**Experiment-1**

WEEK 01

1. **Discuss Success and Failure Stories**

**How can you stay motivated while learning to code?**

Studying web development is a path with many ups and downs. You might have one day where you figure out a tough problem and feel absolutely amazing. But then the very next day, you get stuck on a seemingly easy problem for hours. You end up feeling completely defeated.

**“It’s often hard to see the light at the end of the tunnel. But when you’re in those tough, dark spots, try to find the motivation to hang in there”.**

Remind yourself of your ultimate goal in learning to code. Maybe it’s to get a better job, to provide for your family, or to find a more purposeful form of work.

You can also find encouragement from others who are in the same boat as you. Knowing that there are people who struggle with the same issues can be hugely encouraging. And seeing them fight their way through could be all the motivation that you need to continue.

Reading these stories, I hope that you find inspiration and encouragement.

**Story # 1:**

[How I went from selling food in the street to working for top firms in tech](https://www.freecodecamp.org/news/become-how-i-went-from-selling-food-in-the-street-to-working-for-top-firms-in-tech-6aa61a2d0629/)Alvaro Videla taught himself PHP via the local internet cafe and some books sent to him by a relative. After endless nights studying and practicing, he landed a job with a tech company.If you read just one story from this list, choose this one. It’s a well-written narrative of what’s possible when you really put your mind to something.

**Story # 2:**

**Gwendolyn up late coding one night.**

Gwendolyn Faraday was working full-time at a restaurant and helping her grandmother. In 2015, after dabbling casually in coding tutorials, she decided to go all-in.

She enrolled in Treehouse and became a regular in the freeCodeCamp community. She was constantly learning and putting herself out there. And one of the keys that helped her land a job was expanding her in-person and online network.

**Lessons you can take away from these stories.**

In reading through these stories, I found quite a few common threads. These are principles that many of them put into practice in order to reach their goal.

#### **Commit to learning.**

All these people committed to learning to code, sacrificing sleep and their social lives. They spent hours per day (or night!) dedicated to learning and practicing.

1) Elvis learned HTML and CSS every day after school from W3Schools.

2) Sergei continued to learn more web development on his own even after getting a job.

3) Gwendolyn decided that 2015 would be her year to learn to code, and set aside several hours per day to learn.

4) Danny committed to studying coding at least one hour per day.

5) Ken started work at 6 a.m. and studied PHP and Laravel in the evenings and on weekends.

#### **Follow a structure:** Whether it’s a bootcamp, a book, or an online course, try to follow a preexisting structure. Doing so will make your progress easier to track and will make the end goal more feasible.

1) Sergei created his own learning path based on the most common job listing requirements.

2) Danny latched onto the #100DaysOfCode challenge.

3) Alvaro had a vision for a local maps app and gave himself a deadline to complete it by.

4) Gwendolyn finished a Treehouse track and the front-end portion of freeCodeCamp.

#### **Build things.**

The best way to transform your class knowledge into practical knowledge is to build things. They might be terrible, and full of spaghetti code. But what matters is that you’re creating stuff in the real world and learning from them.

1) Kep built PHP and Laravel apps as he learned.

2) Elvis built a social media app, a group SMS app, and Android apps.

3) Alvaro created a local maps application that he used in his job interview.

#### **Be bold:**

You will never feel truly ready for the next step. But that doesn’t mean that you aren’t ready. Don’t be afraid to take risks that could pay off by advancing your career.

1. Both Ken and Rich reached out to companies for informational interviews, and ended up getting job offers.

**.Importance of ethics for a software engineer**

**Abstract**

In the field of Software Engineering, ethics are essential for an employee. A discussion of importance of ethics in the workplace and the practice of ethics as professionals are highlighted.

**Introduction**

In the cooperate world, ethics play a vital role among the employee and employer. A code of ethic or guideline describes the ethical and professional responsibility of software engineer against peer and legal bodies that can measure their behaviour.

Both employee and employer require set of guidelines to follow during their worktime. It helps both parties to accomplish their tasks and follow a professional behaviour in the workplace. Ethics are important as it helps to decide the course of action.

As for a software engineer, it is truly required to follow ethics to overcome different dilemmas and be a professional. Software engineer programmes the software to satisfy the customer’s requirement but he/she has to be ethical when designing it.

They must be more concerned about the ethical issues; the impact of unethical behaviour and outcome of the project could be harmful on lives and expensive in terms of capital. The Therac-25 system – it took 4 lives and injured 2 lives with severe wounds in 1986 and 1987 according to Fabio (2015) and Ariane 5 -Flight 501 explosion caused by a bug in the inertial navigation system noted by Sommerville (2015) would be examples in the field. IEEE Computer society and ACM promote ethics as an important factor for a software engineer. (Sommerville, 2015)

**Overview**

As employees,

* we tend to think that some activities have to be done by other staff members which is their responsibility – not my problem.
* We tend to swipe away the errors or ignore them with various functionality, but end user uses the product, it could be more harmful – swipe it under the rug.
* Informing clients and management that the program or software is operational while it is under construction
* Providing the management or employer and client that product is deliverable but still serious problems are detected in testing pace.

These are some common ethical dilemmas that an employee or employees to practice overcoming the critical situations and accomplish the project for customers.

**Discussion**

The ethical dilemmas such as not my problem and swipe it under the rug are caused by employees and employers when they are under pressure by top level management or in hurry to complete it. When they are unable to reach the milestones and deadline is near, they tend to cover their work with functionality and pass it next staff member pointing out rest of the work as their responsibility.. As employees – they could ask themselves the reason for why code of ethics? Simple answered could be that their profession which has huge impact on human lives and other things in society.

Programmers have to understand the risk of their project and failure to do so could lead bad accidents such as

* Ariane 5, Flight 501
* Therac 25
* The London Ambulance Service
* Who counts the votes?

These projects are bought a massive damage on human lives as well as capital noted by Vliet (2008). And it highlights the importance of ethics, employee could feel uncomfortable with some guidelines as it might speak high values, but will it be worthy to take a real risk of the outcome of a project. Employees must mind the risk of the project and importance of ethics. Employees (software engineer) could question themselves in three ways,

* How will my software affect to the society? (eliminate those risks).
* Are there any possible outcomes for bad practices? (reduce the possible outcomes)

this would help them to uncover their software’s bad outcomes.

1. **A CASE STUDY**

**Introduction**

Case studies are and will always be an integral part of the business world hence the need for winning case study templates. Whether it is to analyze internal problems and find solutions or convince a client of your capabilities; case study presentations are always involved in the process.

In the following sections, you will understand what goes into making an impactful case study presentation.

**WHAT IS A CASE STUDY?**

A **case study**is your concise and well-reasoned argument, that talks about the **why, how, where**, and **who**, that convinces the audience for your research. Think of it as a solution, to a well-defined question asked to you. Should the company expand in this market? Should we launch this product? What is the reason for our reduced sales? Why are you the best person for the client’s project offering? All these questions are backed by immaculate case study presentations.

