Software Engineering
Prof. Rajib Mall
Department of Computer Science and Engineering
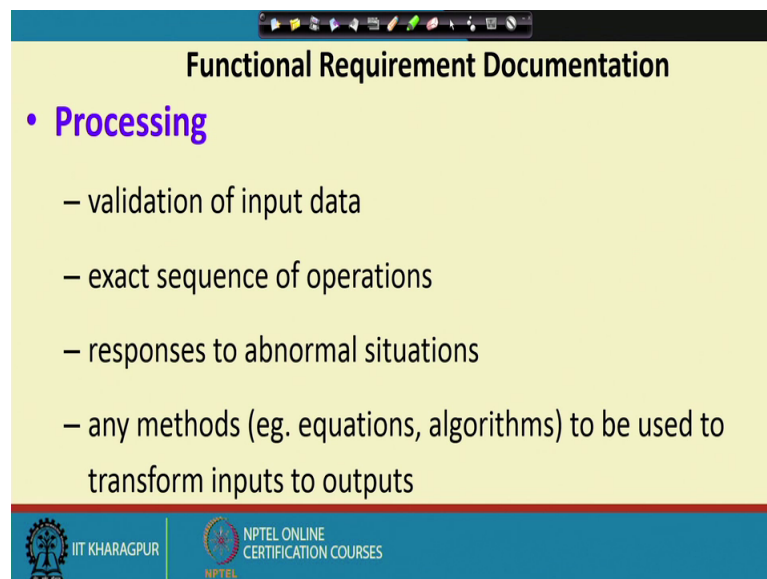**Indian Institute of Technology, Kharagpur**

Lecture – 17
Functional requirements

In the last lecture, we were examining how to write SRS document. We identified what are the important information that SRS document must have. We said it must one of the most important information that is present in the SRS document is the Functional requirement. We are discussing what exactly is a Functional requirement? We said functional requirement is a facility offered by a software to the users using which the user can get some meaningful work done.

And then, we were discussing how to document the functional requirement? We said that for each functional requirement, we must give an overview of the functional requirement that is what exactly it will achieve and then, the input data that will be given and the output that data will be produced and also the procession.

The processing should be in terms of maybe validation of the input data.
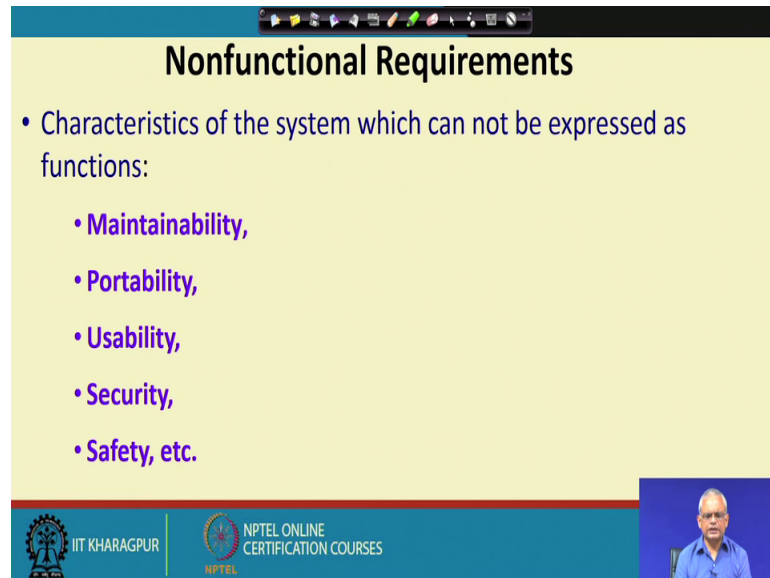
(Refer Slide Time: 01:49)

It can be the exact sequence of operations that will be carried out. It can also contain responses to abnormal situations and also any methods or description of algorithms equations that can be used to transform the input to output.

