INDIAN INSTITUTE OF TECHNOLOGY MADRAS

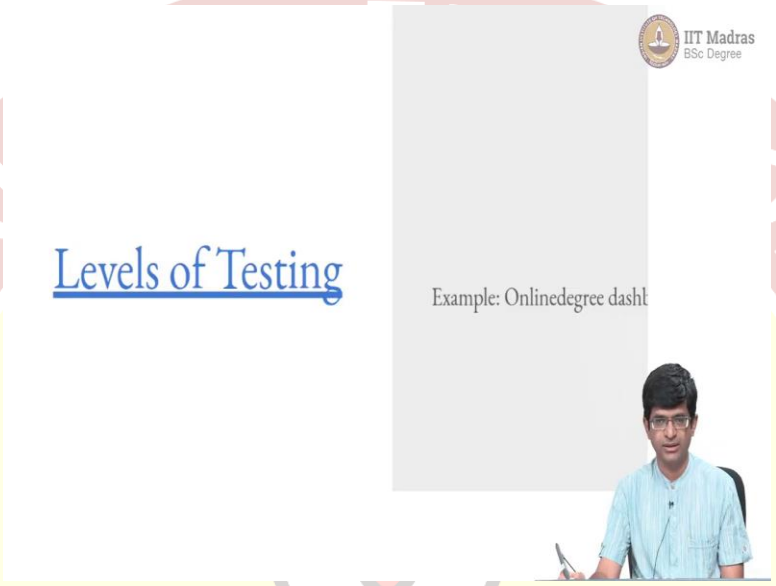
सिद्धिर्भवति कर्मजा

IIT Madras

ONLINE DEGREE

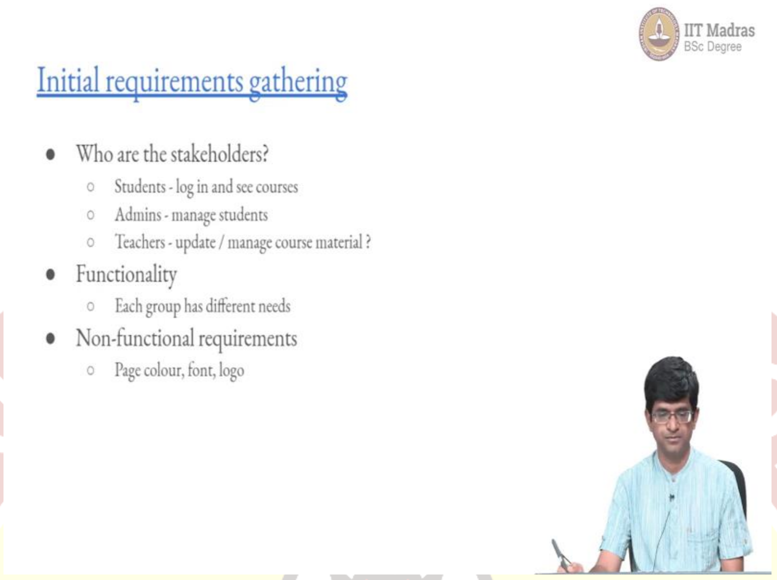
Modern Application Development – l
Professor Nitin Chandrachoodan
Department of Electrical Engineering
Indian Institute of Technology, Madras
Levels of Testing

(Refer Slide Time: 00:17)



Hello, everyone and welcome to this course on Modern Application Development. So, now that we know the basic concepts involved in testing, let us go a little deeper and look at a few different levels of testing that can be applied. And what we are going to do is primarily look at it in the context of a dashboard that hopefully all of you must be familiar with, which is the onlinedegree.iitm.ac.in, the actual dashboard used for that.

(Refer Slide Time: 00:39)


IIT Madras
BSc Degree

Initial requirements gathering

- Who are the stakeholders?
  - Students - log in and see courses
  - Admins - manage students
  - Teachers - update / manage course material ?
- Functionality
  - Each group has different needs
- Non-functional requirements
  - Page colour, font, logo

So, the first question that we can ask is I mean I need to write an app to implement the dashboard for the online degree application. And the first question that I would ask is what, you know, I need to do the initial requirements gathering and this initial requirements gathering happens before you write any code. And it is a very important part of the design process. So, there are several questions that need to be answered when you are doing requirements gathering.

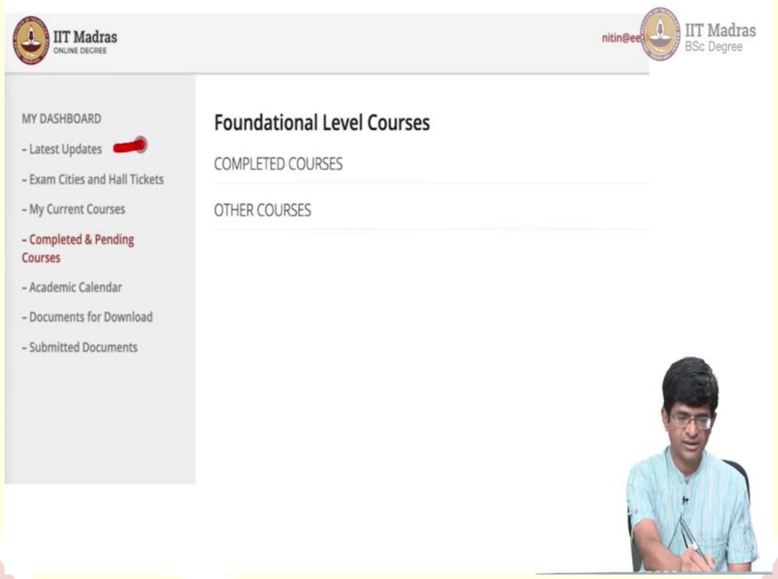
First and foremost is, who are the stakeholders? So, what does stakeholder mean? It means someone who has an interest in actually seeing this thing function. And in the case of the dashboard for the online degree, obviously students are major stakeholders, because all students are expected to log in, they should be able to see the courses for which they are registered, they should be able to add courses, drop courses, modifier type of a course, and also download certificate for courses that they have already completed.

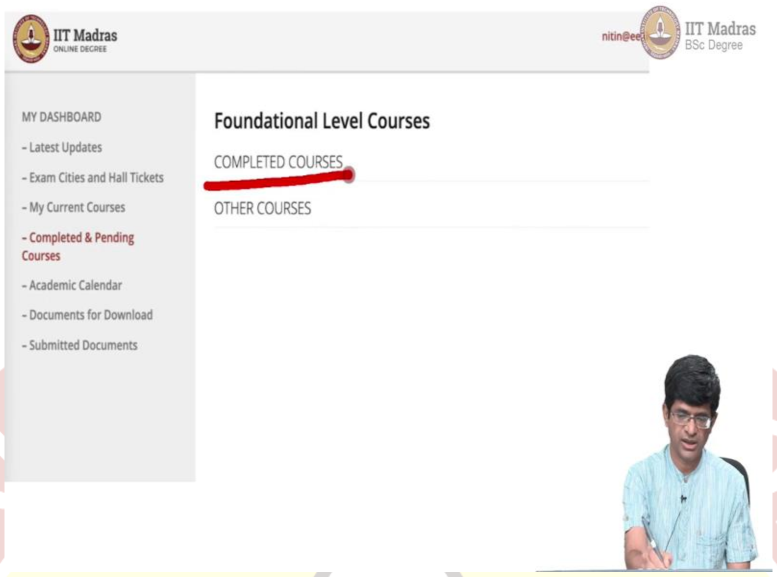
On the other hand, we also have the dashboard being used by certain other administrators, who have the responsibility for sort of managing the overall database, they need to be able to see all the students, they need to be able to make changes to the student database entries, if necessary. And typically, this will be a small number of people, I mean, you do not want too many people to have admin permissions, but at the same time, they should have that ability, if needed.

And then comes the teachers who would probably be updating course material over there, in the case of the online degree that is not done using the same dashboard, but it is a possibility, we might have chosen to also, do the same thing using the dashboard for the teachers as well. Now, now that you know the stakeholders, you need to sort of keeping each of them in mind, say, what is the functionality that each group needs? What are the kinds of things that a person should be able to do on the dashboard?

And then you might also have a certain set of non-functional requirements, which do not directly implement the behavior of the app, but are important in terms of the actual usability, examples would be the page color, the font size, the type of font that is used, where to put the logo, what logo should be implemented, all of those things are not directly implementing impacting the functionality, but are important for the overall sort of look and feel of the app.

(Refer Slide Time: 03:08)





IIT Madras
ONLINE DEGREE

nitin@ee

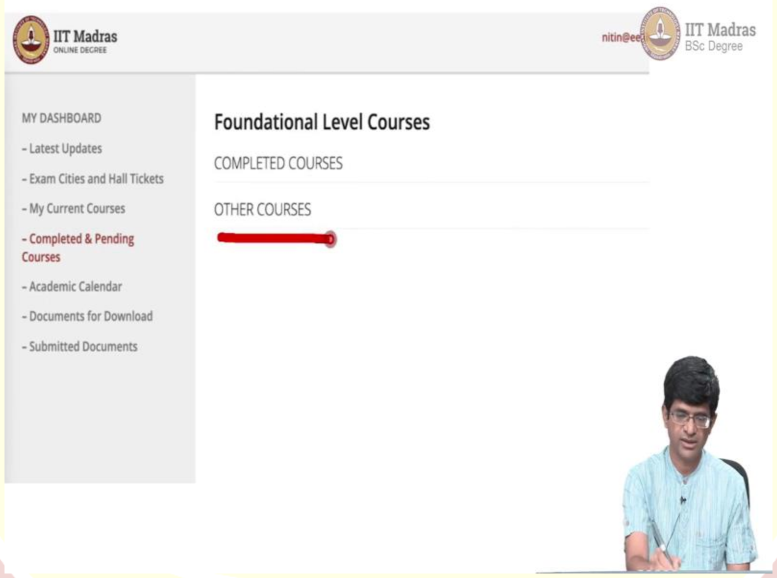
IIT Madras
BSc Degree

MY DASHBOARD
- Latest Updates
- Exam Cities and Hall Tickets
- My Current Courses
- Completed & Pending Courses
- Academic Calendar
- Documents for Download
- Submitted Documents

Foundational Level Courses

COMPLETED COURSES

OTHER COURSES



IIT Madras
ONLINE DEGREE

nitin@ee

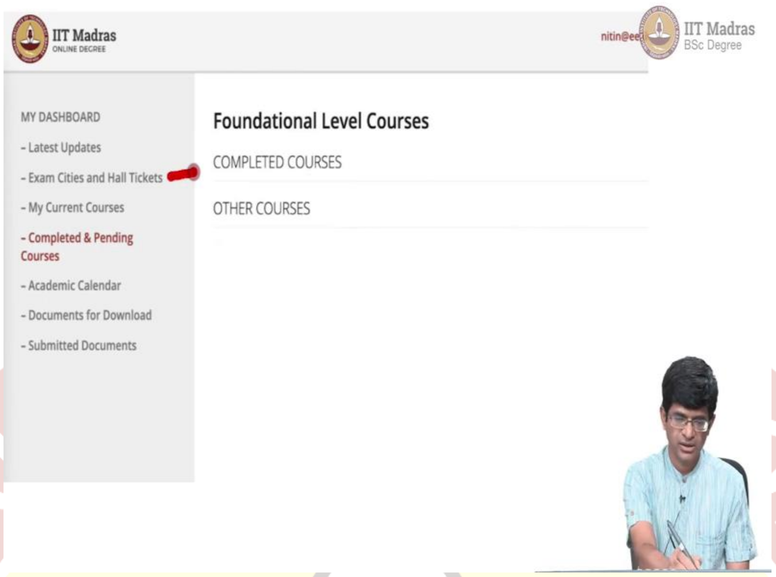
IIT Madras
BSc Degree

MY DASHBOARD
- Latest Updates
- Exam Cities and Hall Tickets
- My Current Courses
- Completed & Pending Courses
- Academic Calendar
- Documents for Download
- Submitted Documents

Foundational Level Courses

COMPLETED COURSES

OTHER COURSES

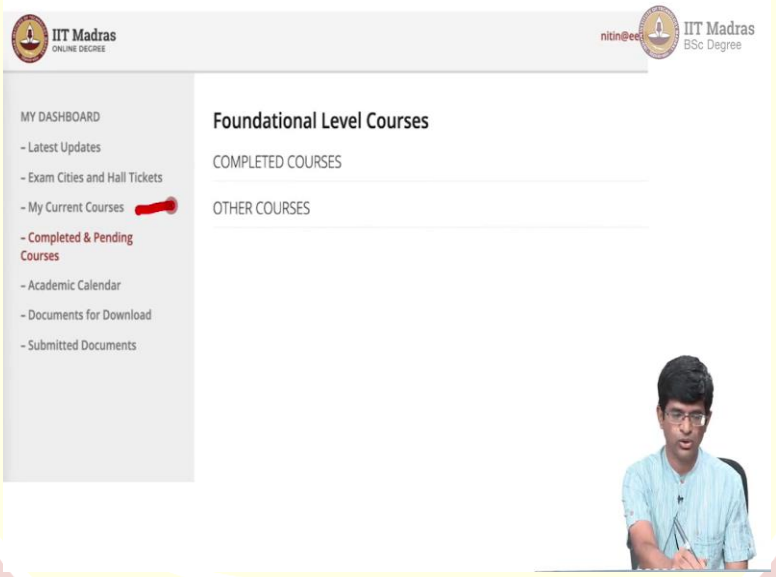


MY DASHBOARD
- Latest Updates
- Exam Cities and Hall Tickets
- My Current Courses
- Completed & Pending Courses
- Academic Calendar
- Documents for Download
- Submitted Documents

