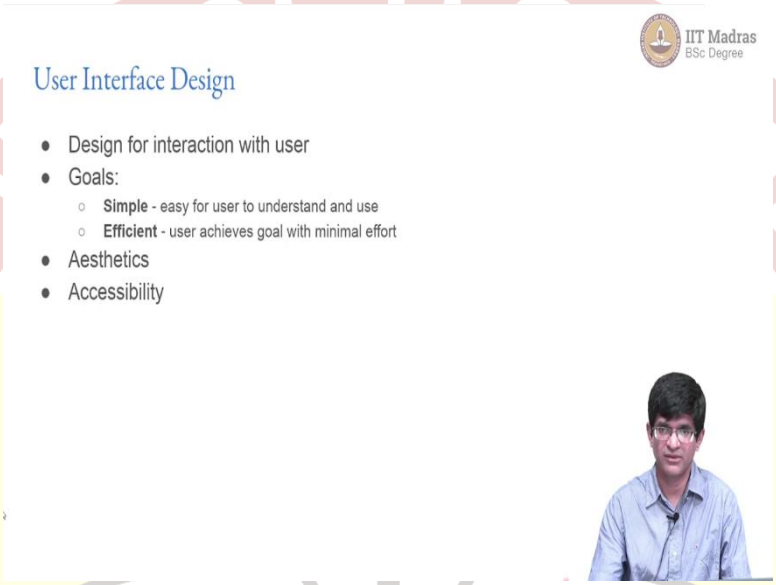
IIT Madras

ONLINE DEGREE

Modern Application Development – I
Professor. Nitin Chandrachoodan
Department of Electrical Engineering
Indian Institute of Technology, Madras
User Interface Design

Hello, everyone, and welcome to this course on modern application development.

(Refer Slide Time: 0:16)



So, there are a few principles with regard to user interface design that I am going to touch upon over here, there are some links that I provide over here that are useful reading material. And I would strongly recommend that you go through them. In fact, it is almost essential that you need to go through outside material, because what can be covered just by giving examples over here is very minimal.

There are so many ideas out there and so many different theories of user interface design, that it is not possible to have one particular approach and say this is the right way to do it. But some very broad principles are what I am going to lay down over here and try and convey in terms of what we have.

So, the user interface design is designed for interaction with the user. And the primary goals of any kind of user interface design should be right. I am not saying that they always are, because you are very likely to come across user interfaces that are non-intuitive and not very nicely done. But the goal of user interface should be simplicity, which means that it is easy for the user to understand what that interface and to use it. And also, it should be efficient.

In other words, the user achieves the goal with a minimum of effort. A good example of this is actually the push and pull handles that are used on doors. So, going back to that non computer example, it is when you see a sort of slab of metal with push written on it, it is actually very easy to understand what it implies. As long as of course, the one problem is that you do need to be able to read it and understand it.

But even if you do not, that slab of metal on a door usually indicates that if you press over here, something is likely to happen. So, it is easy to understand, even a person who does not know English is likely to figure it out fairly quickly. So, it is simple, is it efficient? Yes, all you need to do is press on it and the door opens.

Contrast this with something which maybe had a very fancy looking handle, with maybe a bunch of buttons on it, and a speaker and a microphone and video display. And person comes stands in front of it and does not really know how to open the door, they have no clue. Do I press something? Do I speak to it? Do I need to push something else somewhere? How do I get this door open? That is the question a person is going to ask.

And if it turns out that what you actually need to do is, you know, press a button, it will then connect you with somebody, you then need to speak to them, you need to ask some permission, they will do something else, and then they will open the door for you. Maybe that is required, from a security point of view, it is definitely not the most efficient from the point of view of opening a door.

So, in the same way, you will find that anytime that you are looking at different kinds of applications, you can sort of intuitively form a judgement of whether or not this particular structure or this application is good or not. What I mean by good is simple and efficient. There are a couple of very important aspects over here. One of them is aesthetics, how good does something look. Now, aesthetics is very difficult to teach.

On the other hand, I mean there are actual, like full courses devoted to the Fine Arts and Design. And typically, you need to have some kind of grounding in the arts in order to appreciate what is meant by aesthetics. For the most part, aesthetics just refers to what is pleasing, what is nice. So, the simple way to look at it, when you ask the question, is this aesthetically pleasing is does it look nice?

Now, very often, especially when we are starting to build something with web pages, we see what HTML is capable of, we see that we can make fonts larger, smaller, we can centre them,

we can change colours, there is a tendency to get carried away, why do not I use these different fonts? Why do not I make something blink on the side? Why do not I use a different colour over here? Unfortunately, that is most of those choices are a bad idea from an aesthetics point of view.

There are multiple reasons why they are bad, most of them have like fairly, good analysis in the domain of the fine arts. And for the most part, you are actually going to be better off by looking for some standard principles that have already been tried and tested by people who have a good background in the arts, and they say, use these colours, use these kinds of fonts use this kind of layout. And all of this is visually pleasing. It is not just that somebody sat out there and said, this is what is right and everybody else should follow this.

There are people with a good sense of taste, who come up with these ideas. And then they experiment on it. They try it out on different people, they see what kind of reactions people have, and they find unit. So, especially when you are a beginner going into application design, it is a good idea to look for designs that are already known to be aesthetic known to be visually pleasing. And try and use the same kind of approaches.