(Refer Slide Time: 02:13)
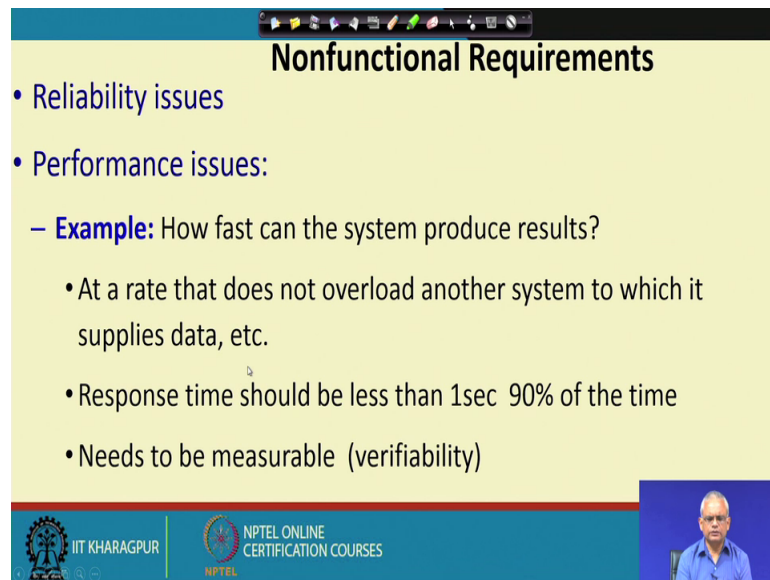
Now, let us try to see what exactly is a non-functional requirement? A non-functional requirement is a characteristic which cannot be characterized or expressed as a function. A function is a feature using which a user can get meaningful work done; but there are some requirements, which are not like the functional requirements. For example, we might say that it should run using both Windows and the UNIX environment. It should be easily maintainable; it should have a graphical user interface that can be used in a standalone mode or through a web browser. It should be extremely secure etcetera.

So, these are all examples of non-functional requirement, these apply to the software as a whole and these are not a functionality or a feature offered to the user to get some meaningful work done.

(Refer Slide Time: 03:54)

It can also include reliability issues that it is should not fail frequently and if it fails should be able to recover the data quickly. It can also contain performance issues. Non-functional requirement can also contain performance issue. For example, how fast the system should produce result? We can say that the response time should be less than 1 second 90 percent of the time or we may say that it should not produce result to fast such that other systems that work with it are not overloaded.

But for all non-functional requirement, we have to ensure that somewhere we can test that; that is we should be able to measure and see if the non-functional requirement, there is a confirm conferment's to that another important section.

(Refer Slide Time: 05:09)

In the requirement documents are the Constraints. For example, what hardware to be used; what are the other software with which it needs to work? The operating system to be used the capabilities of the I O devices whether in a standards complaint is needed? Data representation the interface system, these are all constants.

(Refer Slide Time: 05:36)

And then, the external interface requirement. This is also an important section need to document the requirements for the user interface. The interfaces with other hardware with which it needs to work with other software the interfaces. For example, we might

say that for other software, it need to produce a XML file or maybe a comma separated file a text file comma separated text file.

The communication interface with other systems file export formats etcetera.

(Refer Slide Time: 06:30)

There is another section which is the goal of implementation. For both functional requirement non-functional requirement constraints, we need to check whether after the software is complete we need to test whether these are satisfied, where is goals of implementation are the general suggestions, which can help the developer to see what is required. But this will not be tested.

For example, under the goal of implementation, we might say that it should be very reliable because you will use it in because this will be used in a very critical application. But then, as long as it is under the section goal of implementation, this will not be tested; it is only a guideline or it is a suggestion to the developers.