Foundational Level Courses

COMPLETED COURSES

OTHER COURSES

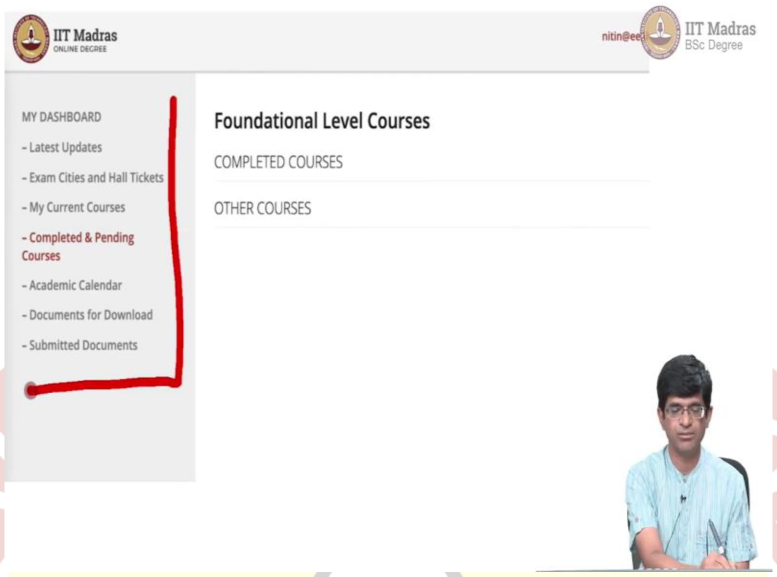

IIT Madras
ONLINE DEGREE
nitin@ee
IIT Madras
BSc Degree

MY DASHBOARD
- Latest Updates
- Exam Cities and Hall Tickets
- My Current Courses
- Completed & Pending Courses
- Academic Calendar
- Documents for Download
- Submitted Documents

Foundational Level Courses

COMPLETED COURSES

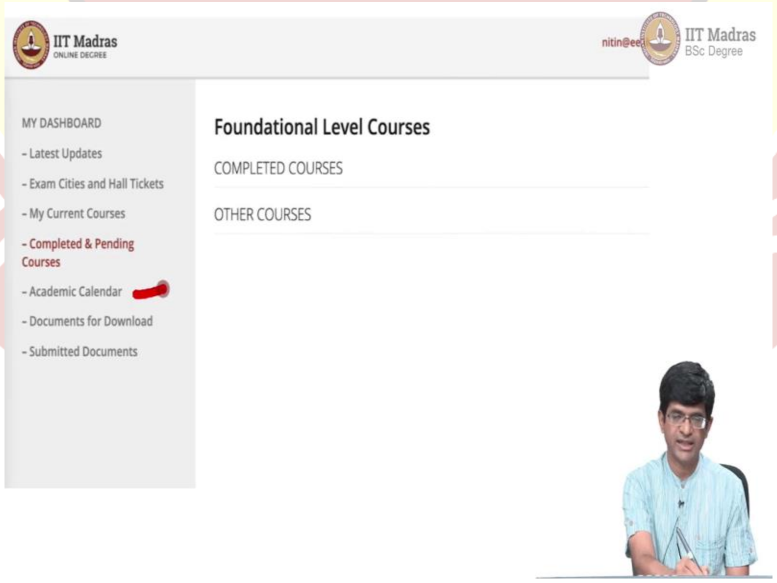
OTHER COURSES


IIT Madras
ONLINE DEGREE
nitin@ee
IIT Madras
BSc Degree
MY DASHBOARD
- Latest Updates
- Exam Cities and Hall Tickets
- My Current Courses
- Completed & Pending Courses
- Academic Calendar
- Documents for Download
- Submitted Documents

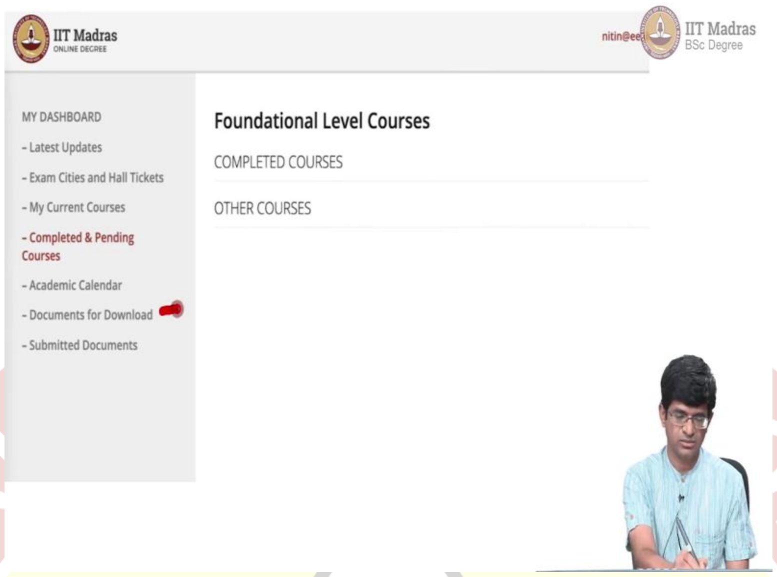
Foundational Level Courses
COMPLETED COURSES
OTHER COURSES

So, this is a screenshot of what the dashboard itself looks like, most of you should be familiar with this. And as you can see over here it has several entries on the left hand side, it has the updates, so you can basically go there and view updates, you can also look at what are the completed courses, any other courses that you have done, you can register for your exam hall preferences, download hall tickets you can also look at what are your current courses and make changes in particular you can drop a course if you want to.

(Refer Slide Time: 03:40)


IIT Madras
ONLINE DEGREE
nitin@ee
IIT Madras
BSc Degree
MY DASHBOARD
- Latest Updates
- Exam Cities and Hall Tickets
- My Current Courses
- Completed & Pending Courses
- Academic Calendar
- Documents for Download
- Submitted Documents

Foundational Level Courses
COMPLETED COURSES
OTHER COURSES


IIT Madras ONLINE DEGREE

nitin@ee

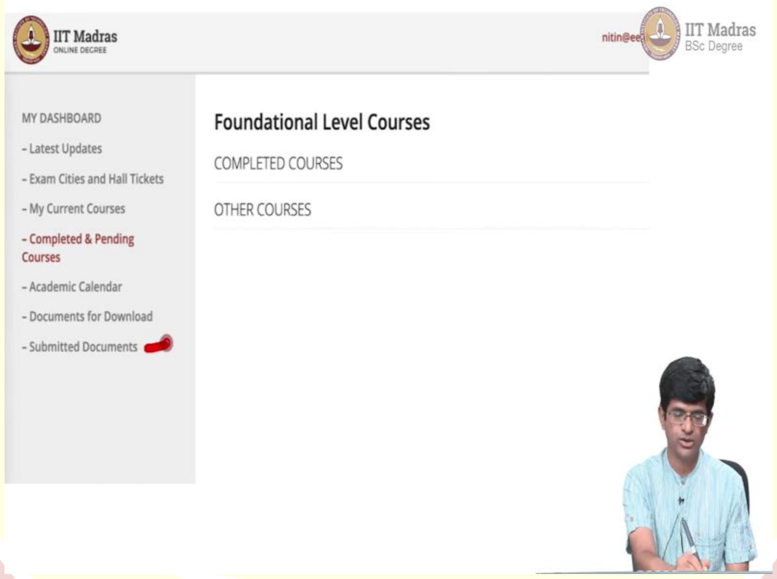
IIT Madras BSc Degree

MY DASHBOARD
- Latest Updates
- Exam Cities and Hall Tickets
- My Current Courses
- Completed & Pending Courses
- Academic Calendar
- Documents for Download
- Submitted Documents

Foundational Level Courses

COMPLETED COURSES

OTHER COURSES


IIT Madras ONLINE DEGREE

nitin@ee

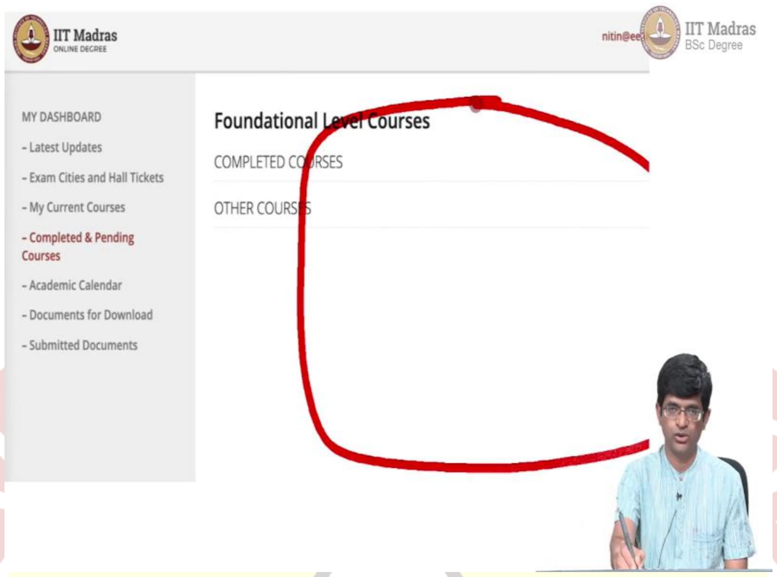
IIT Madras BSc Degree

MY DASHBOARD
- Latest Updates
- Exam Cities and Hall Tickets
- My Current Courses
- Completed & Pending Courses
- Academic Calendar
- Documents for Download
- Submitted Documents

Foundational Level Courses

COMPLETED COURSES

OTHER COURSES



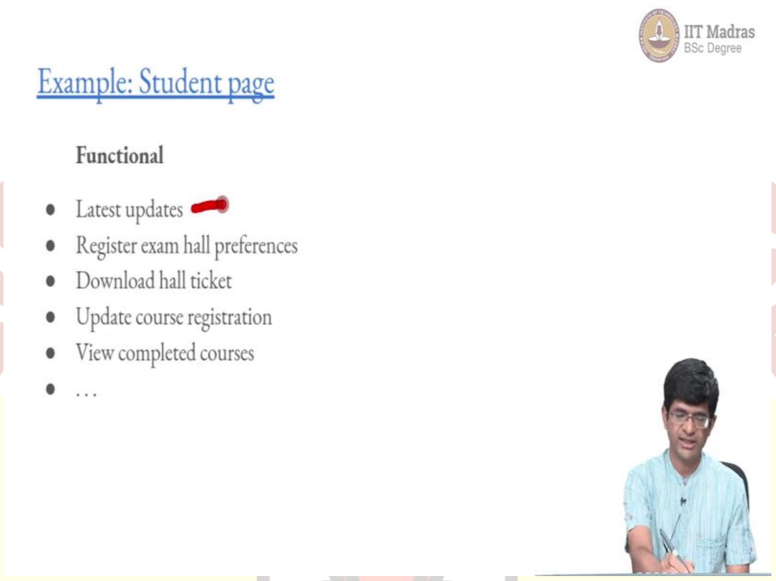
The academy calendar can be seen from here, certain documents that need to be downloaded from here can be downloaded, or if you need to submit any documents that can also be done. So, in other words, this left hand side that we are seeing over here is essentially some kind of navigation. It provides you with a clean interface that says look, you know, all the information that you need, it is sort of like a table of contents into this page.

And it says, okay the dashboard has all of these different functionalities, and this is how you can access the different things. And of course, you have the main sort of the view pane or here where most of the information will be displayed, along with that you also have something to sign out and the currently logged in person, some footer information that talks about the honor code, and some other information that is useful, and finally this logo on the top left.

So, this is a fairly typical structure of what a page would look like. Why do we use this kind of structure? It is once again, you know, a lot of usability research has gone into it, all of this refers to the user experience guidelines, Nielsen's work, for example that we discussed early on in the CSS part of the course, all of that sort of plays into how you go about designing a user interface.

And there are some general good guidelines which say that look, the navigation should be structured in a certain way use the header for providing these kinds of navigation elements, use the footer for some other things, dedicate the main part of the image or the browser screen to the actual content that you want to display. So, these are sort of general principles that are mostly followed in most cases. But more importantly, as far as we are concerned, there is also the functionality, what are the kinds of things that I want to be able to do, so all of these things over here correspond to different types of functionality.

