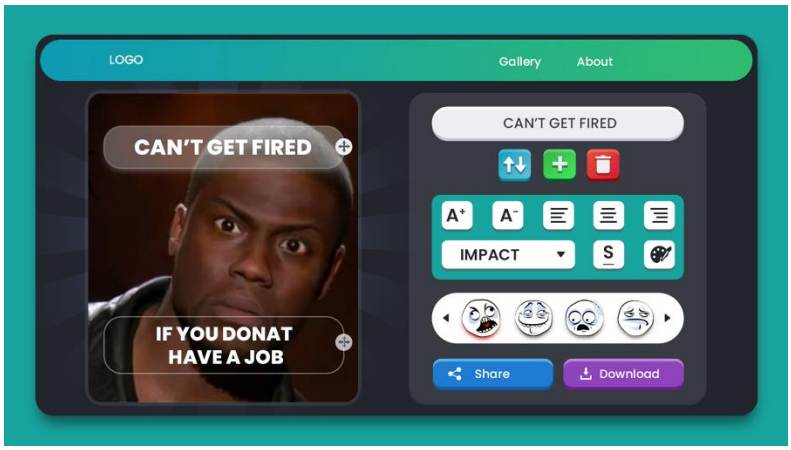
Ultimate Meme Generator

Sprint 2 Challenge

Lets code an awesome Meme Generator that looks good and works great on both desktop and mobile.



Delivery Instructions

There are 3 folders to submit your code – Wednesday 20:00, Thursday 20:00 and Saturday 22:00.

Publish and test your app on Github Pages.

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 some of the following apps on both desktop and mobile and pay attention to details.

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)

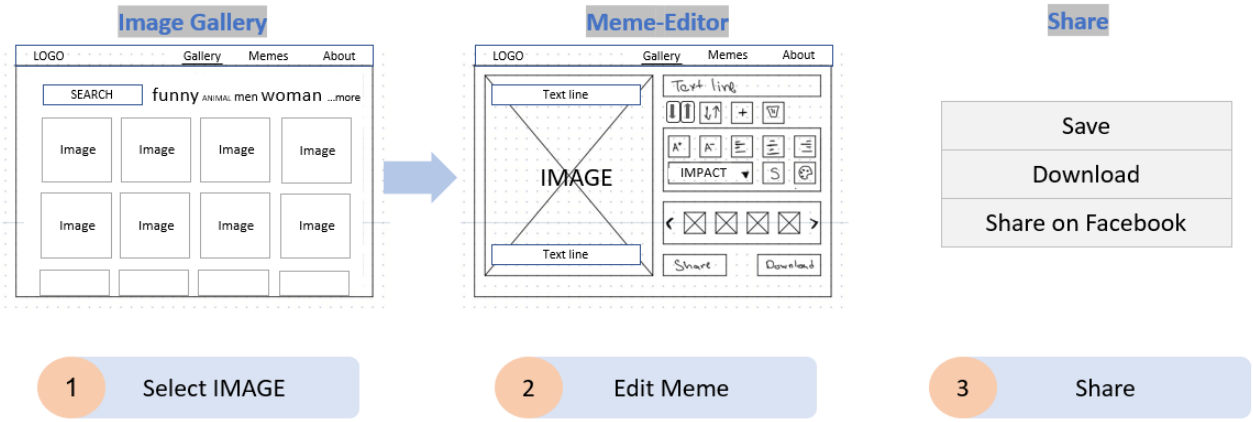
https://itunes.apple.com/us/app/meme-generator-by-zombodroid/id645831841?mt=8 (iOS. Look for more)