(Refer Slide Time: 07:39)
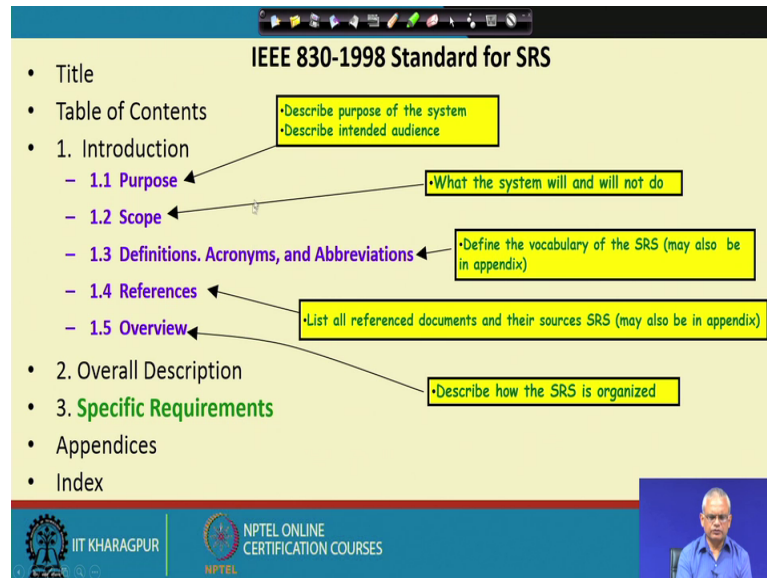
Now, let us look at the IEEE 830 standards. This is a standard format of writing the SRS document. Most developers use the standard and of course, there are small variations to the standard; but if we know the IEEE 830 standard should be able to have small variations of this. Now let us look at what are the sections of the IEEE 830 standard title that is name of the software table of contents and then, there are 3 main sections; one is the Introduction, Overall Description and the Specific Requirements and then, there are the Appendices and Index.

The introduction should have a purpose; subsection called as Purpose, Scope, Definition, Acronym, Abbreviations, References and Overview. The purpose should say that what the system will solve. So, general introduction and describe who are the users. Scope very overall idea about what the system will do; what it will not do, very crisply written and overall description of what is expected of the system and what it should not do. Then, there are the various terminologies; the definitions, acronyms, abbreviations that will be used in the SRS document.

But, often variation to the standard, these are put under the appendices. Similarly, a setup documents that are referred in the SRS documents. These references can also be put under a separate appendices and then, an overview of the document or organization of the document; how the SRS is organized and that is about the introduction. Now, let us look at the overall description. This is the section 2.

(Refer Slide Time: 10:32)

The sub sections here are the Product Perspective, Product Functions, User Characteristics, Constraints, Assumptions and Dependencies. In the product perspective, we write here the business case that is how it will help the customer; how it will benefit how it will be used and so on and even met write about the external interfaces. Briefly the overview of what type of users, hardware, software etcetera will be used and then, any constraints in the memory are operational constraints site constraints and so on.

The product functions this not really the functional requirement specification; but a brief overview or a summer summarize a summary of the functional capabilities. For example, in a library, we might just write very briefly saying that the library software should serve the users for their issuing of books returning for the librarian to create members manage books and so on and for the accountant to manage the financial aspects of the library.

The user characteristics are the technical skills of each user class. This is also required because finally, the user interface development we will need the user characteristics. If it is software is to be used by factory workers as well and programmers, we need to have different user interfaces. So, need to identify the user characteristics and how much what are the technical skills how will their convergent with the computers and so on. The constraints here, what are the platforms that will be used? The database that will be used; the network development standards and so on.

And then, if there are any assumptions here dependencies need to mention that and then let us look at the third section. This is possibly the most important section. Here we have the external interfaces functions; this is the major section here. This contains the functionalities the functional requirement.