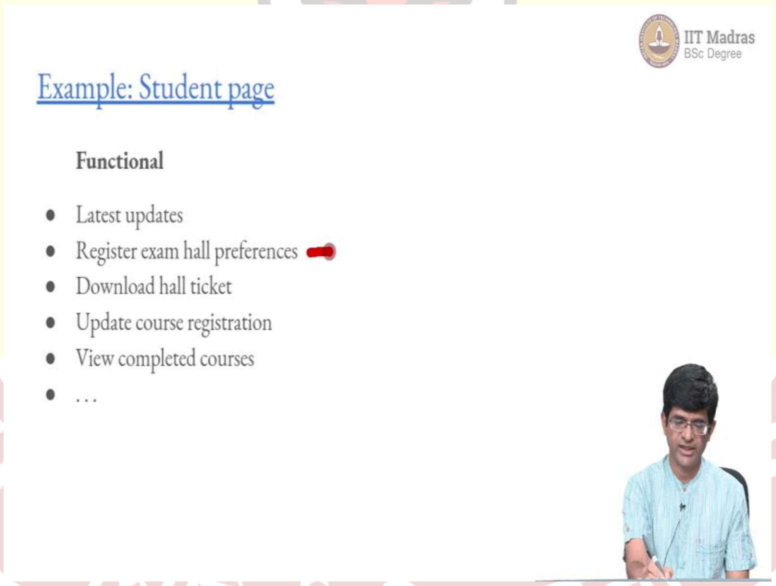
(Refer Slide Time: 05:38)



Example: Student page

Functional

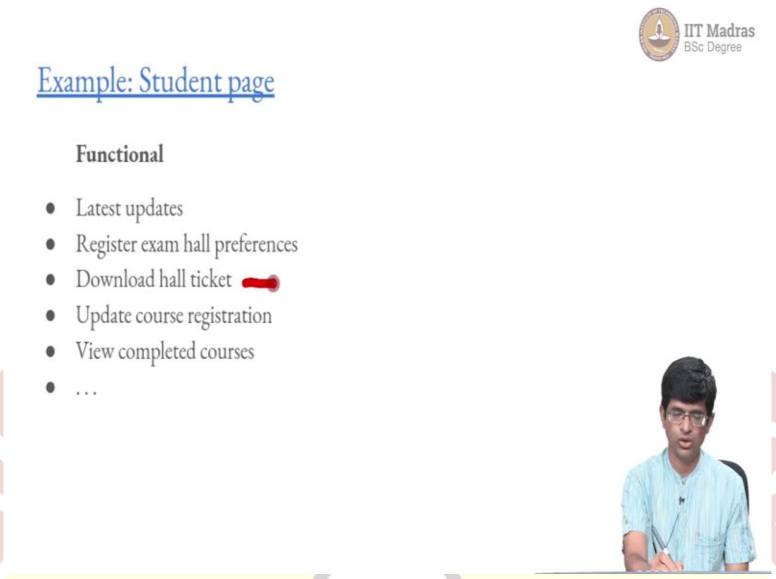
- Latest updates
- Register exam hall preferences
- Download hall ticket
- Update course registration
- View completed courses
- . . .



Example: Student page

Functional

- Latest updates
- Register exam hall preferences
- Download hall ticket
- Update course registration
- View completed courses
- . . .

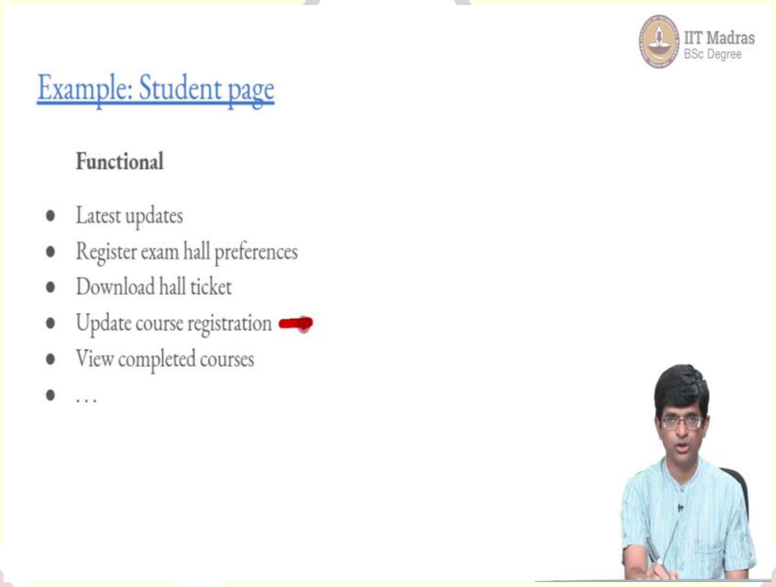


IIT Madras
BSc Degree

Example: Student page

Functional

- Latest updates
- Register exam hall preferences
- Download hall ticket
- Update course registration
- View completed courses
- . . .

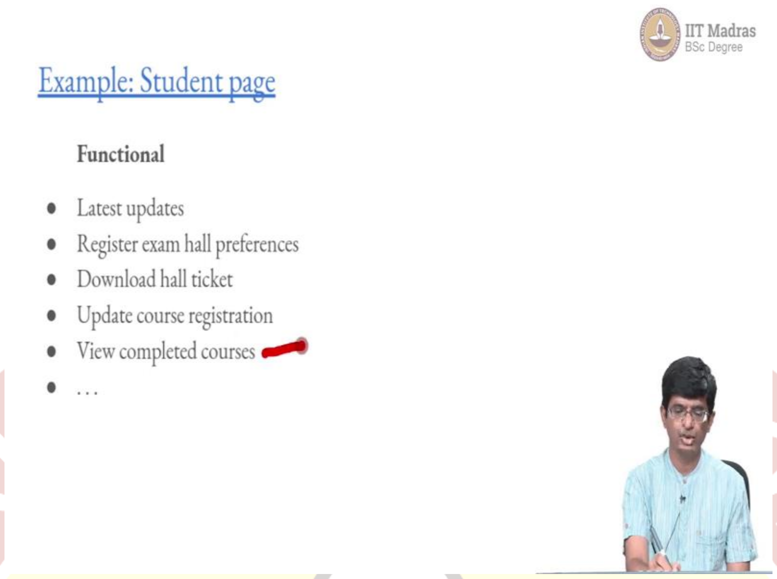


IIT Madras
BSc Degree

Example: Student page

Functional

- Latest updates
- Register exam hall preferences
- Download hall ticket
- Update course registration
- View completed courses
- . . .


IIT Madras
BSc Degree

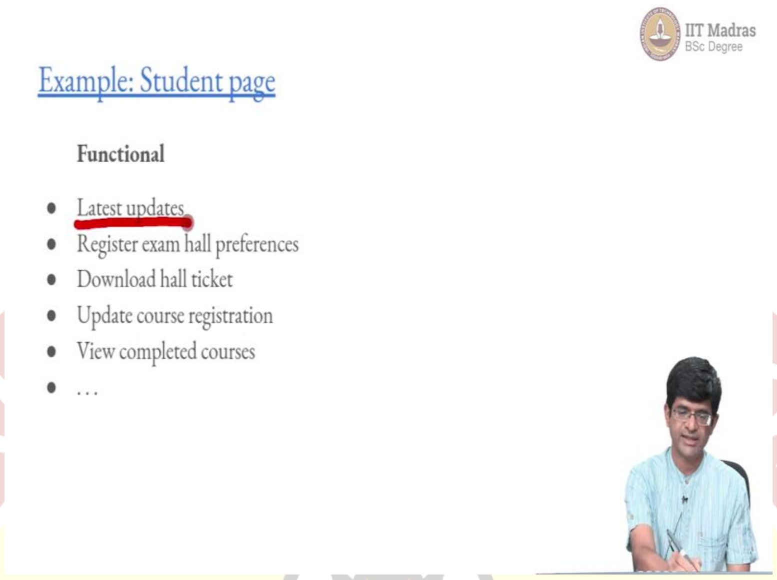
Example: Student page

Functional

- Latest updates
- Register exam hall preferences
- Download hall ticket
- Update course registration
- View completed courses
- . . .

So, we could, for example put down requirements and say that for a student page the functional requirements, I should be able to see the latest updates, I should be able to register my exam hall preferences, download the hall ticket, update my course registration, meaning that if I want to drop a course for example I should be able to do it from here, view the list of complete courses, so keeping in mind what we have already seen in this course, you might be able to already correlate this with different ways of implementing it. So, for example each of these essentially corresponds to some kind of a controller action. And each, in turn should result in a specific view.

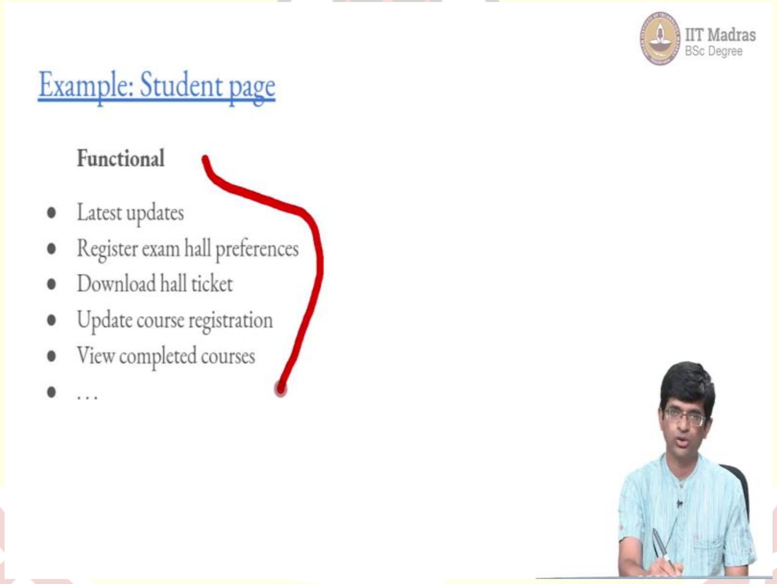
(Refer Slide Time: 06:20)



Example: Student page

Functional

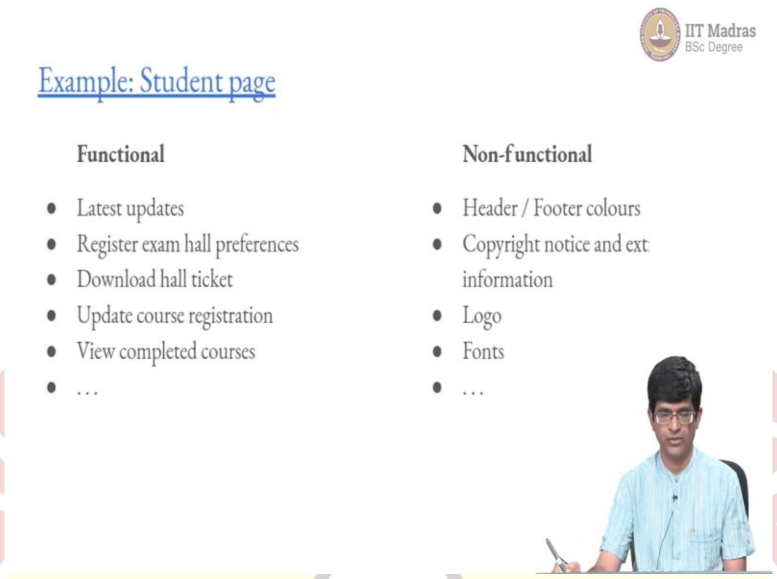
- Latest updates
- Register exam hall preferences
- Download hall ticket
- Update course registration
- View completed courses
- . . .



Example: Student page

Functional

- Latest updates
- Register exam hall preferences
- Download hall ticket
- Update course registration
- View completed courses
- . . .



Example: Student page

Functional

- Latest updates
- Register exam hall preferences
- Download hall ticket
- Update course registration
- View completed courses
- ...

Non-functional

- Header / Footer colours
- Copyright notice and ext information
- Logo
- Fonts
- ...

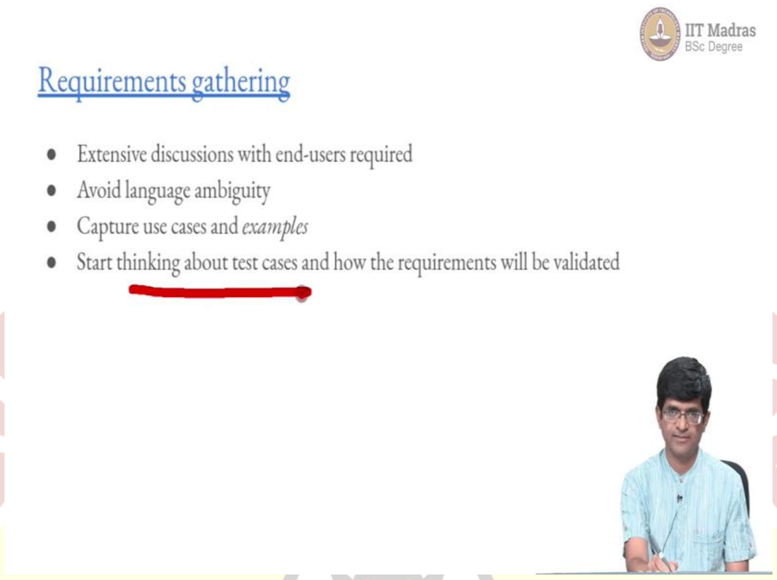
So, the latest updates, for example, I can imagine what it would be is, there will be a controller that gets triggered when I click on this, what would the controller do? It would need to reach into the database or not directly access to the database, but it would have to interact with some model that corresponds to news perhaps.

So, maybe I need to think about there has to be a model for how I store updates and news about the system. And it should be able to retrieve that set of latest updates, and there should be a view that basically shows it, in the main pane of the screen, right, maybe sorted by date, or by some kind of importance, or which is the most urgent piece of news something of that sort.

Similarly, for registering exam hall preferences, there will be a controller which basically is some kind of an edit controller, and that edit controller should be in such a way that it gives me a maybe a drop down list where I can select which exam hall I want to take and once I click on submit, it would post into that controller, and that in turn would do an update to the student model and would register this information saying that they the student wants to take the exam in this hall.

So, the moment I have this kind of functional requirement over here, you can start translating this into a set of controllers, and the corresponding views. Similarly, of course, there would be a bunch of non-functional requirements, what color to use, what font to use, where to put the logo, things of that sort do not directly impact the functionality, these are not for example, going to translate into controllers, but they are ultimately required for the good user experience as well.

(Refer Slide Time: 08:06)



Requirements gathering

- Extensive discussions with end-users required
- Avoid language ambiguity
- Capture use cases and *examples*
- Start thinking about test cases and how the requirements will be validated

So, requirements gathering is although it is not directly part of the testing process is a very important part of the overall system design. Extensive discussions with the end users are required, and in particular, an important thing that is needed over here is very often what we say in a language like English, versus what gets implemented in code are not the same.