Product Definition - flow of the app

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

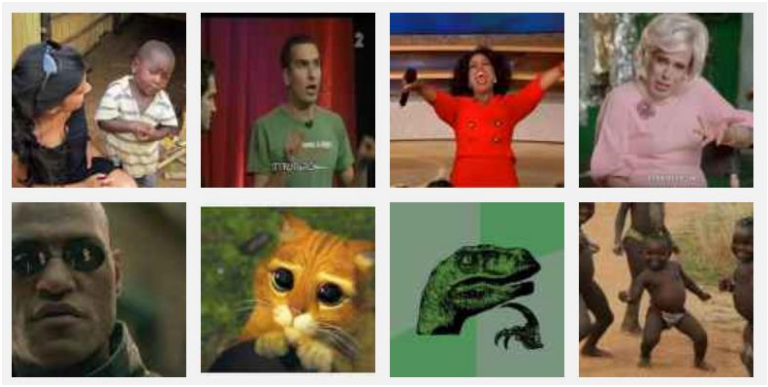
The UI (design of the app) definition is given in Appendix2 below.

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

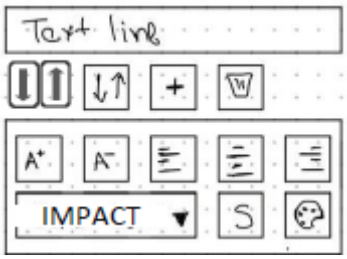
1. Gallery: Show a Gallery of images



   Start with square images.

Functionality: Meme-Editor

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



   The use can switch the selected line by clicking the switch-line button, he can also click the text on the canvas.
   Mark the selected line (so the user can see which line is selected)

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 (you may hide them later if you implement line dragging).
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

Design and Responsivity

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

Use pure CSS – do not use CSS Libraries and/or jQuery.

Test your app on your mobile devices by accessing the app hosted in Github pages

Keep correct 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.
Tip: calculate the image aspect-ratio and use it to calculate the canvas height from its width.

Select one of the UI designs given in Appendix3, you may also allow yourself some creativity but focus on implementing the app.

Model and Code - Recommendations

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

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

```
var gKeywordSearchCountMap = {'funny': 12,'cat': 16, 'baby': 2}

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

    lines: [
            {
                txt: 'I sometimes eat Falafel',
                size: 20,
                align: 'left',
                color: 'red'
```

```
            }
    ]
}
```

Recommended Order of implementation

Here is a recommended order of the first few development phases

Phase1 (~4-8 hours)

1. Build an initial home page (index.html, main.js,main.css)
    a. Create an empty section for the gallery and for the editor
2. Commit and Push to github, setup Github pages
3. Create a memeController, Code a function renderMeme() that renders an image on the canvas and a line of text on top
4. Add a memeService with a gMeme variable and a function getMeme(), the function renderMeme() can now render that meme
5. Add a text input and when it changed by the user –
    a. update the gMeme using the memeService function setLineTxt()
    b. then renderMeme()
6. Create a galleryController, with a renderGallery presenting two images.
7. onImgSelect – memeService.setImg() and then renderMeme()

8. Phase2 – Basic line operations:
    a. Add a color picker button
    b. Add the button "increase/decrease" font

9. 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
10. Phase4 - Basic CSS:
    a. Build the page layout
    b. Make it look good on mobile (both gallery and Editor)