The non-functional requirement Logical Database Requirement, Design Constraints, Quality Attributes and any Object Oriented Models; in this section has to be written very carefully and in sufficient detail. So, that the designers can have all the information they need to carry out the development work and for the testers to be able to write the test cases. It is a very important section. While writing the functional requirement, we should make sure that we have as far as possible all the input or stimulus to the system; output that is produced by the system and the functionalities.
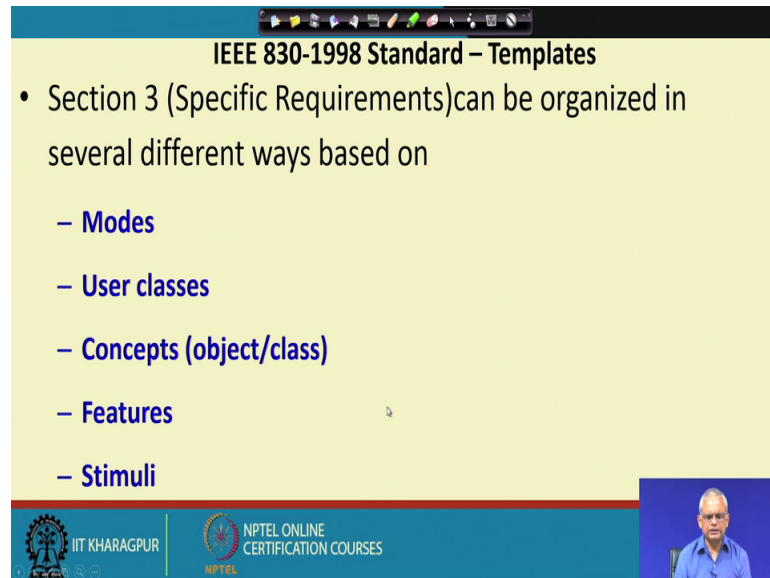
Now, first let us look at the external interface. Here, we have to write about all the user interface maybe you can give a GUI screenshots, the different file formats that may be needed. It should actually there are some part that were mentioned in the section 2; external interface. It should actually elaborate that and should not duplicate information there. This is the main section the functional requirements. This will contain the detailed specification of each functionality or use case including collaboration and other diagrams we will see these diagrams later.

Then, the Performance Requirements; the Logical Database Requirement; what are the different data that will be stored in the database the relationship? The Design Constraints are standard complaints. For example, we may say that I have to use UML 2.0 and have to use let us say (Refer Time: 16:08) for writing the code. The quality attributes and then any object oriented models like class diagram, state collaboration diagram, activity diagram etcetera can be include.

(Refer Slide Time: 16:26)

But then, the section 3 that we just saw now is just one example and IEE 830 specifies that there can be different organization of the section 3 based on the modes of the software that is software maybe having a expert mode, a novies mode and so on. If there are modes are there then, we might have to organize it such that the modes the software is written in terms of the modes. Sorry, the SRS document is written in terms of the modes.

Similarly, depending on different user Classes, Concepts, Features, Stimuli; there can be slightly different organization of the section three. You just look at a we examined a every typical organization of the section 3; but there can be variations the section 3.

(Refer Slide Time: 17:32)