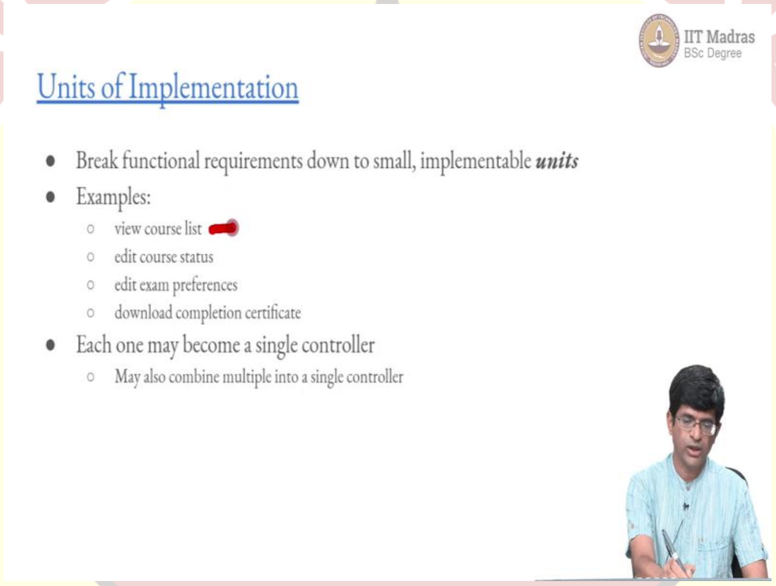
English and other human languages are ambiguous, meaning that very often it is possible to interpret the same thing in multiple different ways. So, avoiding that kind of language ambiguity is very important. So, for example, when I say oh you know the student should be able to update their course registration. Does that mean a student can add a course? Does that mean they can add a course at any point? Does that mean they can go and change somebody else's registration?

Lots of questions over there, need to be like, given very clear answers in order to get a complete specification. So, that requirements needs to have all of those things answered properly. And very often the best way to do this is by so called use cases, and examples of how the system is supposed to be used.

So, if I can actually create a set of use cases and say look a student wants to add a course, this is the set of steps that they would need to follow, it becomes a lot clearer and easier for the person implementing the code to actually get it right. Which means that right at this point, when you are doing requirements gathering, you should start thinking about test cases.

So, for example if I know that I need to have the ability to add a course, I can start thinking okay how would I test that behavior? Maybe I can create a dummy database which has one student and one course and then say for the student to add this course, they should be able to call this particular endpoint, when they do that, they will get this kind of screen, the screen should have these elements in it, the elements should respond like this, when I click on them, all of those are tests that you can start thinking about and straight away put down the code corresponding, put down the tests corresponding to those.

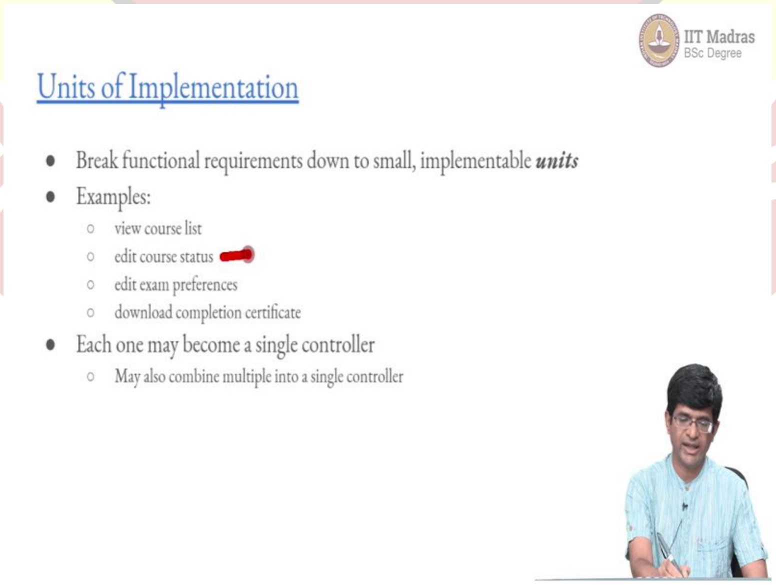
(Refer Slide Time: 10:20)



Units of Implementation

- Break functional requirements down to small, implementable ***units***
- Examples:
  - view course list
  - edit course status
  - edit exam preferences
  - download completion certificate
- Each one may become a single controller
  - May also combine multiple into a single controller

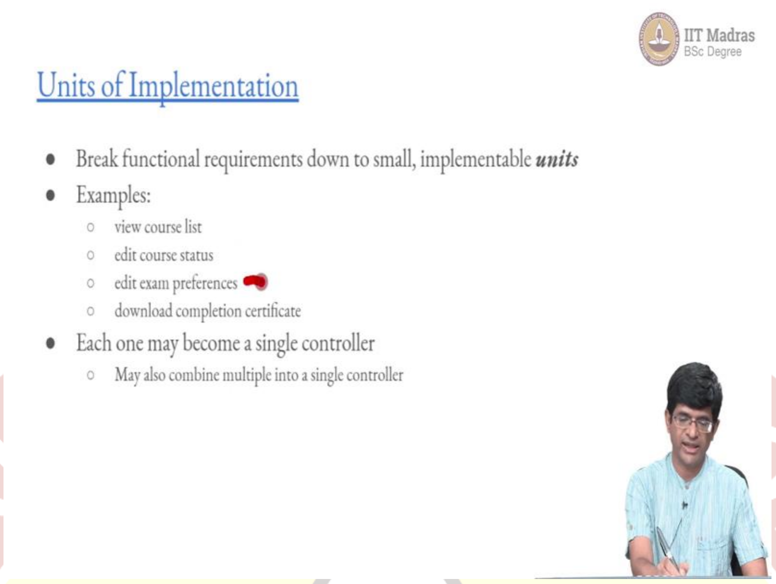
IIT Madras BSc Degree



Units of Implementation

- Break functional requirements down to small, implementable ***units***
- Examples:
  - view course list
  - edit course status
  - edit exam preferences
  - download completion certificate
- Each one may become a single controller
  - May also combine multiple into a single controller

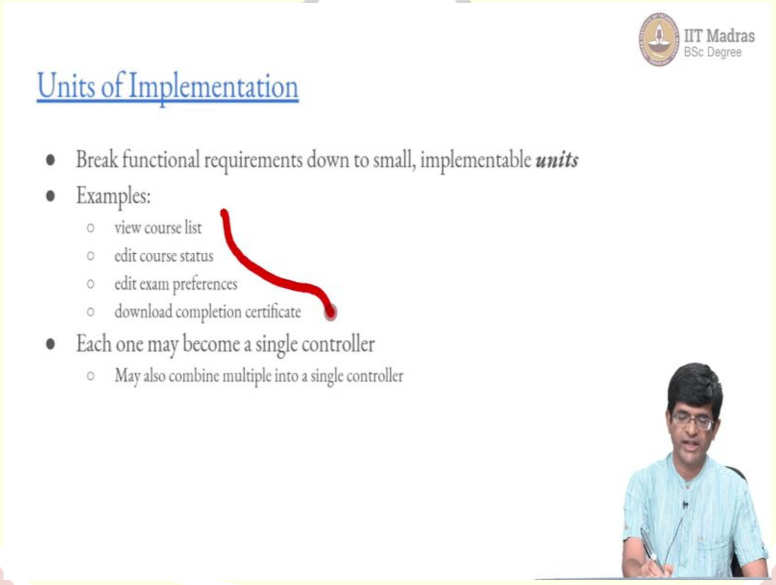
IIT Madras BSc Degree



IIT Madras
BSc Degree

Units of Implementation

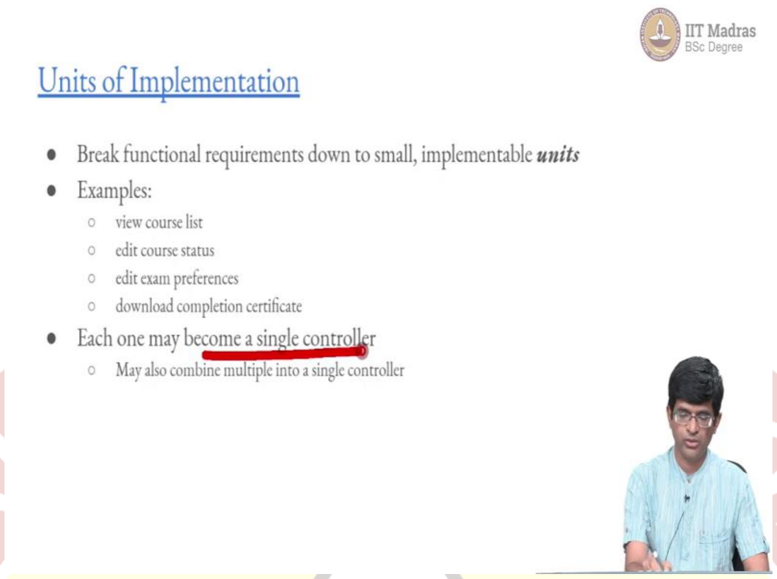
- Break functional requirements down to small, implementable ***units***
- Examples:
  - view course list
  - edit course status
  - edit exam preferences
  - download completion certificate
- Each one may become a single controller
  - May also combine multiple into a single controller



IIT Madras
BSc Degree

Units of Implementation

- Break functional requirements down to small, implementable ***units***
- Examples:
  - view course list
  - edit course status
  - edit exam preferences
  - download completion certificate
- Each one may become a single controller
  - May also combine multiple into a single controller



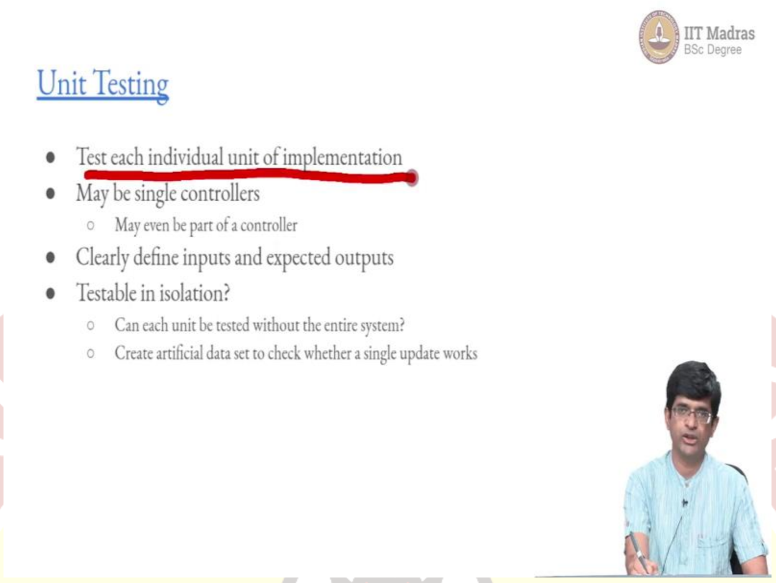
Units of Implementation

- Break functional requirements down to small, implementable ***units***
- Examples:
  - view course list
  - edit course status
  - edit exam preferences
  - download completion certificate
- Each one may become a single controller
  - May also combine multiple into a single controller

Now, once you have that, it usually makes sense to start breaking the requirements down into small units of implementation. An example would be, how would I view a list of courses? This is one unit. Similarly, how do I edit a course status? How do I edit an exam preference? Each one of these effectively corresponds to one controller, maybe one or more views, and interaction with one or more models, which is why we would call these as sort of individual units that need to get implemented. And what that means is? Most of them would translate into a single controller, but the important part is now each of those is a unit that needs to get implemented.

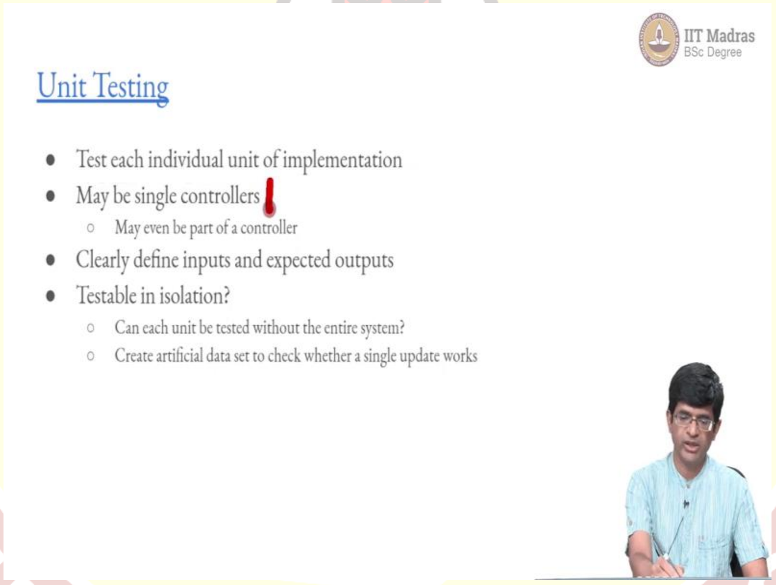
The good part is, it means that you can even break this up and say if you have a team of three people, you can tell each one of them you implement the view course list, you implement the added course status, you implement edit exam preferences, and because of the fact that there is nice compartmentalization between each of these, then they all put their code together, it should still work.