This begs the question, when should one do a case study analysis. Some of the scenarios where case studies become necessary are –

1. Convincing clients of the benefits of your services and capabilities.
2. Analyzing and presenting solutions to internal issues for a company.
3. Offering real-life examples to back your arguments.
4. Systematic discussion on the viability of an idea, with respect to the effect on the firm (client or self).

The task here is not only to find a solution but also to influence the audience with your findings. Organizing a lot of information in a succinct and engaging manner for your clients and stakeholders is important. Hence, is a need for an excellent case study presentation template.

Let’s see how to make a well-structured case study template, to wow your audience.

**FEATURES OF EFFECTIVE CASE STUDY DESIGNS**

A good case study design should/should be:

1. Brief and highlight the essentials
2. Clear and concise
3. Describe your vision clearly
4. Demonstrate the value and benefit the project brings to business or client
5. Consistent in style

2. Organize and play games to understand the agile process like, morning wake up game

#### [**“Wake up in the morning” Game – A quick fun way to understand the basic concept of Iterative Incremental Development**](https://www.agilesparks.com/wake-up-in-the-morning-game/)

[Agile Mindset](https://www.agilesparks.com/category/agile-mindset/), [Guidance](https://www.agilesparks.com/category/guidance/), [Kanban](https://www.agilesparks.com/category/kanban/), [Product Management](https://www.agilesparks.com/category/product-management/), [Product Ownership](https://www.agilesparks.com/category/product-ownership/), [Scrum](https://www.agilesparks.com/category/scrum-2/), [Thought Leadership](https://www.agilesparks.com/category/thought-leadership/)

A fundamental aspect of Agility is incremental and iterative development. It’s so basic that when introducing Agile you usually mention this idea in the first 5 minutes. The core understanding that moving to small batches significantly improves speed, quality and risk management, helps you move from an all or nothing approach to a world of options.

The challenge we are facing is how should we break down the scope?

How should we define these small batches and construct them in the right manner?

This challenge is surfacing with each new initiative. When I support and coach an organization moving to Agile, I find that people adopt the terms of User Stories and Epics quite immediately, but the idea of incremental and iterative development is not really understood.

Lately I have been experimenting the use of the  “Morning Routine Story Mapping” exercise Jeff created (which is also detailed in his book [User Story Mapping: Discover the Whole Story, Build the Right Product](https://www.amazon.com/User-Story-Mapping-Discover-Product/dp/1491904909)  and also quickly outlined [here](https://agileworldblog.wordpress.com/2015/09/04/user-story-mapping-for-fun-and-profit-lynne-gurney-johnson-charlotte-philippe/) as the [“getting out of bed” and out the door game”](https://agileworldblog.wordpress.com/2015/09/04/user-story-mapping-for-fun-and-profit-lynne-gurney-johnson-charlotte-philippe/)). the game is a major success for Product Ownership workshops when introducing the story mapping technique.

**Step 1 – Individually list the morning activities (5 minutes)**

**Step 2 – Collaboratively grouping the activities (5 minutes)**

**Step 3 – Order the group of activities by time  (3 minutes)**  
**Step 4 – Order the activities by criticality  (5 minutes)**

**Step 5 – Drama! (5 minutes)**

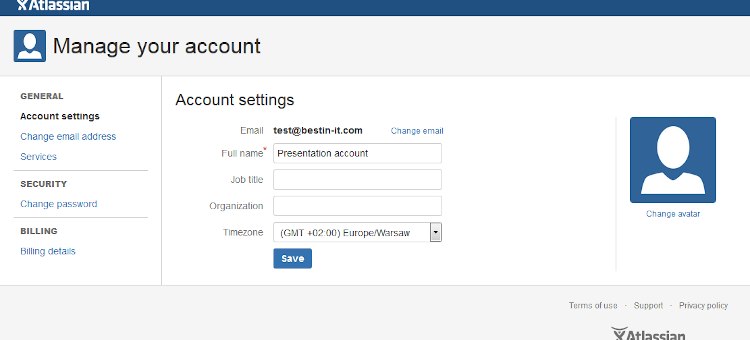
**Final Briefing:**

The exercise demonstrates the following important concepts:

* With the constraint of time our aim is still to realise the full “value” of getting on time to the office. In the process of minimizing the activities, we removed many of them in each step and left the process very thin and lean but still end-to-end.
* Since we have a constraint of time, as we eventually want to be fast which means minimize the time to reach value. in many cases we will go too deep in a single step and not realise the full end-to-end value. in this exercise we demonstrate how it should be done across the map and how in every increment we build we have the full end-to-end.
* We don’t invest equally in each step – in some steps we left only one activity and in some we left more, depending on the step. Some steps where even entirely removed.
* Choosing the depth of each step is easier when the full picture is available since the alternatives are visible.

Focusing on a single activity but in the context of end-to-end value helps development teams better understand the scope of the requirement. For example, preparing breakfast in the context of getting out of the house in 15 minutes is totally different from preparing breakfast for the family on a vacation morning.

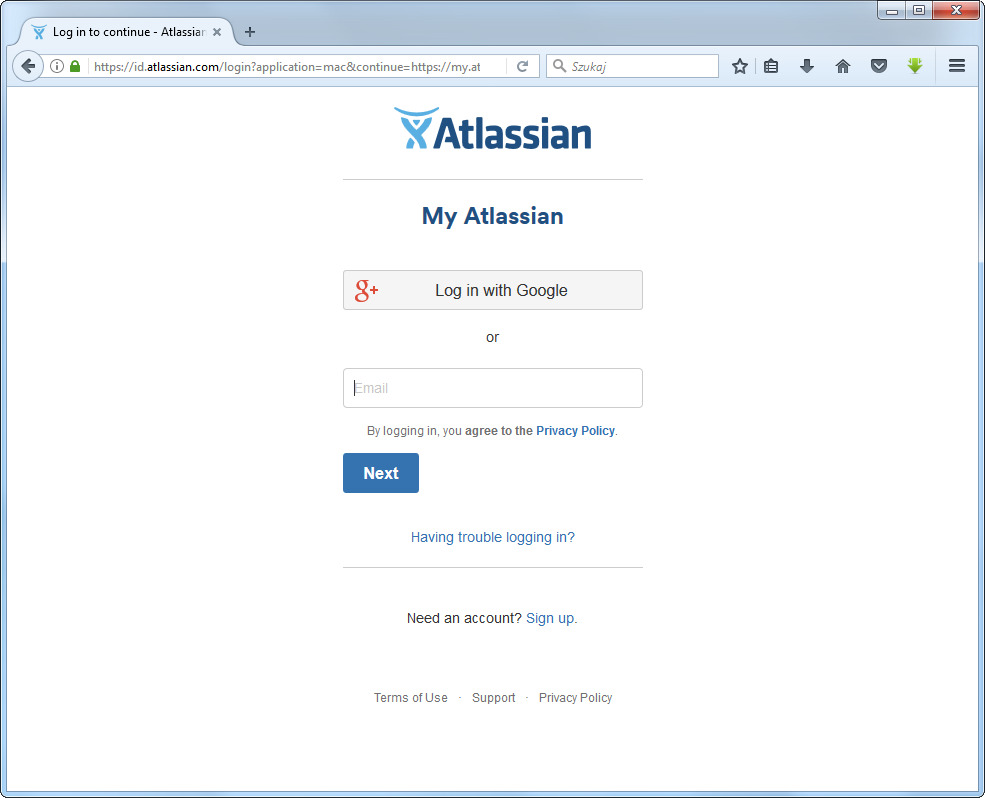
1. **Create JIRA account and learn interface**