Now, let us look at the functional requirements as we said that this is one of the most important part of the document. Let us take an example of a Academic Administration Software or AAS. So, write here section 3.1 functional requirements and then, we write various functional requirements and they might have sub functions.
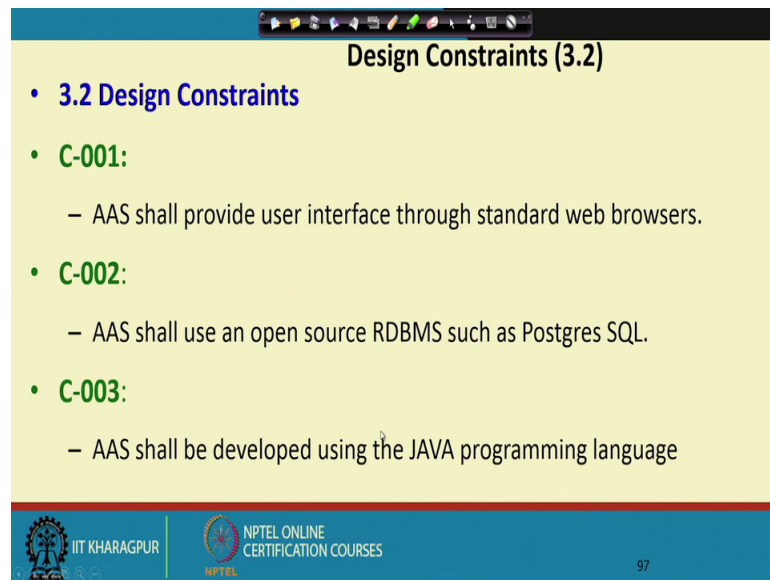
For example, we organize the subject registration that is 3.1.1 and then there can be several functionalities for subject registration. In the first function may be the system shall allow a student to register subject; second function can be allows a student to drop a course; third one is that it support checking how many students have already registered for a course.
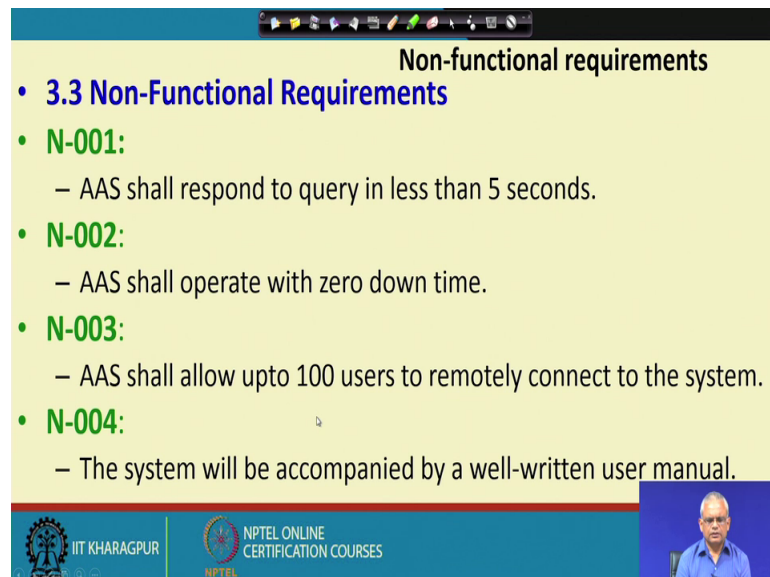
(Refer Slide Time: 18:40)

Similarly, the design constraints; we give a number to each constant and then write that AA software, provide user interface through standard web browsers. It should use an open source RDBMS. It should be developed using JAVA programming language. So, these become different constants.

(Refer Slide Time: 19:09)

And then, the non-functional requirements again we give a number to each (Refer Time: 19:15). It should respond to query in less than 5 second. It should operate with zero down time. It allow up to 100 users to remotely connect to the system. The system

should be accompanied by a well written user manual. So, these are non-functional requirements.

(Refer Slide Time: 19:37)

Now, let us focus on the functional requirements because we should be able to write a good SRS document; you must know what are the granularity to which we must document the functions and what exactly are the functional requirements.

Let us look at it in further detail. As we have already mentioned that every software we can view it as performing a set of functions. Let us say a software. So, this is the software which let us say supports 3 functions. Function 1, function 2 and function 3; using each function, the user should be able to achieve something and each function will typically take some input and the system should process that input and produce some output..

You can think of each function as taking some input data and through some processing activity or function or a function transfer the input data to some output data. Let us take an example of a query book; takes the name of the book as the input data; searches the book database and produces all the books details that match the given book name.

(Refer Slide Time: 21:52)

Now, let us take some example let us say the search book. The search book let us say it takes name of the book sorry an author's name. So, then we can write the input is author name and the output is details of the authors books and the locations of this book in the library.