(Refer Slide Time: 11:30)



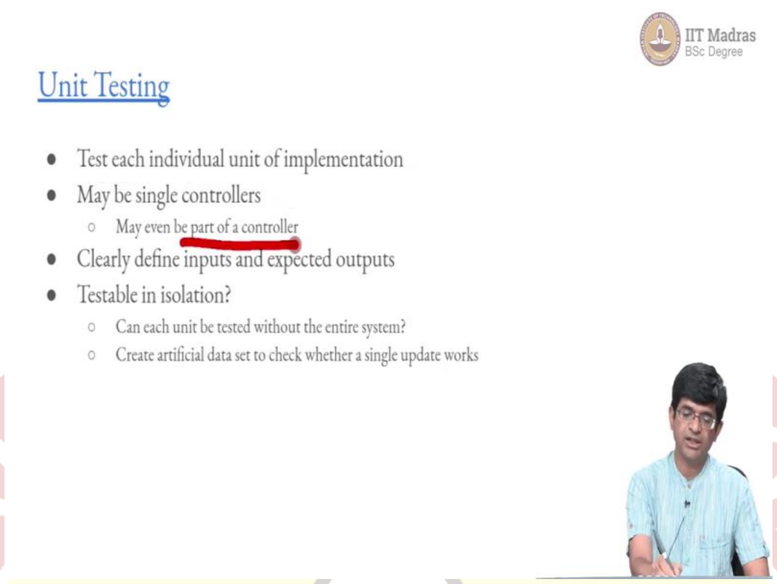
Unit Testing

- Test each individual unit of implementation
- May be single controllers
  - May even be part of a controller
- Clearly define inputs and expected outputs
- Testable in isolation?
  - Can each unit be tested without the entire system?
  - Create artificial data set to check whether a single update works



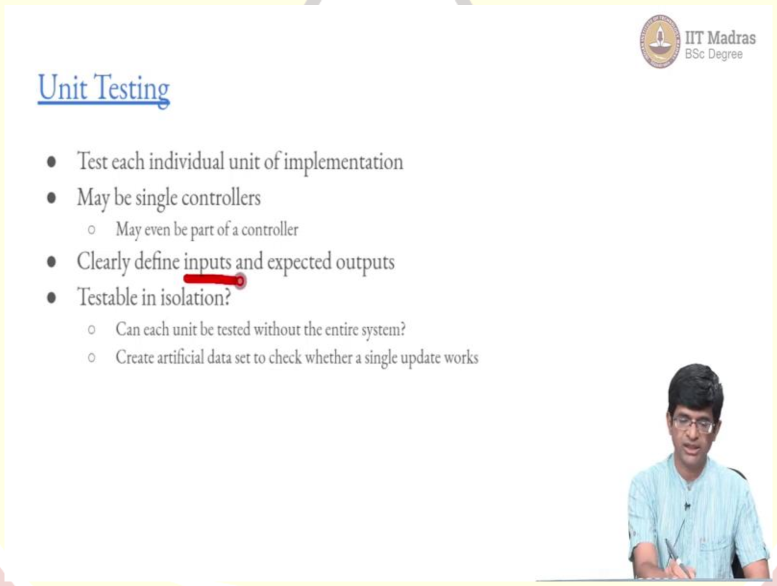
Unit Testing

- Test each individual unit of implementation
- May be single controllers
  - May even be part of a controller
- Clearly define inputs and expected outputs
- Testable in isolation?
  - Can each unit be tested without the entire system?
  - Create artificial data set to check whether a single update works



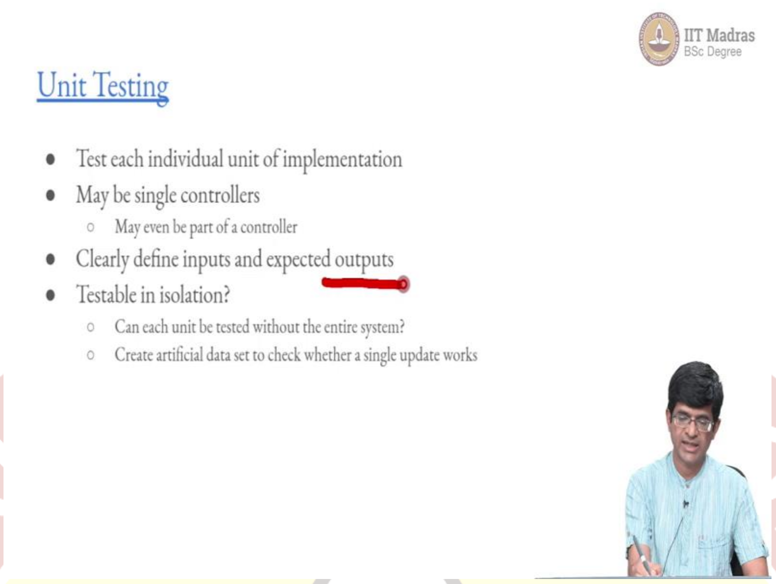
Unit Testing

- Test each individual unit of implementation
- May be single controllers
  - May even be part of a controller
- Clearly define inputs and expected outputs
- Testable in isolation?
  - Can each unit be tested without the entire system?
  - Create artificial data set to check whether a single update works



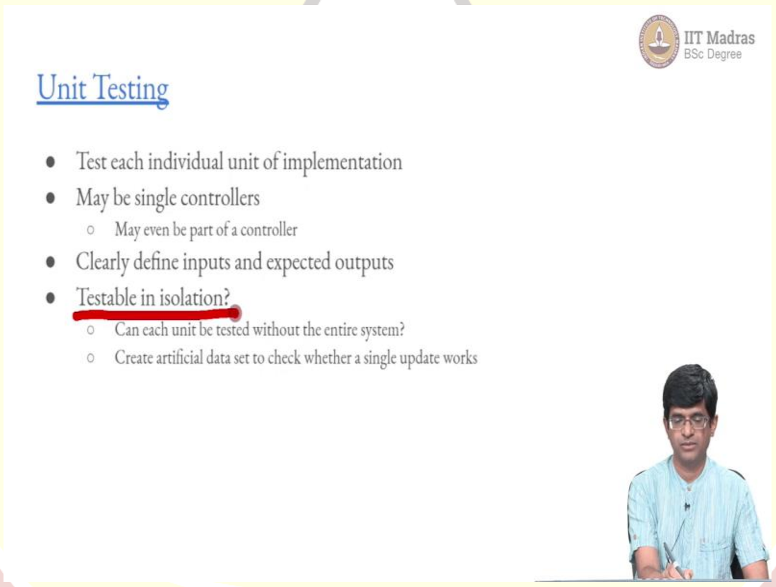
Unit Testing

- Test each individual unit of implementation
- May be single controllers
  - May even be part of a controller
- Clearly define inputs and expected outputs
- Testable in isolation?
  - Can each unit be tested without the entire system?
  - Create artificial data set to check whether a single update works



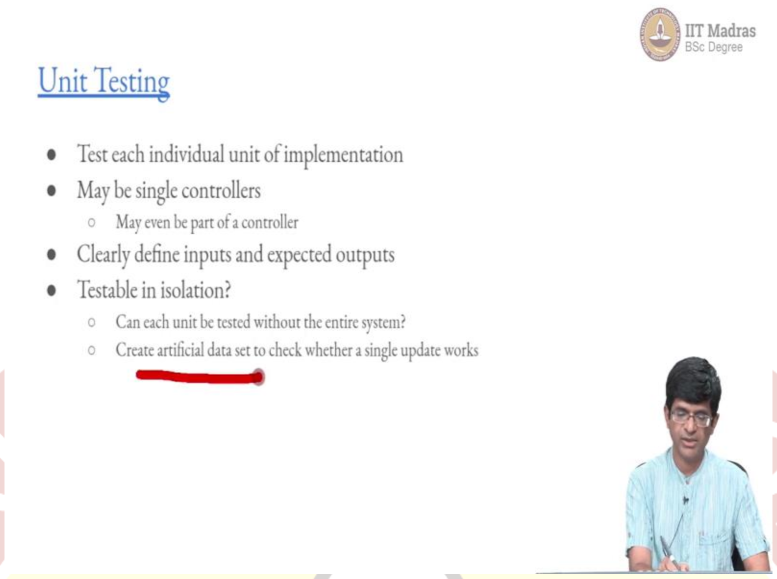
Unit Testing

- Test each individual unit of implementation
- May be single controllers
  - May even be part of a controller
- Clearly define inputs and expected outputs
- Testable in isolation?
  - Can each unit be tested without the entire system?
  - Create artificial data set to check whether a single update works



Unit Testing

- Test each individual unit of implementation
- May be single controllers
  - May even be part of a controller
- Clearly define inputs and expected outputs
- Testable in isolation?
  - Can each unit be tested without the entire system?
  - Create artificial data set to check whether a single update works


IIT Madras BSc Degree

Unit Testing

- Test each individual unit of implementation
- May be single controllers
  - May even be part of a controller
- Clearly define inputs and expected outputs
- Testable in isolation?
  - Can each unit be tested without the entire system?
  - Create artificial data set to check whether a single update works

But a unit test just tests a single individual unit of the implementation. So, usually it would be a single controller, it may even be a part of a controller, for example, I might have one controller whose job is, if it's a get request, then you know, display the edit page, if it is a post request then actually update and send back. And I might create two different tests, one of which tests the get request and one which request tests the post request.

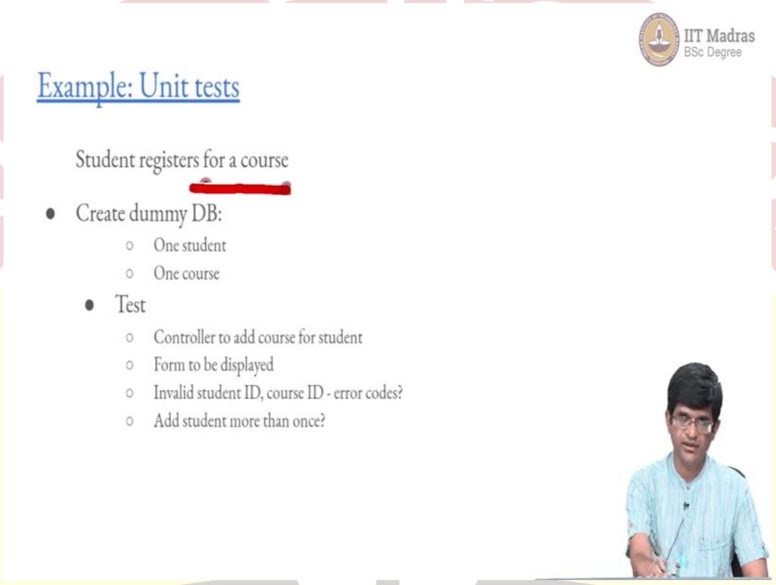
And in fact, maybe multiple tests for the post, because I want to also see what will happen if I give invalid information, things like that. This, which means that I should be able to clearly define the inputs and the expected outputs for each of these individual units, and one question that comes up out there is, is it possible to test that unit in isolation?

What do I mean by that? I have students and I have courses and I want to be able to add a student to a course, and say specify their marks for that course. Now, the problem is that part which involves testing the marks, now adding a student can be tested by itself, adding a course can be tested by itself, but in order to be able to add a student to a course and specify their marks, I need to have students and courses already in the system.

Which means that I should be able to create some kind of an artificial data set that already has a student, already has a course, and only then I can run the test for adding a student to a course. So, that is an example of a test which cannot be done completely in isolation, it requires certain other parts to already be implemented.

But still, how do I test it? Do I need to be going around making function calls to all the other, you know add student, add course and so on? Or can I create an artificial data set, pre populate with some information, and then just use that to see whether a single update to the system works? All these are different ways that you need to think about the unit test cases.

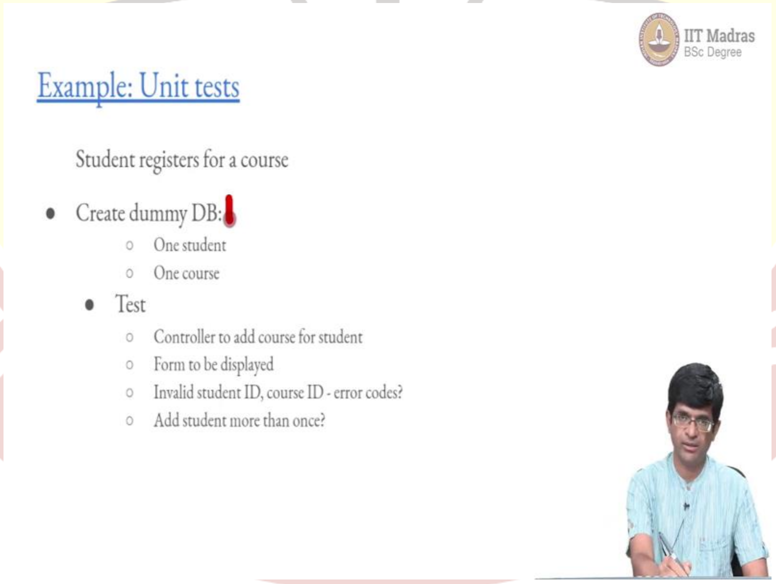
(Refer Slide Time: 13:31)



Example: Unit tests

Student registers for a course

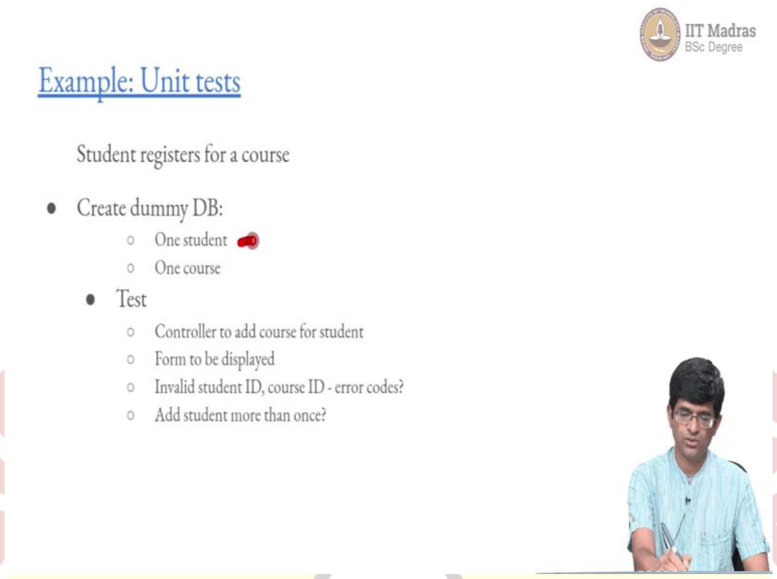
- Create dummy DB:
    - One student
    - One course
- Test
    - Controller to add course for student
    - Form to be displayed
    - Invalid student ID, course ID - error codes?
    - Add student more than once?