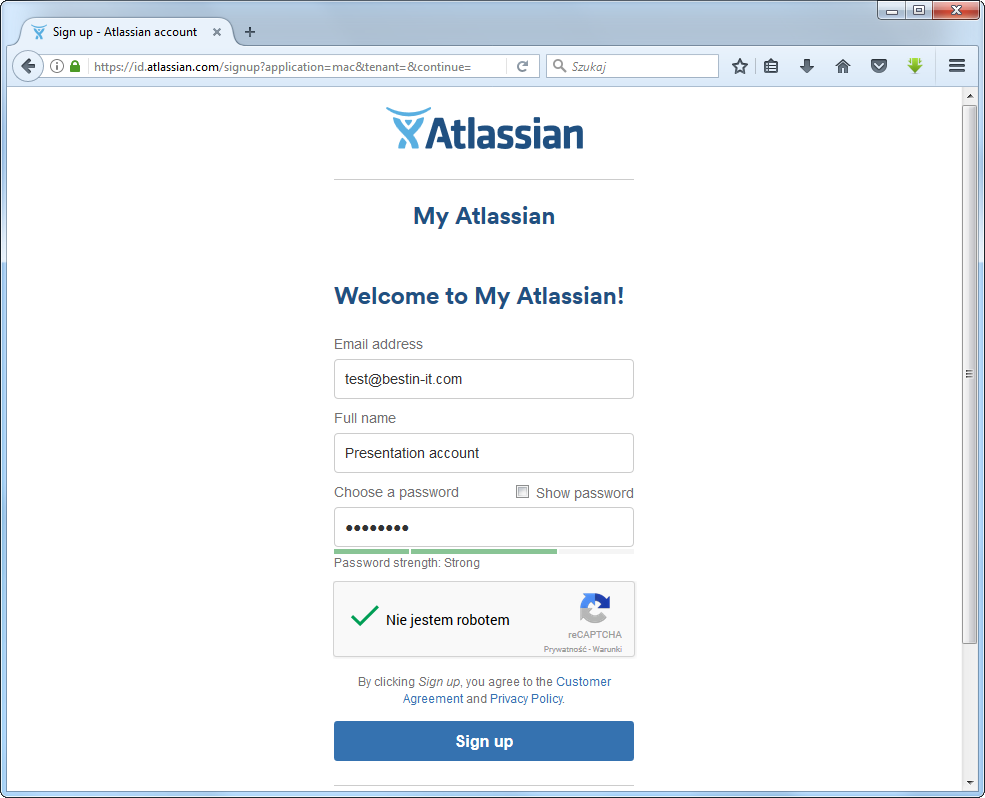
### **Creating an account in atlassian to install Jira**

In order to install server version of Jira we need to start the process: creating an account in atlassian website. Atlassian keeps all information about licensing in their website id.atlassian.com. If you would like to buy another products like for example Bamboo or Bitbucket you need to have an online account to be verified.

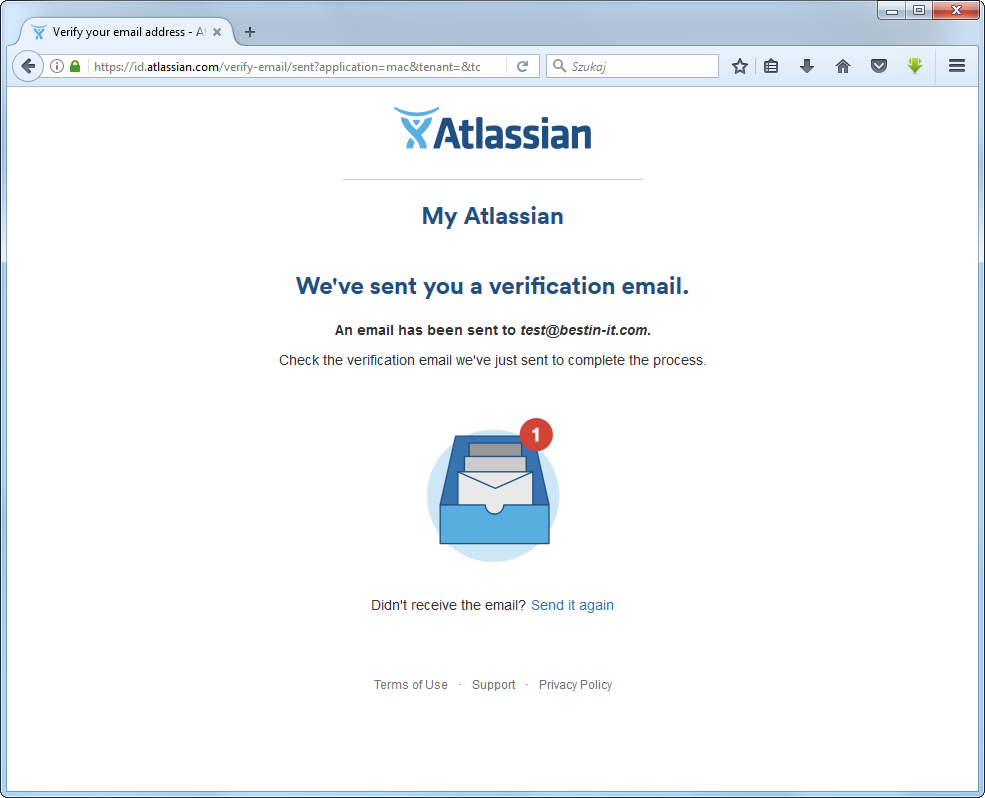
1.So let’s go to http://id.atlassian.com to start registering which is fully free process.