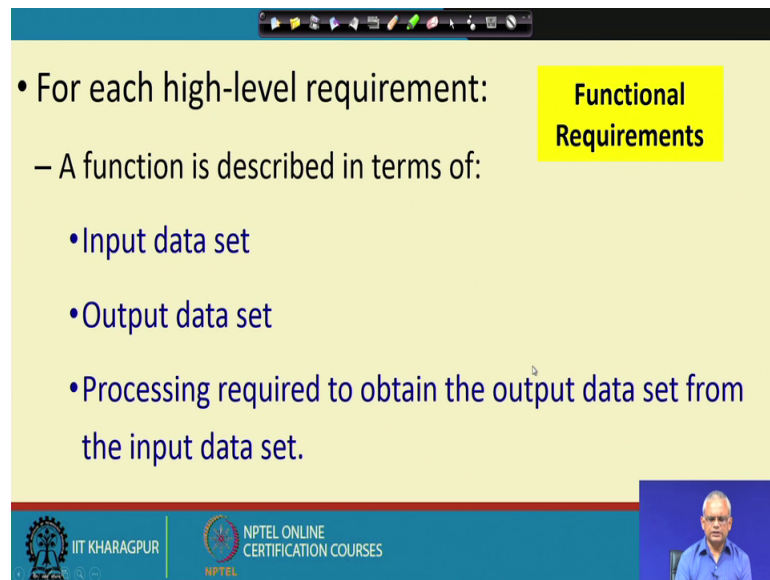(Refer Slide Time: 22:13)

Each functional requirement we called as a high-level requirement a high-level requirement takes some data and produce a some result; but then, each high-level requirement may consists of sub requirements or sub functions.
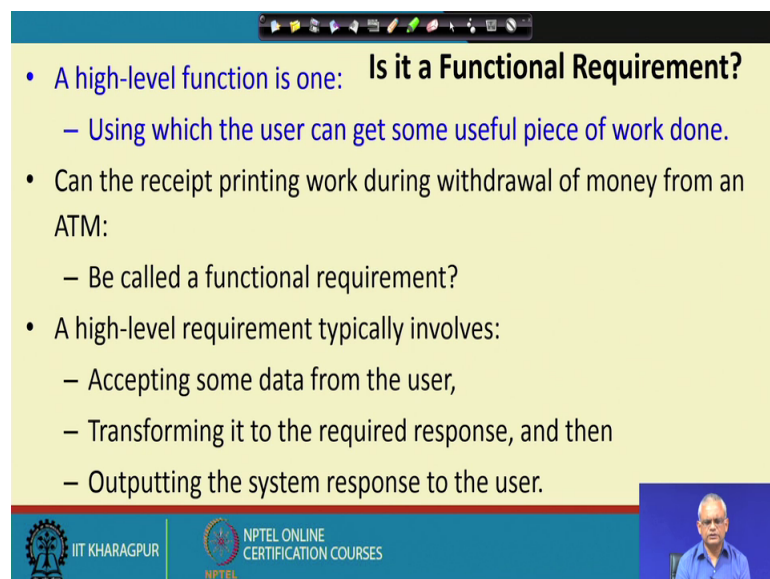
(Refer Slide Time: 22:44)

Each functional requirement is described in terms of input data. The data that is input which the function uses to produce the output data and what processing it does to produce the output data from the input data.

(Refer Slide Time: 23:03)