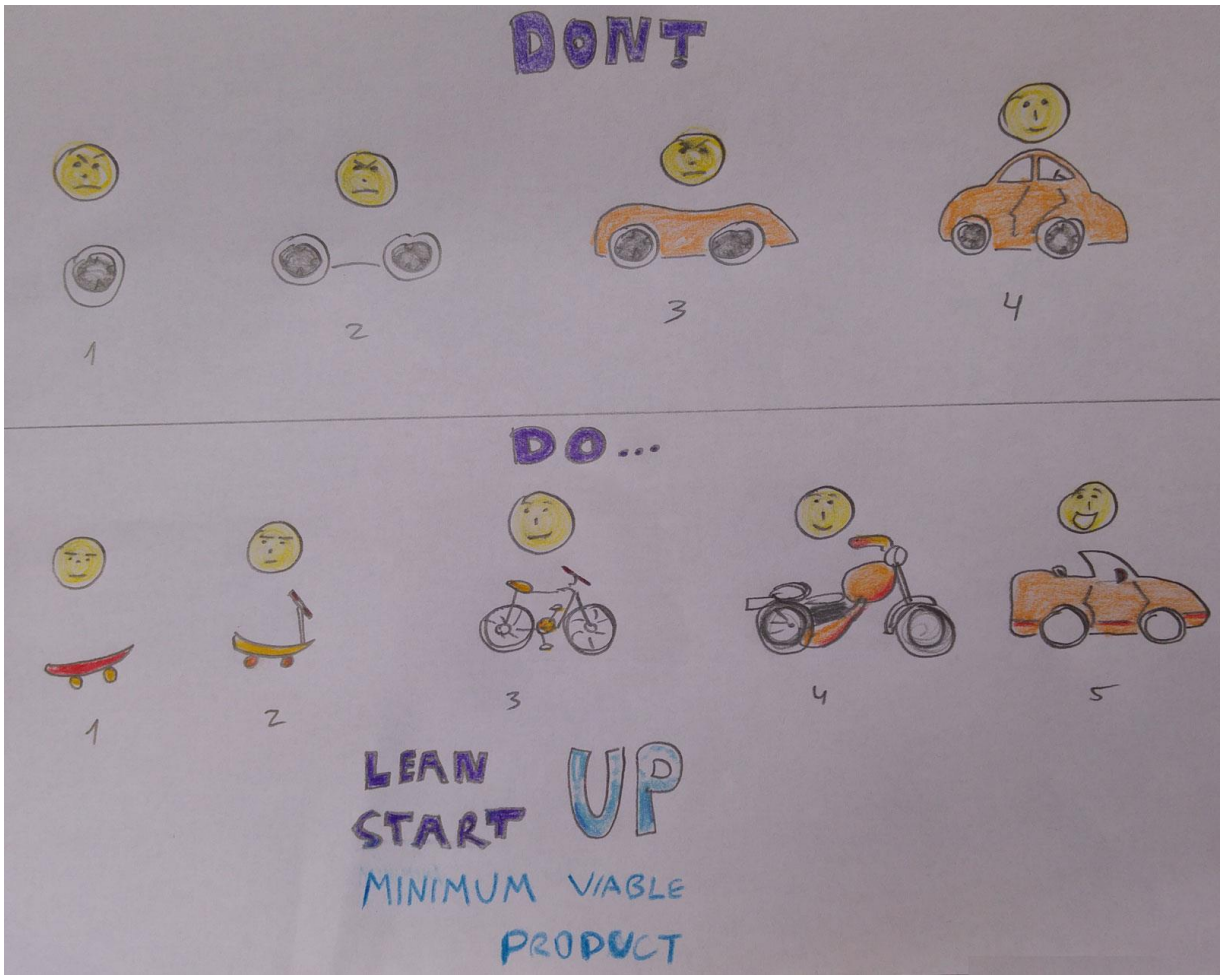
Bonuses and Advanced features

A. Save created Memes to the "Memes" TAB (using local storage)
B. Image gallery filter (use <datalist>)
C. Add stickers (can be images or emojis characters such as: 😜😎)
D. Support "Drag&Drop"
E. Share on Facebook (use the sample code provided)
F. Support using various aspect-ratio of images, use the images from "meme-imgs(various aspect ratios)" folder
G. Allow using an image from your computer
H. Add "search by keywords" to Image-Gallery Page.
   Each word size is determined by the popularity of the keyword search - so

each click on a keyword makes that keyword bigger
TIP: use an initial deo data so it will look good from the beginning

I. Inline (on Canvas) text editing
J. Resize / Rotate a line
K. Website theme: celeb-meme, politic-meme, ani-meme, kid-meme, Mondial-meme
L. Use the new Web Share API to share your meme
M. i18n for Hebrew