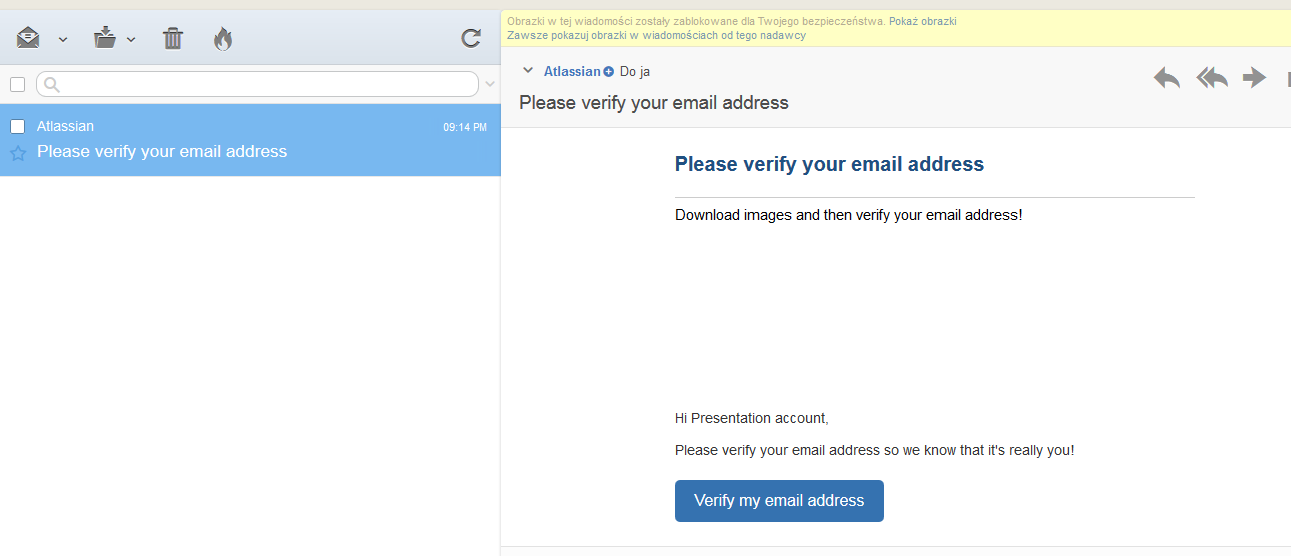
2.We need to click ‘Sign up’ and then we are asking for email, full name and password. From experience I can only advice you to use as a username your email.



3.When you type all data correctly you will be informed that an activation email was sent to you.

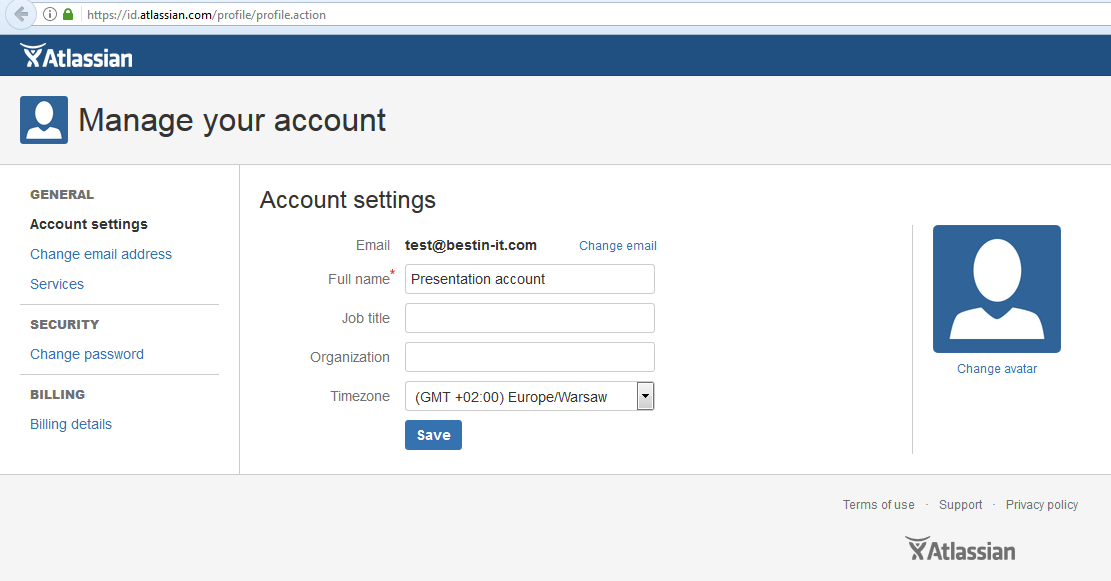


1. In the ‘Please verify your email address’ email you will be asked to verify your account. Just click on ‘Verify my email address’.

5.

5.When you do all the steps correctly you will be redirected to the online panel with information about your profile settings.

Having this account, you can manage all services in Atlassian like for example Jira. This account is also very usefull place where you can check all your billings. This is important when you have different services run in Atlassian and you don’t need to log into different accounts. You also will be using this account to make any support requests to Atlassian team. From my experience I can only tell you that their support is very quick and they also solve problems very quickly



Experiment-4

Not found

**1.Conduct warmup activities to ignite Design thinking :**

Warm-ups can be described as exercises one normally runs right before the main proceedings to help participants relax and ease people into a group activity or learning situation. Warm-ups go very well with design thinking because they support many of its attributes, such as being curious and having an open mindset as well as being mindful of and collaborating with other people.Consequently, a well-chosen warm-up can add real value to a design thinking workshop or project, but then, a poorly chosen warm-up can also have the opposite effect, making people feel nervous, uncomfortable and confused. So, when choosing your warm-up, choose it purposefully! Here are some pointers to bare in mind:

* Firstly, warm-ups are not per se part of design thinking, they just have proven to be a useful way of promoting team work and supporting certain work attitudes.
* Warm-ups as well as other methods and exercises should generally be selected to suit the team, so you should know your audience and the people you are working with.
* While it’s important to be mindful of the people, it is also vital to read the current mood and situation and select the warm-up accordingly — it should fit to the given circumstances.
* Let the participants understand that you don’t just want to do a ‘warm-up’ with them now. Communicate the goal and reflect on it afterwards if necessary. Especially when using an ‘educative’ warm-up, e.g. ‘Marshmallow Challenge’ before prototyping, you should debrief it — active reflection increases the likelihood of understanding and learning.
* For the conduct of the warm-up, give clear instructions and know when a short demo might be necessary for your audience to better understand the activity.
* Lastly, I would like to add that you as a facilitator should love and understand the warm-up you’re choosing and get excited when using it. Only then will the spark be transmitted to the participants.