We had already said that high-level function is one or a functional requirement using which the user can get some useful piece of work done.
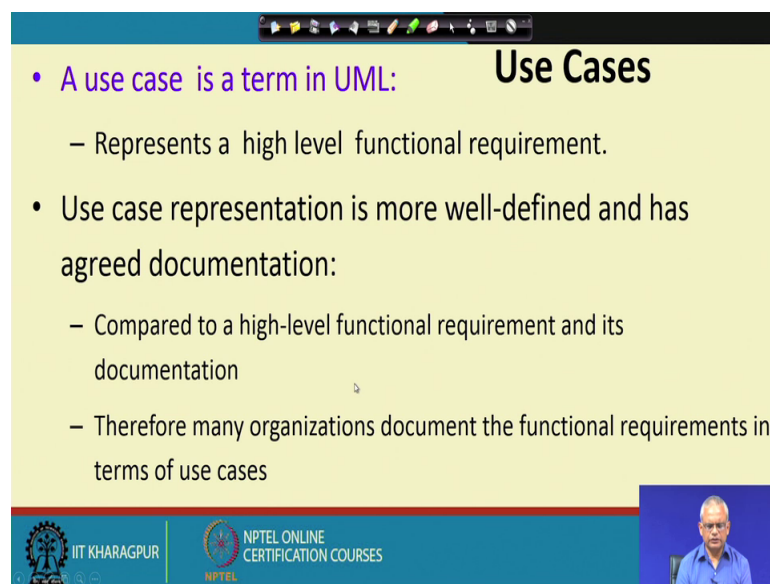
But suppose, it take the example of ATM machine a bank ATM what are the functionalities or the requirements of this ATM machine. One can be withdraw cash; one

can be query balance; one can be deposit cash, but let us say during the withdrawal cash, it asks whether to print a receipt or not.

Is printing the receipt can be considered as a functional requirement? The answer is no because as per our definition of a functional requirement, it takes some input from the user and produces some useful piece of work for the user..

The receipt was actually printed as part of the withdrawal. It is not that just printed the receipt as the user requested the print receipt and it printed; it is always executed as part of the cash withdrawal and therefore, it is a sub function of the cash withdrawal requirement. Every high-level requirement typically accepts some data from user; transforms it response and then output these response to the user.

(Refer Slide Time: 25:18)
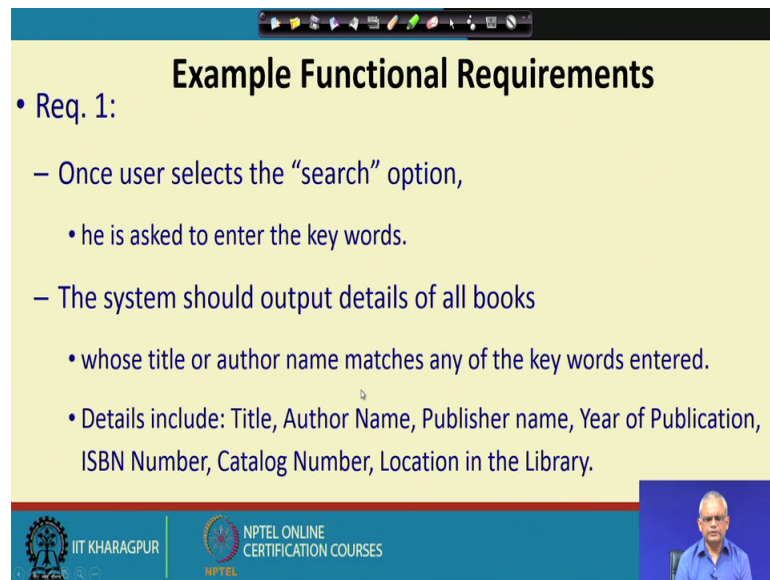
Little later, we look at the UML a where the requirements are specified using use cases. The use cases have become standard. These have definite notation how the functionalities are represented and these are much more well defined then just loosely writing the requirements.

So, many times the requirements are specified using use cases.
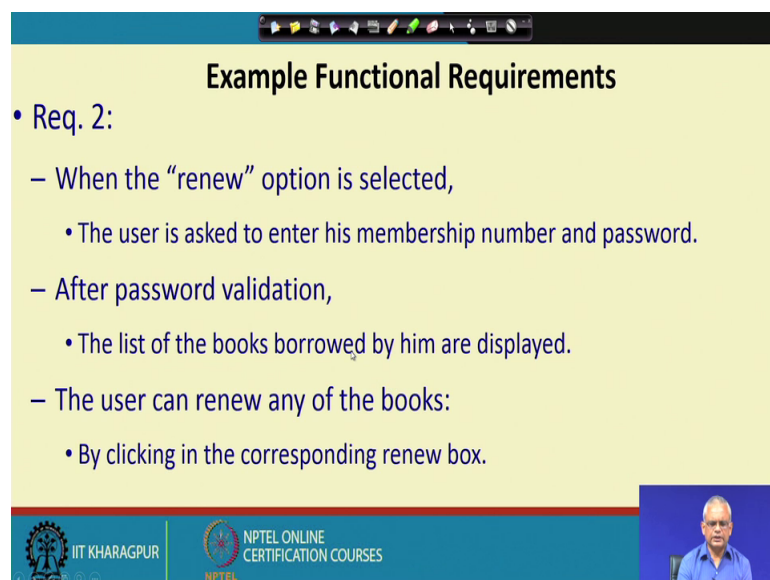
(Refer Slide Time: 25:57)