This is where, for example, libraries like bootstrap, the CSS styling library called bootstrap, which we will be talking about, again, as part of the assignments over here, is very useful and good to hang around, because it has essentially distilled out a lot of useful information that people have found to be aesthetically pleasing. How should I display buttons? What size should the buttons be? What kind of colour? Should I use rounded corners or flat corners? How far should they be spaced? How big should a button be?

A lot of those decisions have already been made for you. It gives you a good amount of freedom in terms of how you actually go about choosing it. But at the same time, using those things is very likely to make your application visually pleasing to someone who is looking at it for the first time.

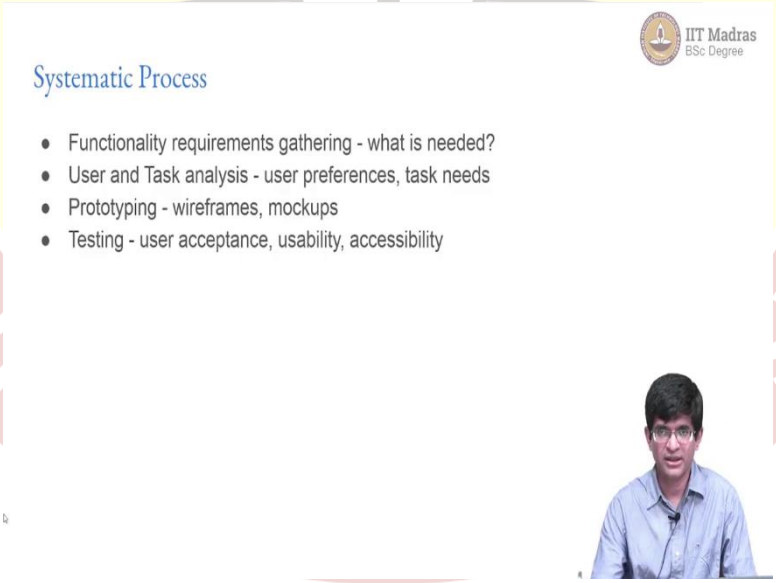
Now, an equally important, if not more important aspect these days, especially is something called accessibility. And accessibility refers to how accessible or how usable is this website for someone who has some kind of disability or impairment. I will be talking about this later on. For the time being, the main thing to keep in mind over here is there is usually a big sort of conflict between aesthetics and accessibility.

Very often the kind of colour schemes and layouts that we would like to use in order to make something look aesthetically pleasing, make those things difficult for a person, let us say with low contrast vision, who has a visual impairment is not completely visually impaired in the sense that they have vision. But it is something called low vision. Meaning that they cannot make out colours that are close to each other, they cannot make out distinction. So, then they need high contrast. And they also need large fonts.

The moment you try and get something which is suitable for a person with those kinds of problems, the aesthetics for the general public are a bit harder to match. People usually find that those are jarring colours. They are bright colours, yes, but they are not really visually pleasing. At the same time, you try and make it visually pleasing, it ends up being a problem for a person's accessibility.

So, aesthetics and accessibility are very important aspects to keep in mind from the point of view of user interface design. And there are sort of again, standardised techniques and principles that are recommended to be followed for this, which we will again briefly look at later.

(Refer Slide Time: 8:15)



Now, this process of user interface design can best be handled by following a systematic approach. And some of the techniques that are done, I am just breaking it down into a set of steps is you need to first go about doing functionality requirements, gathering. Meaning that you need to talk to the people who want to use the application. And find out what exactly are your functionality requirements.

Now, this obviously if you are designing an app, where you have some new idea, and you want it to be for, the mass public consumption, you do not really have an audience, you cannot go around asking people what kind of functionality do you want? In that case, your functionality is already decided by you. But the majority of apps are actually designed in a different context. You might be maybe, contracted by a company to design a specific set of applications for their intranet.

And maybe it is something which, it might be the grade book application, and then you probably need to go to the academic section of the university and ask, okay, what exactly are the requirements of the grade book? And they probably tell you, okay, look, I need to be able to enter a list of students. I need two ways of doing it, either I should have a form into which I can enter the names one by one, or I should have some way by which I can Bulk Upload, because you know, if I have 10,000 students, and I have to do them one at a time, that is nearly impossible.

So, if you generate a single form and give it to the academic section and tell them, you know, go ahead and start typing in the names of 10,000 students, they are never going to use the app. So, functionality requirements. Do I need to be to upload one at a time, n at a time, update how many, what kind of updates are likely to happen, all of those are things which you probably need to talk to the consumer of the application to find out clearly.

Similarly, a related part is what is called user and task analysis, the person who is going to end up using the actual application will tell you that, these are the kinds of tasks that I am going to do, and this, these are the kinds of people who are going to be using them. It might be a person who has access to all the data and just needs to enter it. And it is not really otherwise in control of the students or the courses.

Once you have this information, you can then go about prototyping. A prototype is something which is just trial design, so to say, and these prototypes very often are done using something called wireframes or mock ups. They do not have proper functionality yet. They just sort of put up a screen or there and tell you this is what it is going to look like. And they ask you will this be good enough? Will you be able to work with something like this?

And eventually, after all of this is done and implemented, you would then need to actually go through the testing user acceptance. So, the users who have already given you their inputs on what kind of tasks they needed to complete and what kind of information they would have

will finally have to get back to you and say, yes, good enough or no, we need this to be improved.

So, this kind of a systematic process is pretty much typical of any kind of engineering. And ultimately, that is what you need to do for this as well. It for any kind of application that you are trying to develop.