**Below, you find some examples of what for and when you can use a warm-up:**

• Create a positive group atmosphere

• Help people to get to know one another (better)

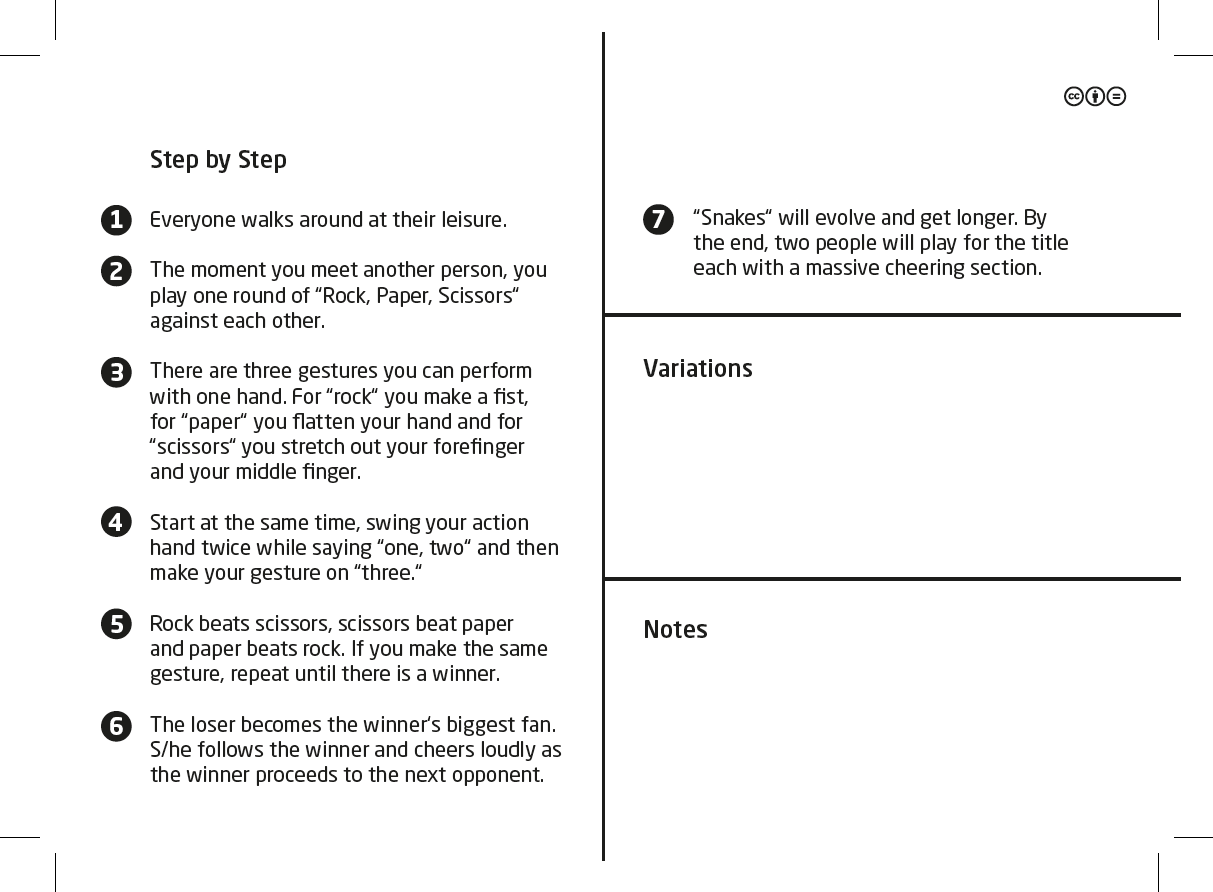
• Break down social barriers

• Reduce pressure

• Energize

• Distract the group temporarily to better focus afterwards

• Prepare the team for a certain mode of working / phase / mindset.



**6.3 Configure JIRA for the managing the project to solve the identified problem**

To configure a project:

1. Choose Settings >**Projects**.
2. Choose the project you want to configure.
3. Go to your project and click **Project settings**.
4. Use the links on the sidebar navigate between the different project settings. Read the sections below for a description of each setting.

**Project details**

Click **Details** in the **Project settings** sidebar, and edit the project details as desired. Once you've completed your edits, don't forget to click the **Save** button.

**Issue types**

Jira enables you to keep track of different types of things — bugs, tasks, helpdesk tickets, etc — by using different *issue types*. You can also configure each issue type to act differently, e.g. to follow a different process flow or track different pieces of information.

Click either **Issue Types** in the left menu or one of the issue types under it, e.g. **Bug**, **Task**, **Story**, etc:

* **Issue Types**: Click this to configure which issue types apply to this project. You can also configure the workflow, fields and screens for the issue type in the project, but it is easier to do this by clicking one of the issue types.
* **One of the issue types (e.g. Bug, Task, Story)**: Click this to configure the workflow/screen for the issue type in the project. The workflow screen (**Workflow** tab) shows the workflow designer. The screen (**View** tab) shows the screen designer.

Identify problem and prepare epics and user stories

**6.3 Configure JIRA for the managing the project to solve the identified problem**

To configure a project:

1. Choose Settings >**Projects**.
2. Choose the project you want to configure.
3. Go to your project and click **Project settings**.
4. Use the links on the sidebar navigate between the different project settings. Read the sections below for a description of each setting.

**Project details**

Click **Details** in the **Project settings** sidebar, and edit the project details as desired. Once you've completed your edits, don't forget to click the **Save** button.

**Issue types**

Jira enables you to keep track of different types of things — bugs, tasks, helpdesk tickets, etc — by using different *issue types*. You can also configure each issue type to act differently, e.g. to follow a different process flow or track different pieces of information.

Click either **Issue Types** in the left menu or one of the issue types under it, e.g. **Bug**, **Task**, **Story**, etc:

* **Issue Types**: Click this to configure which issue types apply to this project. You can also configure the workflow, fields and screens for the issue type in the project, but it is easier to do this by clicking one of the issue types.
* **One of the issue types (e.g. Bug, Task, Story)**: Click this to configure the workflow/screen for the issue type in the project. The workflow screen (**Workflow** tab) shows the workflow designer. The screen (**View** tab) shows the screen designer.

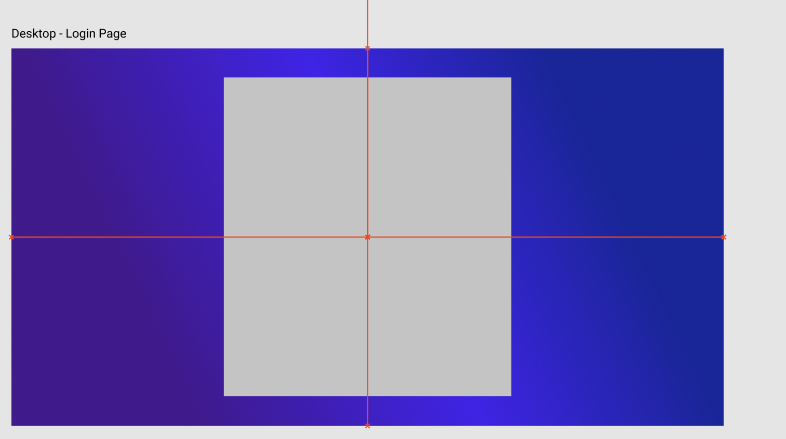
**8.Create sitemap and wireframe for above created user stories.**

Design a Login Page for Desktop

So for the login page, I want the basics. I want two input boxes and a submit button. I want the logo somewhere in there. I want a welcome message and a prompt for the user to log in. Also, maybe “Forgot Password?” and “Sign Up” anchor links, since every website has that.

For the actual design, I think I want the whole login form contained within a container centered in the middle. And I want that with rounded edges.