Now, let us look at some few example functional requirement, we need to number the requirement and then write that once user selects the search option asked to enter the keywords. The system should output details of all books whose title or author match the keywords and the details that will be output include these items the title, author name, publisher name, year of publication etcetera.

(Refer Slide Time: 26:33)

Similarly, you can number the second requirement; requirement 2 when the renew option is selected the user is asked to enter the membership number and password; after

password validation list of books borrowed by him are displayed and then, the user can renew the book any book by clicking on the corresponding renew box.

Just say that this is written in rather free form; in the way that the sequence of operations are carried out that initially the renew option is selected, this is done by the user and the computer ask the user to enter the membership number and password and after password validation, the books borrowed by him are displayed and then the user can renew the book by clicking the corresponding renew box.
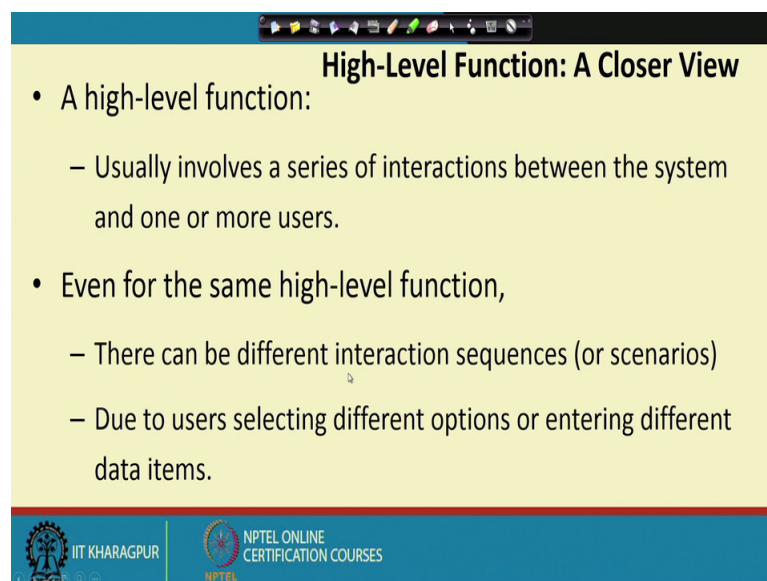
(Refer Slide Time: 27:35)

You can realize that every high-level function actually is a series of interaction between the system and the user. The user input something based on that the system does some processing; produces some output and based on that the user enters certain other thing and so on.

So, you can think of every high-level function as a series of interactions between the user and the computer; but then, for every functional requirement, we must also document the scenarios. The scenarios arise because there are different ways certain things can get accomplished.

For example, let us say the cash withdrawal this is a functional requirement and then, the normal course of cash withdrawn is that the user specifies the amount to be taken withdrawn and the system dispensers the cash. There can be other scenarios that the user

entered the amount to be withdrawn and the system should display that it exceeds the account balance.

There can be another scenario that the user entered some amount in the system responded that please enter in multiples of 100. So, we have looked that the functional requirement; every software performs a set of functions for the users and this has to be carefully documented each function or high-level requirement consists of a number of scenarios. We look at some examples in the next lecture and we will continue from that point onwards.

Thank you.