Example: Unit tests

Student registers for a course

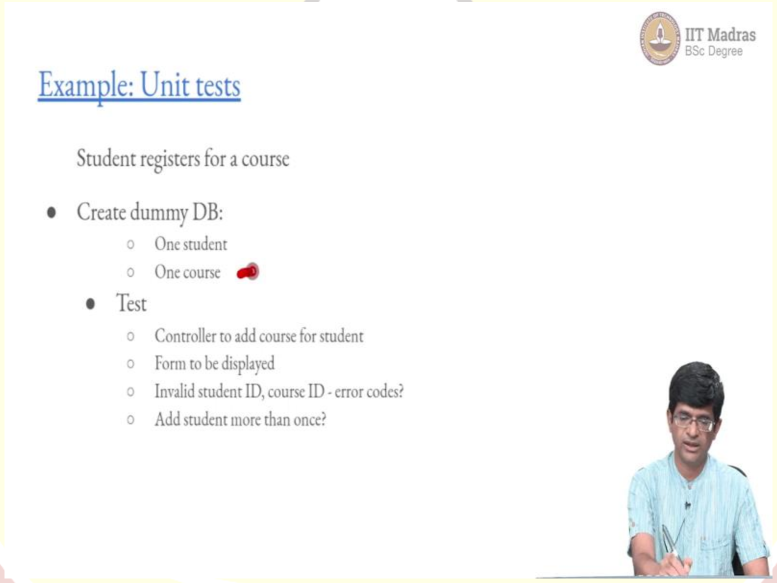
- Create dummy DB:
    - One student
    - One course
- Test
    - Controller to add course for student
    - Form to be displayed
    - Invalid student ID, course ID - error codes?
    - Add student more than once?



Example: Unit tests

Student registers for a course

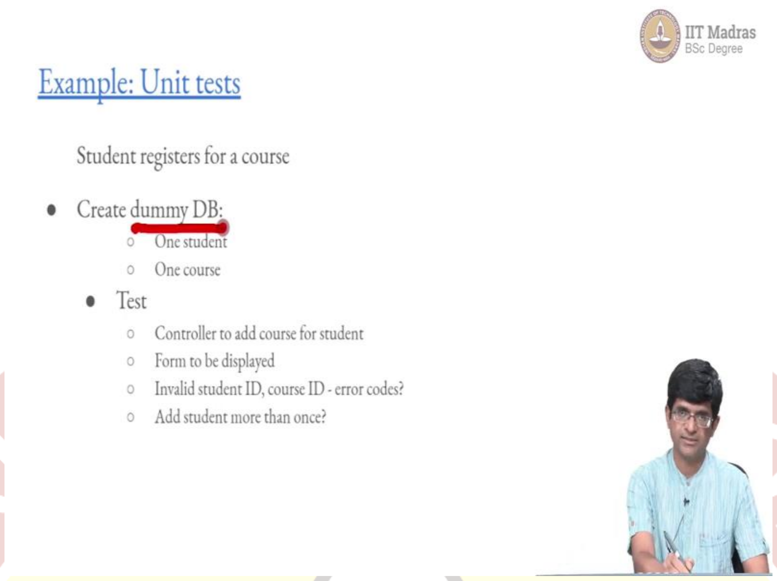
- Create dummy DB:
  - One student
  - One course
- Test
  - Controller to add course for student
  - Form to be displayed
  - Invalid student ID, course ID - error codes?
  - Add student more than once?



Example: Unit tests

Student registers for a course

- Create dummy DB:
  - One student
  - One course
- Test
  - Controller to add course for student
  - Form to be displayed
  - Invalid student ID, course ID - error codes?
  - Add student more than once?



Example: Unit tests

Student registers for a course

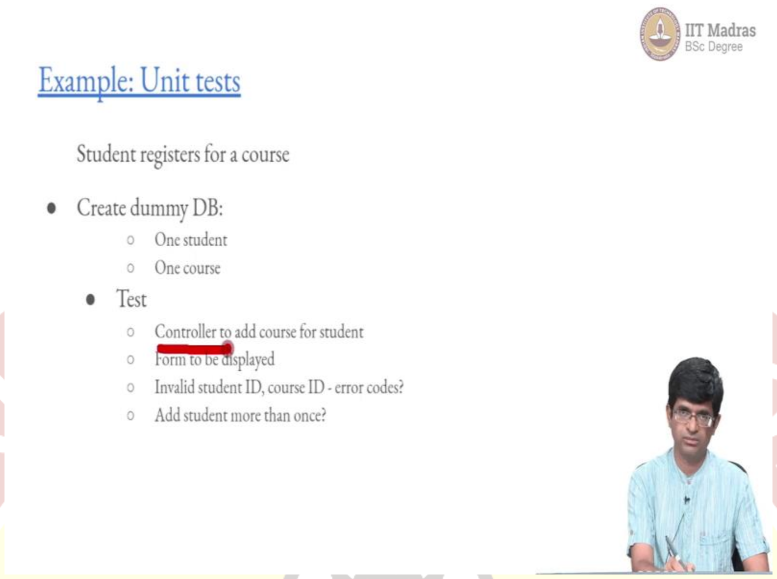
- Create dummy DB:
  - One student
  - One course
- Test
  - Controller to add course for student
  - Form to be displayed
  - Invalid student ID, course ID - error codes?
  - Add student more than once?

So, an example like I said, would be that a student wants to register for a course, and you want to sort of see how that would be done. Which means that I need to have one student and I need to have a course already in my system. So, I can create a dummy database, which has one student and one course in it, why am I creating this dummy database?

Because keep in mind that the test is something that has to be repeatable, I do not want the test to be run against live data, after I have already sort of deployed the online degree web app, I cannot run further tests on it, I can go and check whether user number 1 can be added to course number 10.

Because user number 1 is now an actual person who is registering for courses. And in general, it is a bad idea to meddle with the live database at any given point. So, instead what I do is I create a dummy database, which has one student one course that is enough for the test that I am trying to run.

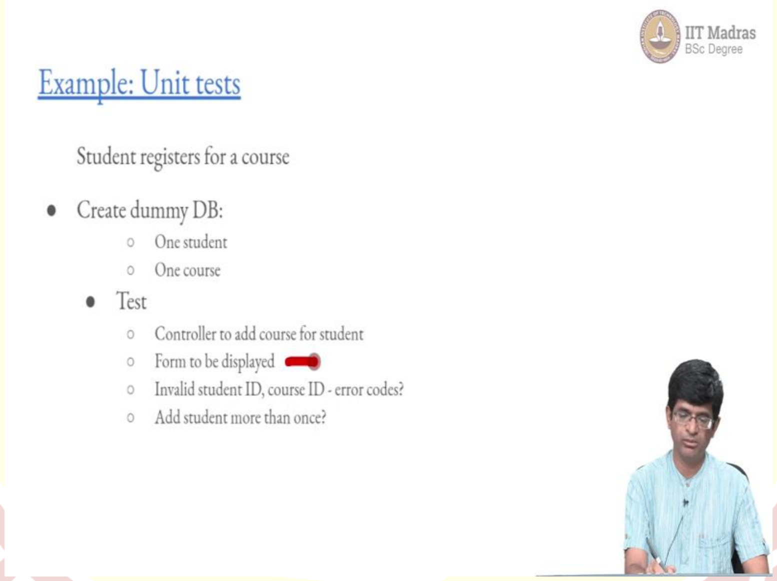
(Refer Slide Time: 14:35)



Example: Unit tests

Student registers for a course

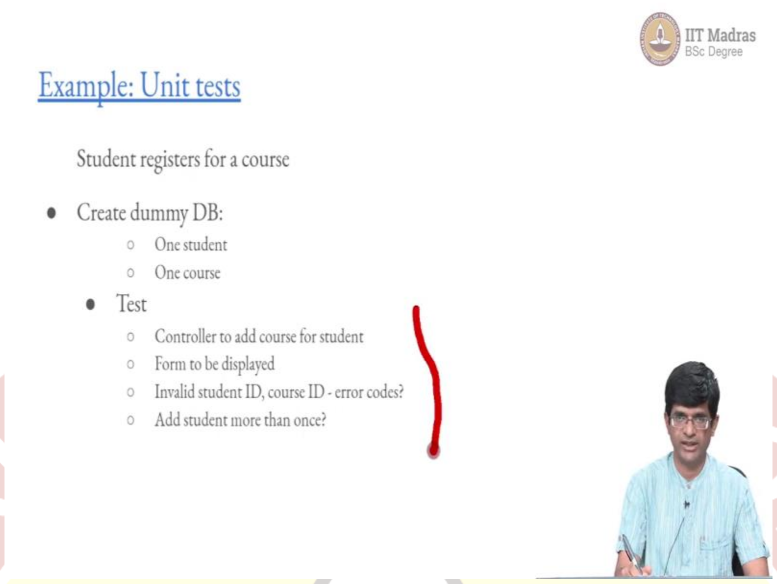
- Create dummy DB:
  - One student
  - One course
- Test
  - Controller to add course for student
  - Form to be displayed
  - Invalid student ID, course ID - error codes?
  - Add student more than once?



Example: Unit tests

Student registers for a course

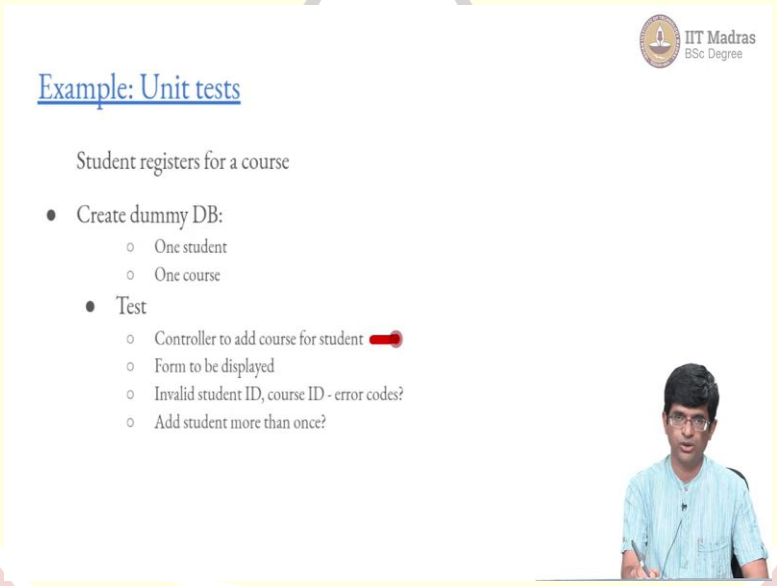
- Create dummy DB:
  - One student
  - One course
- Test
  - Controller to add course for student
  - Form to be displayed
  - Invalid student ID, course ID - error codes?
  - Add student more than once?



Example: Unit tests

Student registers for a course

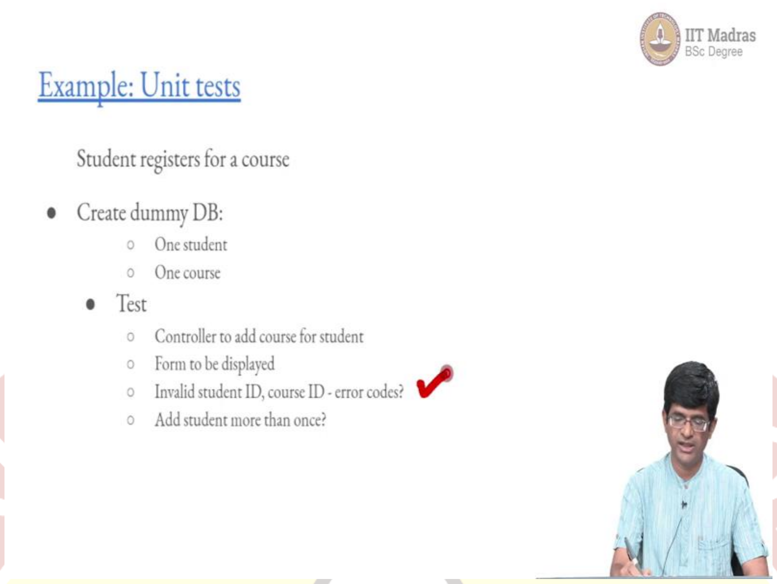
- Create dummy DB:
    - One student
    - One course
- Test
    - Controller to add course for student
    - Form to be displayed
    - Invalid student ID, course ID - error codes?
    - Add student more than once?



Example: Unit tests

Student registers for a course

- Create dummy DB:
    - One student
    - One course
- Test
    - Controller to add course for student
    - Form to be displayed
    - Invalid student ID, course ID - error codes?
    - Add student more than once?