With design tools like Figma, we can just design it here. So let’s get started.



Here, I created a new rectangle that is going to house my login form. Using those red lines — which Figma offers as a positioning tool — I am centering the rectangle on the page.



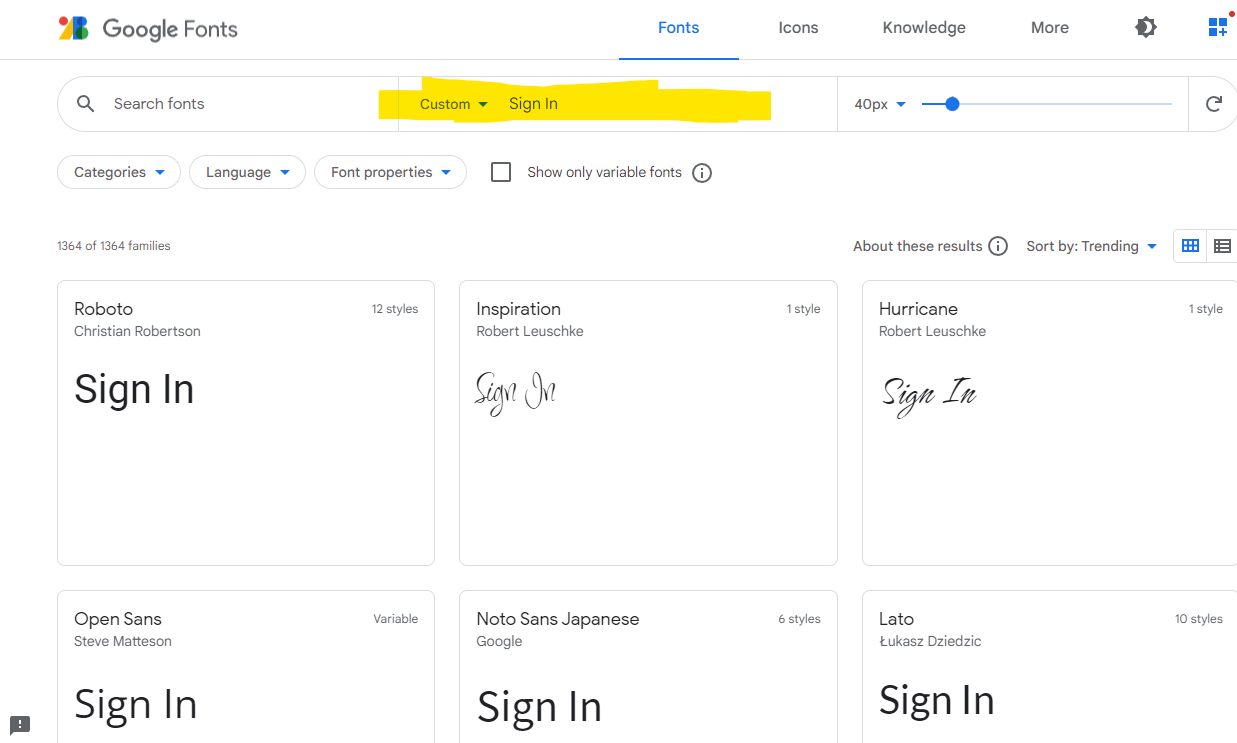
Round the edges using this tool in Design Panel

After, I changed the fill color to black. (Remember, you must select the rectangle layer in order to access the design panel for the rectangle)

Now, I want to round out the edges. Unlike images, Figma shapes have to be rounded using the Design panel (on the right-hand side)..

To start, we are going to create the “Sign In” prompt. After selecting the text, we are going to access the design panel to change the font and the font size.

If you’re a more visual person, head over to [Google Fonts](https://fonts.google.com/).



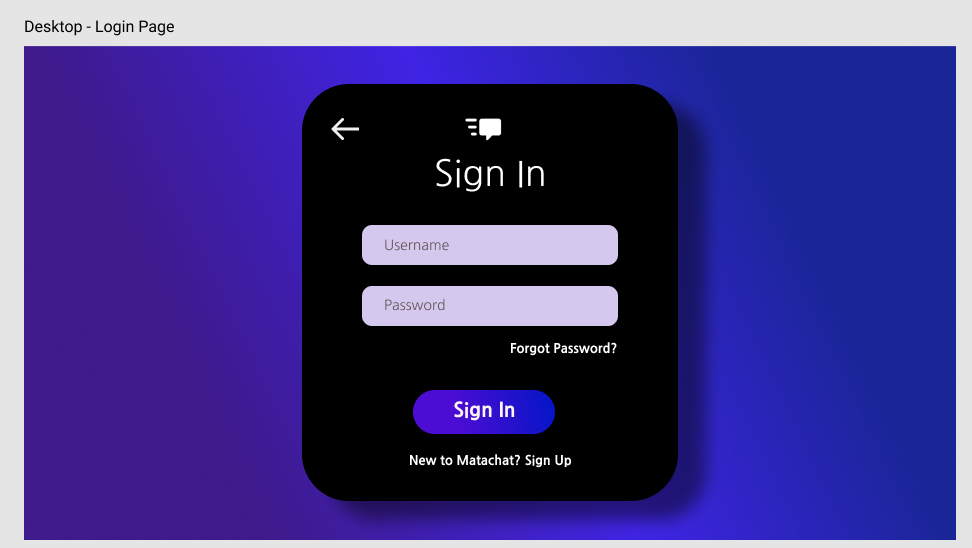
In the Custom text-box, as shown in the screenshot above, type in your custom message which is “Sign In”. Scroll until you see a font you think would look great in your project. All these fonts available in Google Fonts are available in Figma.

Moving on, I’m going to add two input boxes: one for username and one for password. And I will indicate what each box is for using placeholder text.

And to create these input boxes, it’s nothing special. We are still using shapes. Remember, this is a design tool. It doesn’t actually need to function.

I threw in a sign-in button, a “Forgot Password?” anchor tag, and a sign-up anchor tag.

Now that I look at it, a skinny back button would look great at the top left.



**9. Create Git (similar tool) account and configure repository**

## What is a Git Repository?

The purpose of **Git** is to manage a project, or a set of files, as they change over time. Git stores this information in a data structure called a **repository**. In short **Git Repository is a collection of all project files along with their history**.

It is a virtual storage of your project where you keep all the resources/files of the project along with a special folder called **.git.** The **.git** folder in a git repository is used by **GIT** programs to store information about the repository like **Logs**, **Position of Head**, and more. It allows you to save versions of your code, which can be accessed, tracked, and managed.

### **Different ways to Create Git Repository**

We can Create Git Repository using one of the three approaches enlisted below:

1. **Create a bare repository**
2. **Initialize repository in an existing project directory**
3. **Clone a remote repository from Github**

Use of Git command line tool will help you get familiar with the **GIT Commands**. Using Git command line tool gives you more flexibility than [***Git Clients***](https://www.toolsqa.com/git/git-clients/) or **Git GUIs** because they provide only a subset of **GIT Features**. So, to harness the complete power of Git like a pro, **Git command line tools** are the recommended way to use GIT.

