# Minimum viable product (MVP)

Almost all the Web sites that you use in the course of a year are based on content management systems. In a nutshell a content management system (CMS) allows someone to change the content on the Web site without knowing how to code. When you log into Facebook you are allowed to change things because it is a CMS and you are given permission to do so. You should also note that Facebook, and all CMS’s, do not require you to understand the underlying programming – you can easily and naturally add content. In this program we are going to build CMS’s. However, there is no way we can start with Facebook and build that. What are we going to do?

A concept in Agile development is the Minimum Viable Product or MVP. The overall purpose of determining the MVP is to do the least amount of work to create the most amount of value. You are trying to determine the core features that will satisfy the client and as a result provide the audience enough features to use what you are building. Once you have the core features working then you can start asking the audience, either directly or through other means, what new features they would like. What the MVP does is require you to build what is needed now and not worry about making all the bells and whistles that might distract you from finishing the project.

In our case, semester one Web Development program, we are looking into redesigning a Web site – which is to say we are creating a content management system (CMS) that will allow our client to update and make changes to the Web site and in effect take control of it to their benefit. Based on that and your understanding of Web sites what is the least amount of work, or the least number of features, that a CMS needs to work? There could be a lot of debate about this – and that would be a positive exercise in discovering what the MVP is. You could also investigate with the client to determine what they saw as the MVP or what they absolutely needed or wanted. All these are valuable and necessary parts of discovering the MVP. However, we are going to move quickly here: a CMS needs to do several things to be a CMS:

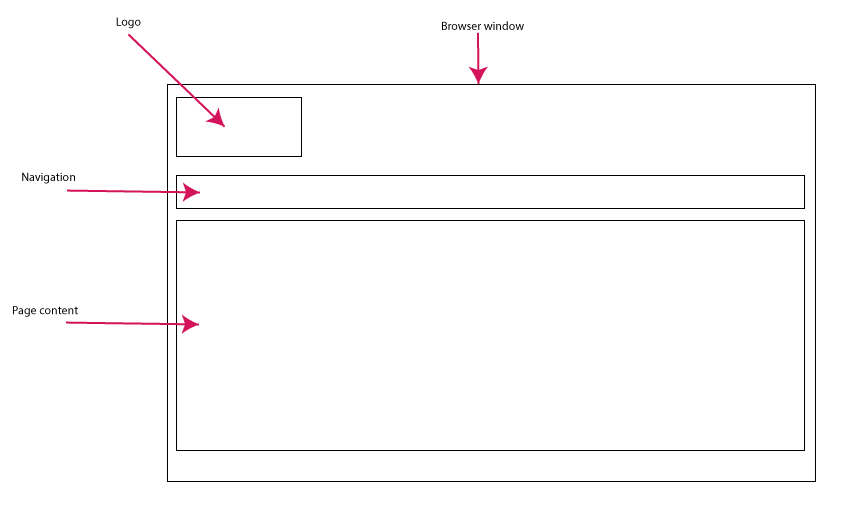
* Have a page,
* Have a link to the page,
* Have links to all pages,
* Have a means to add pages, delete pages, and update pages.
* Have a database to hold the content.

Another way to look as this functionality is CRUD: Create, Read, Update, and Delete. Get to know this term – we will be using it a lot.

## Adding pages and adding navigation

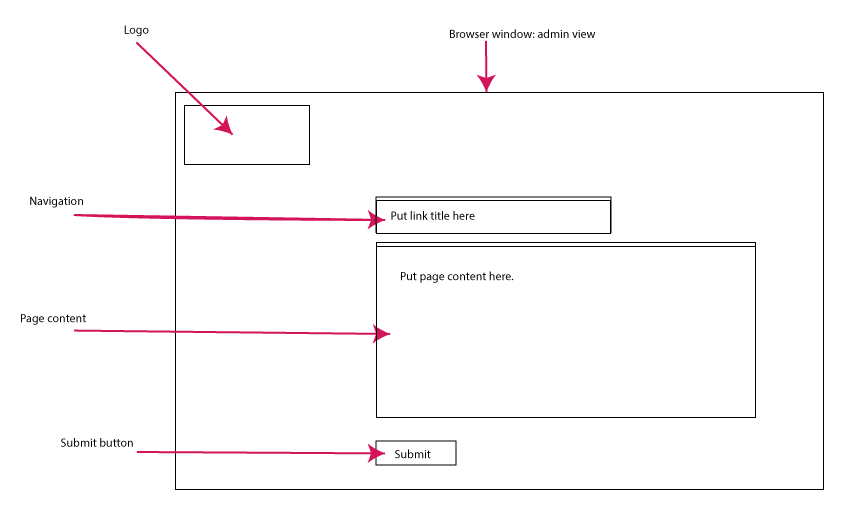
Since we are making a content management system - anything we can add to the database is content managed by our system. However, there is core content that we need to be able to add and that is paged content. A page in this case would be something like the “about us” page or the “copyright page” or anything that is a single topic: content. Content has to be managed and we do that with CRUD.

What follows are wireframes showing different aspects of the main operational features of CRUD – Create, Read, Update, and Delete.