Appendix1 – MVP – Minimum Viable Product

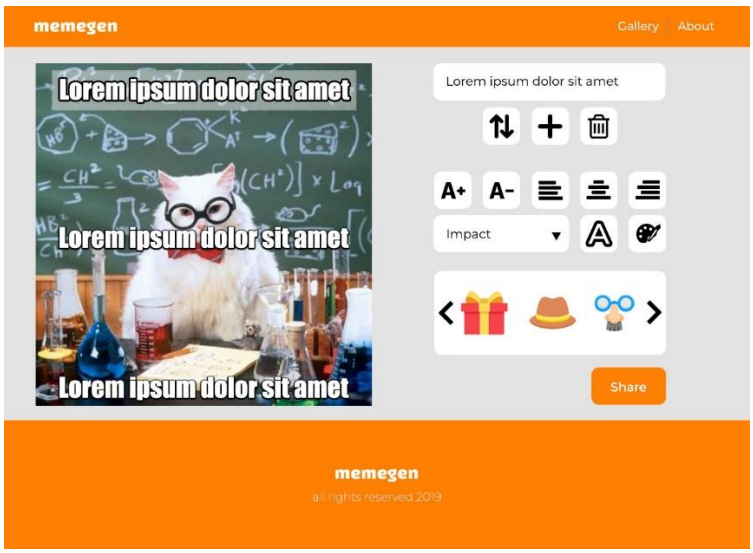


Appendix2: UI

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:

Design1 - Sergei

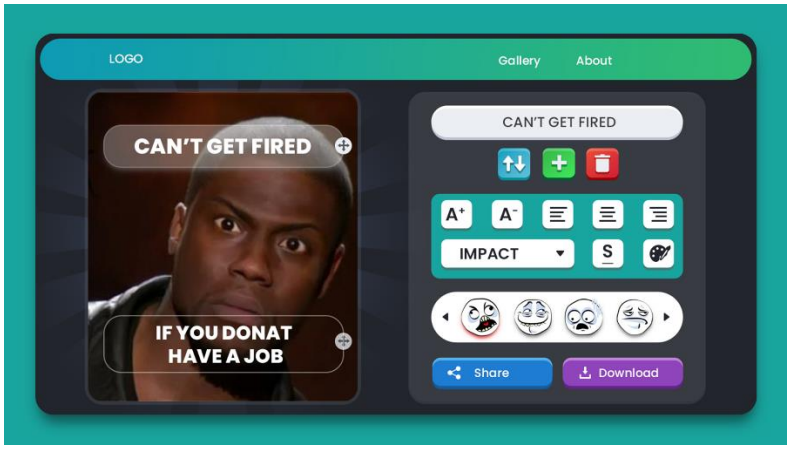


PSD file: Located in the “DESIGN/Meme1 - Sergi/”

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

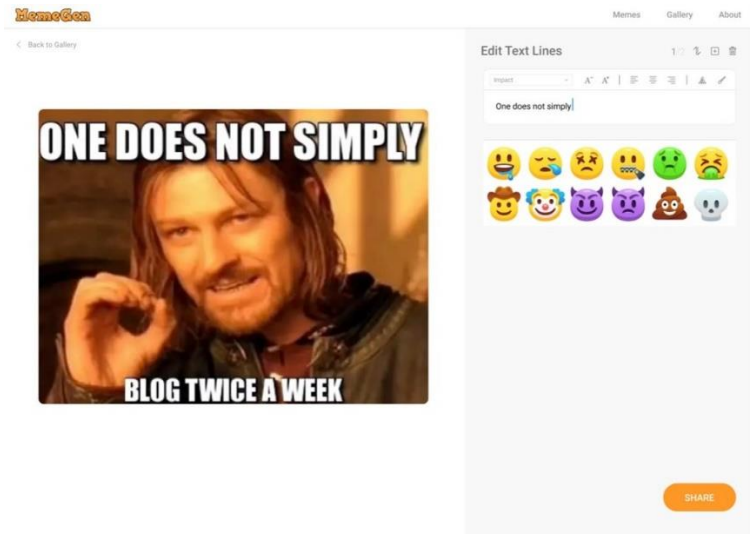
Zeplin intro: https://www.youtube.com/watch?v=agRUrGCTuFs

Design2 – Game Style



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

Design3 - Alex



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