* [***Installing Git on Windows***](https://toolsqa.com/git/install-git-on-windows/)

#### **Create a Bare Git Repository**

Creating a Bare Git Repository for a new project is three-step process:

1. Create a New Project/Folder
2. Browse to New Project
3. Initialize Git Repository for the Project

Once you have Git installed, simply search of **git cmd**  in your system search bar. You will get the Command line tool listed as shown in the below image.

Open the Git command line tool, you will see a command line window;

**Step 1: Create a New Project/Folder** - Now that we have our command line tool open, let us **create a project folder**

Command to create a folder on a Windows and Mac system is mkdir<folderName>. Where folder name is the project name. Let us name our first project **LocalGit**. With this name, the command becomes: mkdirLocalGit

Navigate to this folder by using the command cd LocalGit, both on Windows and Mac systems.

Now you are inside the repository folder where we'll create our Git repositories using the three approaches mentioned above.

**Bare Git repository** means an empty directory with just a hidden **.git** folder. Let us name the project as **BareGitRepo**. Ensure that you're in the **LocalGit** folder before proceeding with following steps. Enter command mkdirBareGitRepo.

**Note**: So now you have your Root Folder (LocalGit) ready and within that the Project (BareGitRepository)

**Step 2: Browse to New Project** - Navigate to the project created in the previous step using the command cd BareGitRepo.

**Step 3: Initialize Bare Git Repository for the Project** - Enter the command git init this command is used to Create Git Repository. Consequently, the execution of this command creates a hidden **.git** folder therein. Or in other words, an empty **Git Repository is initialized**. You'll notice a message stating that an empty **Git** repository is created.

**Note**:  **git init** is a standard **GIT** command and it initializes the directory with a **.git** folder used for tracking versions of project artifacts.

After the execution of **dir**, you will see empty project folder as expected from a **Bare Git repository**. You can now add project files to this repository and they'll be tracked by GIT.

However, if you want to see the .git folder being reported in command line use the command dir /a:hd. This will show hidden folders in the directory and you will notice **.git** folder there.

#### **Create Git Repository for an Existing Project**

We'd also like to track an existing project by using Git. In this case, we'll initialize a Git repository in an existing project directory. There is no rocket science in creating git Repository for an existing project, it is as same as creating a git repository for a new project with the only difference of step 1 is not required:

1. ~~Create a New Project/Folder~~
2. Browse to Existing Project
3. Initialize Git Repository for the Project

**Step 2: Browse to Existing Project** - Navigate to the directory containing your project artifacts. In this case, let us consider that the name of the project is **SampleProject** at the location **C:\Users\admin1\LocalGit\SampleProject**, move to a similar location on your system. One can view the content of the directory by using the command dir.

**Note**: I already have few files in the SampleProject for a demo purpose, so it is not an empty folder.

**Step 3: Initialize Git Repository for the Project** - Initialize the git repository in this project by using the same command used while creating a bare repository i.e. git init.

View the content of the directory and note that a **.git** folder has been created therein.

After Git initializing, the project is tracked by **GIT**.

#### **Cloning a Remote Repository from GitHub**

When you want to contribute to an existing project hosted on GitHub or a similar online Git service provider, you've to **Clone** their repository from the remote server like Github, GitLab, etc. For example, the remote repository of Apache POI (Java library to read from and write to Excel files) is hosted at [***https://github.com/apache/poi***](https://github.com/apache/poi)

**Definition of word Clone in the context of Git is to create a local copy of the Remote repository.** In this case, remote Repository is hosted at https://github.com/apache/poi and we will **Clone** it on our local system.

**Here are the steps to clone (download and track the changes) this repository.**

**Step 1:** **Fetch** and copy the URL, as mentioned in the image below, of the Apache POI repository at **GitHub**. That is, [***https://github.com/apache/poi.git***](https://github.com/apache/poi.git)

**Step 2**: In Git CMD, navigate to the folder for storing all the Git repositories locally. That is **C:\Users\admin1\LocalGit** in this example.

**Step 3:** Create a directory called RemoteCloneRepo to store the source code of Apache POI repository locally by using the command mkdirRemoteCloneRepo Navigate inside this newly created directory by using the command cd RemoteCloneRepo

**Step 4**: To clone the repository, enter the command git clone https://github.com/apache/poi.git

**Note**: **git clone *<repoURL>*** is a standard **GIT** command to clone an existing remote repository.

**Step 5:** Cloning a repository depends on the size of the repository. Usually, it takes a while for a big repository. You'll have to wait until all the files are checked out.

Now you can make changes to the repository. Git will track all the changes.

**Steps to Install Git on Windows**

1. Download the latest [***Git for Windows***](https://git-for-windows.github.io/).

2.Go to the folder where new downloads gets store, at my machine by default folder is **Download** folder. **Double click** on the installer. The installer gets save on the machine as per the Windows OS configuration. My machine is **64 bits**.

1. You may like to keep the installation to another folder, so here is the chance to do so.
2. This is the option to store the shortcut of the Git under the **Program Menu**.
3. This is asking your choice that whether you like to Git from the **Windows Command Prompt** or you like to use some other program like **Git Bash**. As of now just select the Windows Cmd for simplicity of the tutorial, later we will cover Git Bash and other as well.
4. If you have PuTTY/TortoiseSVN installed, you may see this screen, otherwise just ignore this. Regardless, use **OpenSSL** to make things easy.
5. Here, we recommend to choose the option of **Checkout Windows-style, commit Unix-style line endings**. Select next once you have done this.
6. Again, just go with default selection and move forward.
7. Just go with default selections, as we will cover the details in later advance chapter.
8. Now, its all done. This will just take few minutes to complete the installation as per your machine speed.
9. Once done, just click on Finish button.
10. Let's just verify if the installation went well for Git. Go to **cmd** and type **git** and press **enter**. you should get the following output on the screen.

The cmdwildow will display different options and commands you can try with git.

## **Install and configure Jenkins**

## **Steps to Install Jenkins on Windows**

### 1. Install Java Development Kit (JDK)

* Download [JDK 8](https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html) and choose windows 32-bit or 64-bit according to your system configuration. Click on "accept the license agreement."

### 2. Set the Path for the Environmental Variable for JDK

* Go to System Properties. Under the "Advanced" tab, select "Environment Variables."
* Under system variables, select "new." Then copy the path of the JDK folder and paste it in the corresponding value field. Similarly, do this for JRE.
* Under system variables, set up a bin folder for JDK in PATH variables.
* Go to command prompt and type the following to check if [Java](https://www.simplilearn.com/best-java-programs-article) has been successfully installed:

|  |
| --- |
| C:\Users\Simplilearn>java -version |

### 3. Download and Install Jenkins

* [Download Jenkins](https://jenkins.io/download/). Under LTS, click on windows.
* After the file is downloaded, unzip it. Click on the folder and install it. Select "finish" once done.