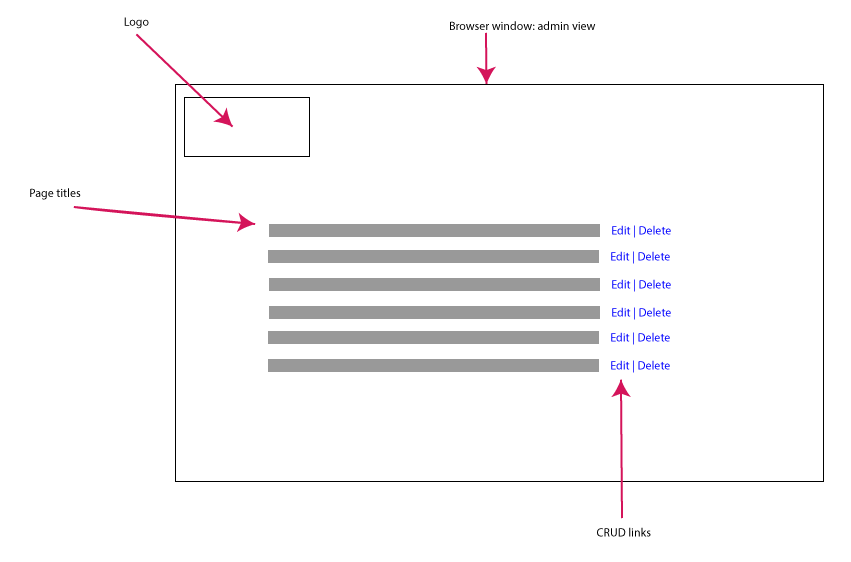
Public view – Read of CRUD

At the same time when we add this content then we need to be able to add a link to this content in the navigation system. The image above shows you a wireframe version of the MVP we are talking about. The logo anchors and identifies the page, the navigation gives us all the links to the available pages, and the page content is for this page and link item. The links to the pages would go into the navigation region and the individual content would go in the larger page area. Input is managed using an HTML form – in fact this is where most of the programming will occur. It is with this form that CRUD operations are done.



HTML form view – Create and Update of CRUD

There are two components to this MVP: add a page, and add a link to that page. This information would be managed by CRUD (Create, Read, Update, and Delete) operations. We would also need an output page that generates a list all of the pages entered into the database. With each page reference (a link to that page) would be options to edit or delete. At the bottom of this page would be a button or link to create a new page.



A listing of all pages – select either Edit or Delete – Update or Delete in CRUD.

### Add a new page feature

In figuring out this content item you need to determine what are the components of a page that you need to give your client access to. A page has a title, body content, navigation term, the ability to add images, and perhaps other documents like PDF files. Think of these as text fields in an HTML form that issues commands to the database. Your database now has several columns for each of these elements including a link to the menu system. Since we are building the MVP we will only work with text for now. But as you can see – adding elements to the MVP is easy to do. We have to be careful that we do not add too many of these new elements as we would no longer be controlling what the MVP is.

### Add a link to the menu feature

The menu system is the Web site navigation that we see on every page. This is the essence of creating a scalable Web site. When we add content we have to have a means of adding navigation. For pages this would be the main navigation and if we were adding products we would add navigation to the products pages. We are creating a series of links that are the main navigation, which are the parent links and the secondary navigation that would be the child links. Your database would have to account for this parent and child relationship. For the MVP we would simply create the primary navigation. For the purposes of this MVP only the primary navigation is necessary.

We can see a database of primary navigation, the parent, and subsequent tables with secondary navigation linked to the primary navigation and then linked to the individual page. These tables would be linked to the each of the pages on the Web site.

The database structure could look something like this:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Primary\_nav | Type |  |  |  |
| Primary\_Key | numeric |  |  |  |
| Foreign\_key | numeric |  |  |  |
| Nav\_word | text |  |  |  |
| Nav\_page | text |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  | Secondary Nav |  | Tertiary\_Nav |
|  |  | Primary\_key |  | Primary\_key |
|  |  | Foreign\_key |  | Foreign\_key |
|  |  | Primary\_parent |  | Secondary\_parent |
|  |  | Secondary\_Page |  | Tertiary\_Page |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  | Page |  |  |
|  |  | Primary\_key |  |  |
|  |  | Page\_title |  |  |
|  |  | Page\_content |  |  |
|  |  | Parent |  |  |
|  |  | Primary\_nav | true/false |  |
|  |  |  |  |  |

Please note: this is not a final database design. We would have to ensure that it is optimized and functional for our needs.

Overall, this is the least of what is needed for a content management system: pages and links. As well as a means to add new pages and add new links. Or to edit those pages and links.

**Your assignment is: in 250 words or less describe either ebay, Facebook, or Youtube, as an MVP. This will be due, by email to me, on Friday October 13th at noon.**

**Questions to answer:**

**What kind of users are there? Who can do what to the content? What can users do? One thing you can consider doing is writing a few user stories to explain how the Web site works.**

**“A registered user can change their password.”**

**You don’t have to consider all functions just enough to make the system work. That would make it the least needed to be functional.**