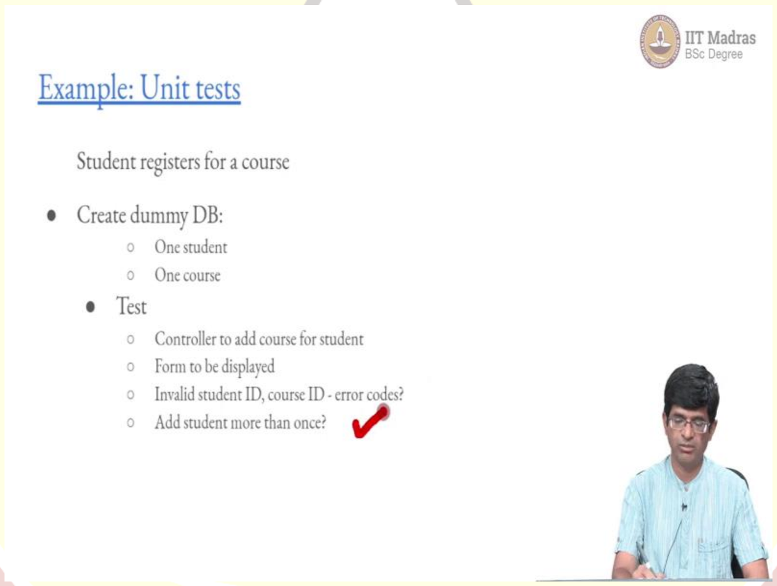


IIT Madras
BSc Degree

Example: Unit tests

Student registers for a course

- Create dummy DB:
  - One student
  - One course
  - Test
    - Controller to add course for student
    - Form to be displayed
    - Invalid student ID, course ID - error codes? ✓
    - Add student more than once?



IIT Madras
BSc Degree

Example: Unit tests

Student registers for a course

- Create dummy DB:
  - One student
  - One course
  - Test
    - Controller to add course for student
    - Form to be displayed
    - Invalid student ID, course ID - error codes?
    - Add student more than once? ✓

And I need to invoke that controller, that is basically all that is going to happen, when I finally sort of use the user interface and click on something, it will activate that controller and say okay for this student at this course. Once I have done that, I need to check a few things, is the form being displayed properly, how do I check that? I retrieved HTML and within that I can check okay you know was this particular tag present, was this form present, was the input field present? If so then I can say yes, you know, the form actually came out right. What happens if I enter or rather do a post with invalid information? It is not the valid user, it is not the correct person, it is not the correct course ID, do I get the correct error codes? Or does the system crash?

What happens if I try adding a student more than once to the same course? Once again, there should be some kind of an error or something that is raised, all of these are valid tests. So, the moment I take something as simple as student registers for a course, I can see that I can start putting down a bunch of tests for that.

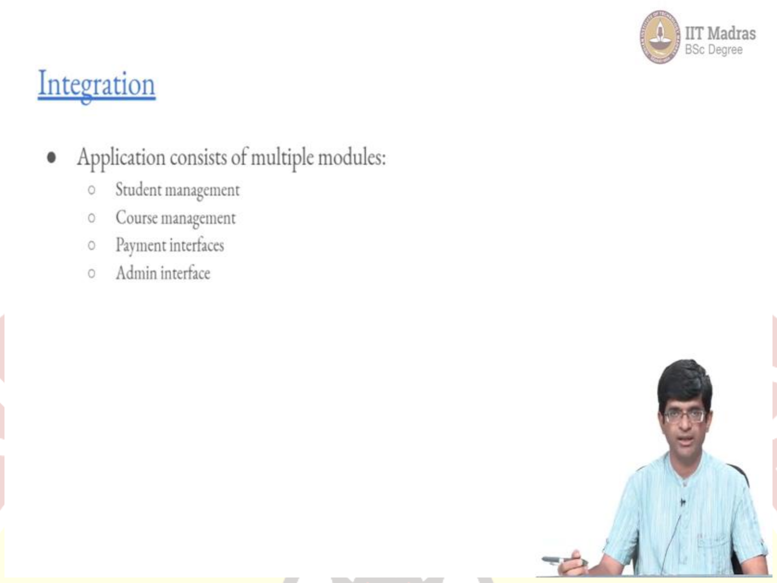
And this approach is, there are variants of this that are used in different styles of software development, but broadly it falls under this category of what is called test driven development, what you do is, before writing your code, you actually put down the tests that you are going to run. And you in fact, create all the tests, and to start with what happens is all the tests fail.

So, you basically run this test suite, and it basically goes through and says, all the tests failed. After that you start solving the problems, you implement the first thing, you create a controller to add the course. Now, how would that be tested? When I invoke the controller, what response code does it give? Tick, one test has been passed.

Next test is the form being displayed properly, put in the data corresponding to the form tick, next test has been passed. So, in the same way I can keep checking off each of these tests one by one, and finally I find that all the tests have passed, the good news is this is now a regression test suite.

And what I mean by that is, later on when you go and start making other modifications, let us say you are writing an admin interface, I need to make sure that that does not break anything corresponding to a student registering for a course. If any one of these test fails, that will be called a regression. And I need to go back, find out why that happened, and ensure that it is something that can be fixed.

(Refer Slide Time: 17:07)

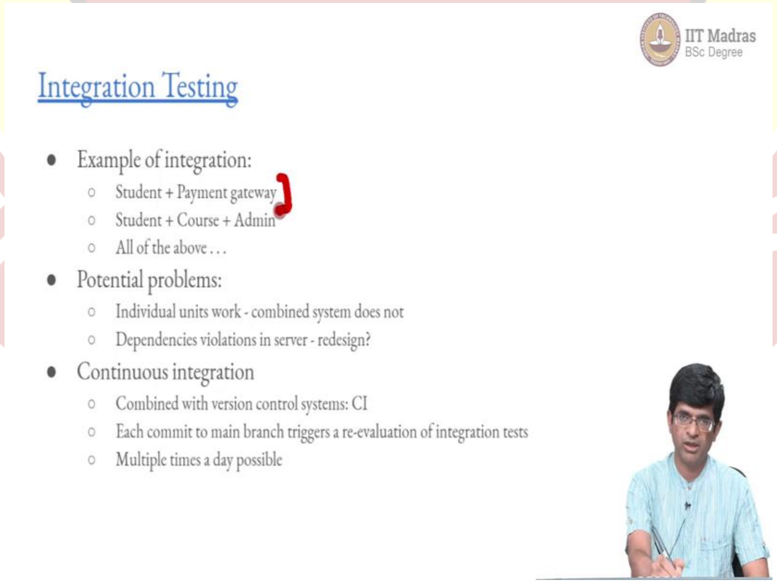


Integration

- Application consists of multiple modules:
  - Student management
  - Course management
  - Payment interfaces
  - Admin interface

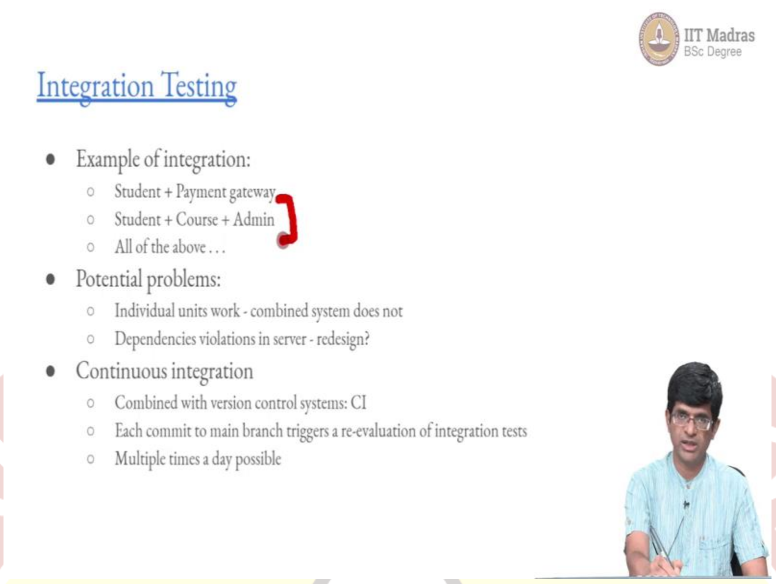
Now, assuming that you have done unit testing, and you have started implementing all of these units properly, at some level you need to start integrating these things. Examples of integration would be when an application consists of multiple modules, you might have student management, course management, payment interfaces, admin interfaces, you need to get all of those together and combine them.

(Refer Slide Time: 17:30)



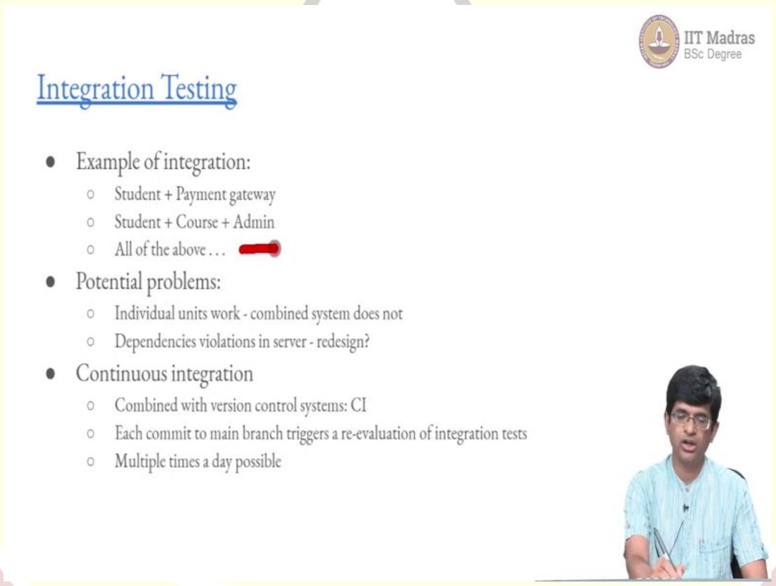
Integration Testing

- Example of integration:
  - Student + Payment gateway
  - Student + Course + Admin
  - All of the above . . .
- Potential problems:
  - Individual units work - combined system does not
  - Dependencies violations in server - redesign?
- Continuous integration
  - Combined with version control systems: CI
  - Each commit to main branch triggers a re-evaluation of integration tests
  - Multiple times a day possible



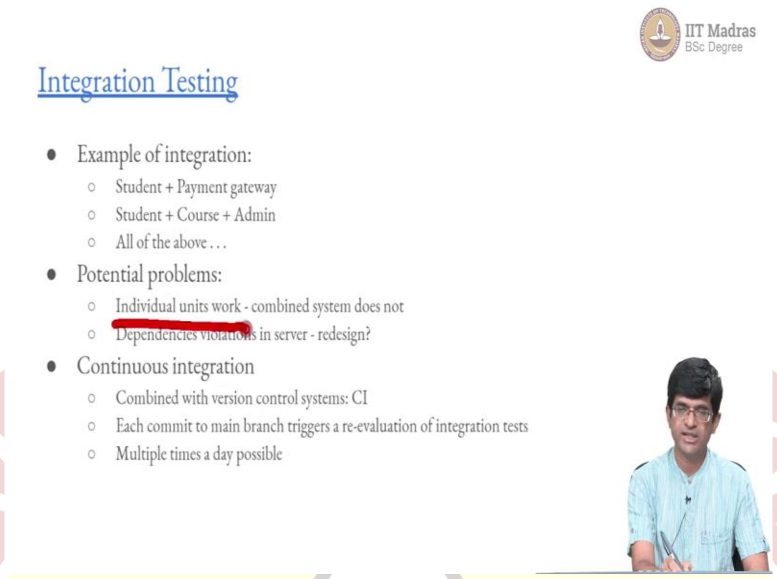
Integration Testing

- Example of integration:
  - Student + Payment gateway
  - Student + Course + Admin
  - All of the above . . .
- Potential problems:
  - Individual units work - combined system does not
  - Dependencies violations in server - redesign?
- Continuous integration
  - Combined with version control systems: CI
  - Each commit to main branch triggers a re-evaluation of integration tests
  - Multiple times a day possible



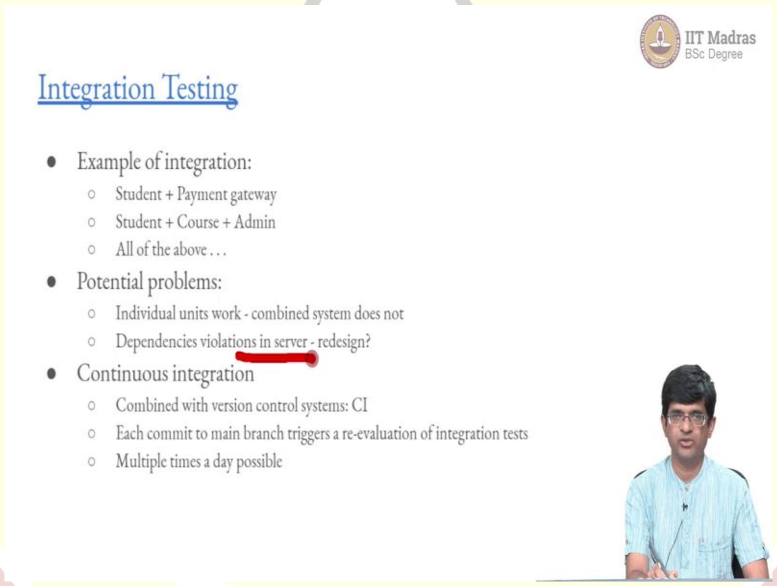
Integration Testing

- Example of integration:
  - Student + Payment gateway
  - Student + Course + Admin
  - All of the above . . .
- Potential problems:
  - Individual units work - combined system does not
  - Dependencies violations in server - redesign?
- Continuous integration
  - Combined with version control systems: CI
  - Each commit to main branch triggers a re-evaluation of integration tests
  - Multiple times a day possible