### 4. Run Jenkins on Localhost 8080

* Once Jenkins is installed, explore it. Open the web browser and type "localhost:8080".
* Enter the credentials and log in. If you install Jenkins for the first time, the dashboard will ask you to install the recommended plugins. Install all the recommended plugins.

### 5. Jenkins Server Interface

* New Item allows you to create a new project.
* Build History shows the status of your builds.
* Manage System deals with the various configurations of the system.

### 6. Build and Run a Job on Jenkins

* Select a new item (Name - Jenkins\_demo). Choose a freestyle project and click Ok.
* Under the General tab, give a description like "This is my first Jenkins job." Under the "Build Triggers" tab, select add built step and then click on the "Execute Windows" batch command.
* In the command box, type the following: echo "Hello... This is my first Jenkins Demo: %date%: %time% ". Click on apply and then save.
* Select build now. You can see a building history has been created. Click on that. In the console output, you can see the output of the first Jenkins job with time and date.

Congratulations, you've just installed Jenkins on your Windows system

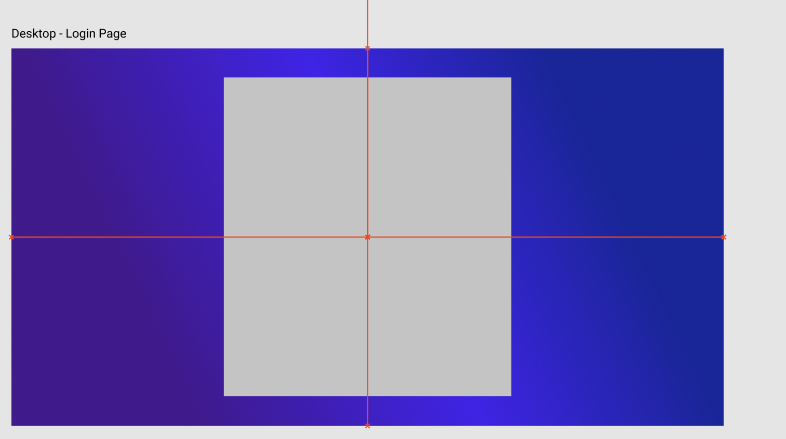
**8.Create sitemap and wireframe for above created user stories.**

Design a Login Page for Desktop

So for the login page, I want the basics. I want two input boxes and a submit button. I want the logo somewhere in there. I want a welcome message and a prompt for the user to log in. Also, maybe “Forgot Password?” and “Sign Up” anchor links, since every website has that.

For the actual design, I think I want the whole login form contained within a container centered in the middle. And I want that with rounded edges.

With design tools like Figma, we can just design it here. So let’s get started.



Here, I created a new rectangle that is going to house my login form. Using those red lines — which Figma offers as a positioning tool — I am centering the rectangle on the page.



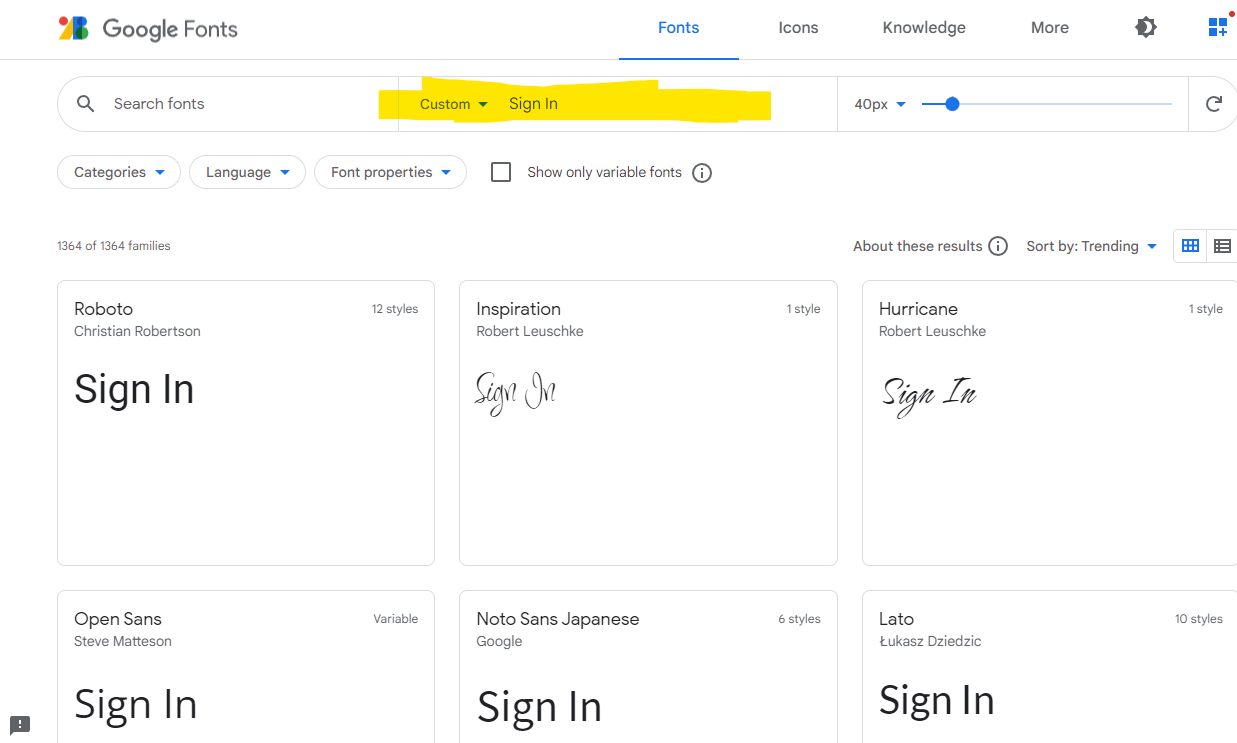
Round the edges using this tool in Design Panel

After, I changed the fill color to black. (Remember, you must select the rectangle layer in order to access the design panel for the rectangle)

Now, I want to round out the edges. Unlike images, Figma shapes have to be rounded using the Design panel (on the right-hand side)..

To start, we are going to create the “Sign In” prompt. After selecting the text, we are going to access the design panel to change the font and the font size.

If you’re a more visual person, head over to [Google Fonts](https://fonts.google.com/).



In the Custom text-box, as shown in the screenshot above, type in your custom message which is “Sign In”. Scroll until you see a font you think would look great in your project. All these fonts available in Google Fonts are available in Figma.

Moving on, I’m going to add two input boxes: one for username and one for password. And I will indicate what each box is for using placeholder text.

And to create these input boxes, it’s nothing special. We are still using shapes. Remember, this is a design tool. It doesn’t actually need to function.

I threw in a sign-in button, a “Forgot Password?” anchor tag, and a sign-up anchor tag.

Now that I look at it, a skinny back button would look great at the top left.