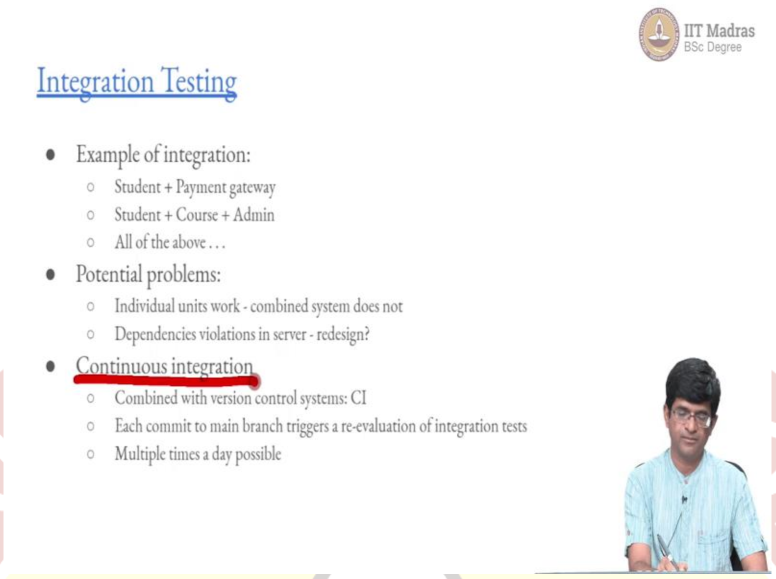
Integration Testing

- Example of integration:
  - Student + Payment gateway
  - Student + Course + Admin
  - All of the above . . .
- Potential problems:
  - Individual units work - combined system does not
  - Dependencies violations in server - redesign?
- Continuous integration
  - Combined with version control systems: CI
  - Each commit to main branch triggers a re-evaluation of integration tests
  - Multiple times a day possible



Integration Testing

- Example of integration:
  - Student + Payment gateway
  - Student + Course + Admin
  - All of the above . . .
- Potential problems:
  - Individual units work - combined system does not
  - Dependencies violations in server - redesign?
- Continuous integration
  - Combined with version control systems: CI
  - Each commit to main branch triggers a re-evaluation of integration tests
  - Multiple times a day possible


Integration Testing

- Example of integration:
  - Student + Payment gateway
  - Student + Course + Admin
  - All of the above . . .
- Potential problems:
  - Individual units work - combined system does not
  - Dependencies violations in server - redesign?
- Continuous integration
  - Combined with version control systems: CI
  - Each commit to main branch triggers a re-evaluation of integration tests
  - Multiple times a day possible

And examples of integration might be, you want to test whether the student and payment gateway together work properly, or student+course+admin work properly, so these are like to sort of different levels of integration, but ultimately all of them need to be integrated together to get the complete working system.

Now, what kind of problems could those happen, you might end up in a scenario where the individual units work, the student module by itself works, but the moment I combine it with the course module, something goes wrong, one possible thing might be the student thing was written by developer A, and they assumed certain dependencies in the Python code, the course module was written by developer B, and they assumed a slightly different set of dependencies, you put the two together, some ends up with violations at the server end, it pretty much means that you have to redesign.

And this kind of thing is like a very big problem, it should not happen at all, and the only way it can happen is if the two developers are never talking to each other. So, that cannot happen as a team when you are working on a larger app, you need to make sure that there is somebody at the highest level who is keeping track of all of these things, and that the individual developers are also talking to each other to make sure things are working.

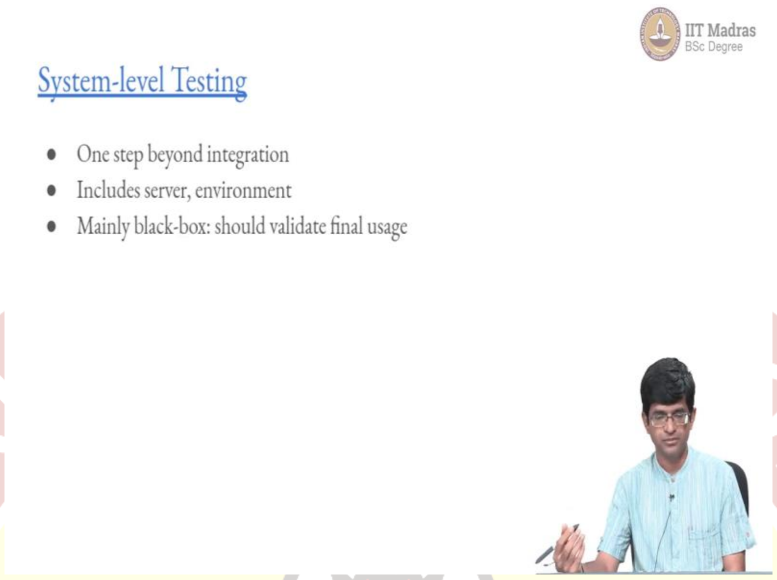
Now, this integration testing is sometimes taken one step further to an idea that is called continuous integration. And in continuous integration, usually this is combined with various version control systems. So, those of you who are using GitHub or GitLab would have seen that there is what they call CI/CD. So the CI is continuous integration, CD is continuous deployment, which is actually also deploying the thing, which is sort of out of scope of what we are doing right now.

But in continuous integration, what we are trying to do is that every time there is a commit to the main branch of code, it triggers a revaluation of all the tests, and these tests are usually integration tests where you are trying to sort of combine different modules together and run tests on the entire setup. And because of the way it works, and because there are multiple software developers working on a system at the same time, it is quite possible to have a scenario where this happens multiple times in a day.

Ideally, you do not want like too many of these because if you have too many integration tests being run, integration tests themselves take should usually take a long time because they are testing integration of multiple modules and if I do this like every two minutes or three minutes, every time I write some code and want to save a file, it triggers an integration test that is a bad idea.

But on the other hand, let us say that I have worked for an hour or two, and I have got a decent amount of code that I feel is working properly, I push it to the main GitHub or Gitlab server, and it says okay now run a set of tests on this, that is entirely possible, and it could happen multiple times a day, and is actually in certain cases, at least considered good practice. What it says is, this is one of the so called more agile forms of software development, where it says that you are doing the tests often, but you are also doing them in a way that you can catch problems quickly.

(Refer Slide Time: 20:53)



System-level Testing

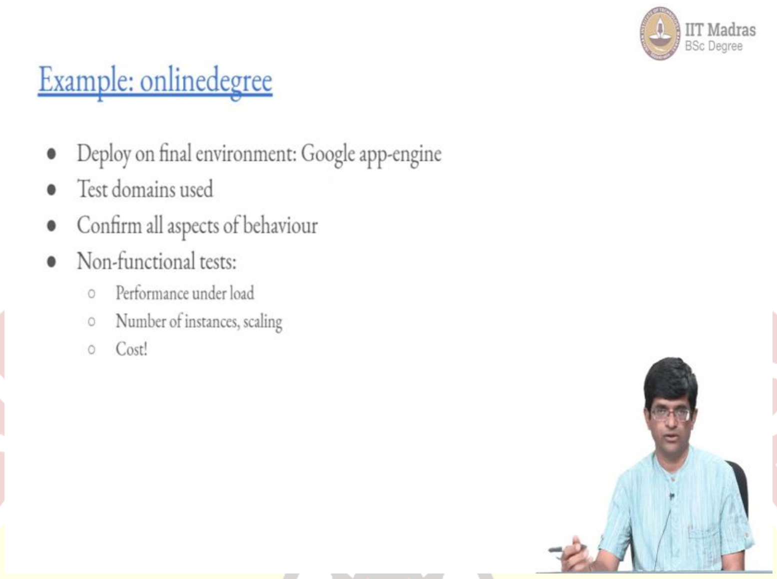
- One step beyond integration
- Includes server, environment
- Mainly black-box: should validate final usage

IIT Madras
BSc Degree

Now, after integration testing, comes system level testing, it is one step beyond integration. Now, over here you might wonder what is the difference, I mean even in integration testing after all I am putting multiple things together. System level usually refers to also including things like what kind of server components you are using, where are you going to deploy it, is it on Google app engine or is it on your own server, or is it one small individual virtual machine on digital ocean.

There are many different ways that could be done over here, it includes the server, includes the environment setup, for the most part system level testing should be done for validating final usage, which means that you are mostly looking at black box testing, you want to be able to see whether the final user interface is working out the way that you want it to be. So, it is related in some ways to integration testing, but it is not something that you would directly do just when you commit code. System level testing actually requires deploying it to a server.

(Refer Slide Time: 22:00)



IIT Madras
BSc Degree

Example: onlinedegree

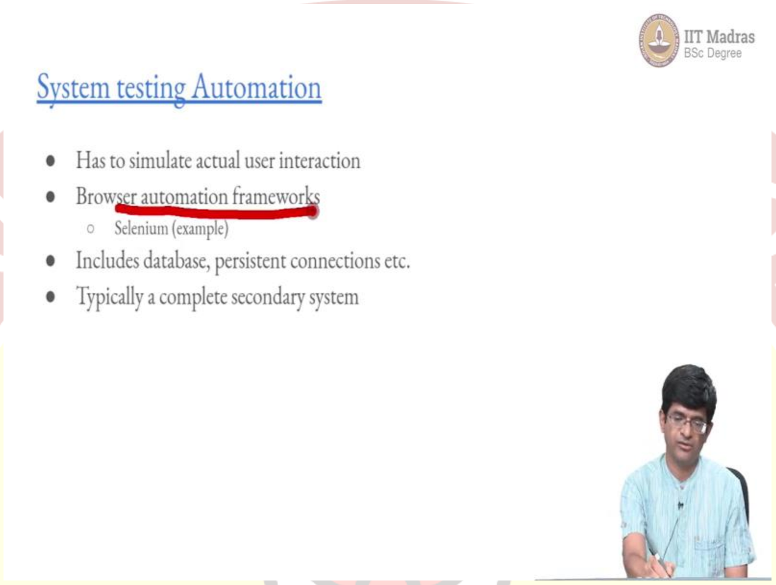
- Deploy on final environment: Google app-engine
- Test domains used
- Confirm all aspects of behaviour
- Non-functional tests:
  - Performance under load
  - Number of instances, scaling
  - Cost!

So, for example in the online degree environment, the final environment where it is deployed is the Google app engine, we usually use some kind of a test domain in order to deploy it, and many kinds of some kinds of tests are run on it, some kinds of aspects of behavior are run over there, some of the tests are automated, some of them have to be done manually, because as you can imagine, system level testing is not just a case of writing a bunch of automatic tests, there might be scenarios where you actually need a person to click through and check whether something works properly.

Now, apart from all of this, at the system level, there can be non-functional tests. And this is not just about whether the logo shows up correctly or not, those are non-functional requirements, but a non-functional test is actually about something else. A non-functional test for something like the online degree app might be, what happens as the number of users increases? What happens to the response time? Does it start sort of failing badly? Does it become very slow? How does it scale? Because after all we are writing these apps in such a way that even as the number of users increases, hopefully you know more server instances should come up that can take up the load.

Are the is the scaling happening in a way that means the user experience is still quite smooth, but at the same time what happens to cost? Does it sort of blow the cost through the roof? Does it start creating too many instances and make it expensive? All of those are non-functional, what I mean by that is, in the worst case even if something slows down, you could argue that yes it is still behaving correctly, it might be almost unusable for the end user because it is slow, but at least it is not doing anything wrong. So, functionality is correct, but non-functional tests in this way can still be a very important part of the overall working off the system.

(Refer Slide Time: 23:56)


IIT Madras
BSc Degree

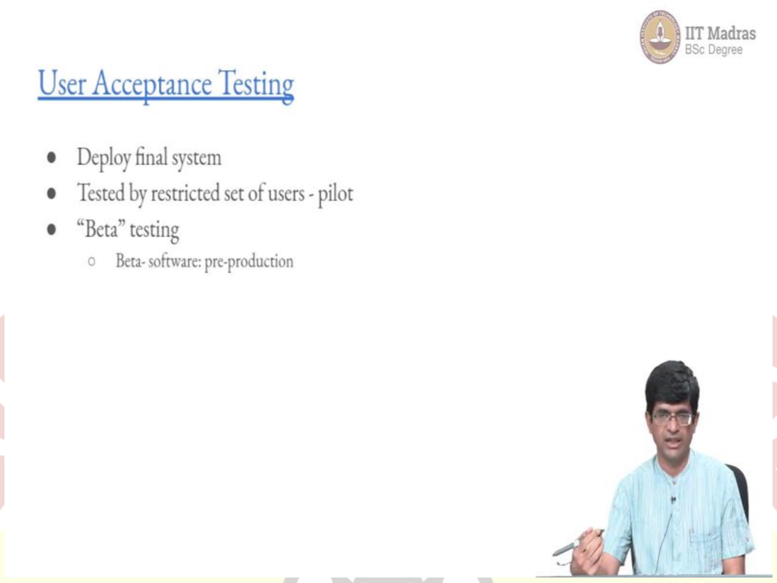
System testing Automation

- Has to simulate actual user interaction
- Browser automation frameworks
  - Selenium (example)
- Includes database, persistent connections etc.
- Typically a complete secondary system

So, system testing, automation is harder in some ways than any of the other kinds of tests that we had, partly because you are actually deploying it, which means that you need to be testing it the way that a person uses it. And very often, we use so called browser automation frameworks over here. A browser automation framework is something which actually allows you to script a browser, it actually allows you to create something like a Chrome browser and automatically go and click on different parts of it, of the page that is displayed.

Usually, system testing would require setting up the complete system which means that you need to have a proper database, persistent connections and so on, which means that very often it will be a secondary system, which is running in parallel with the main deployed system and which is used for testing, and once everything works properly on the secondary system, you can then shift it over into production and say, the system is now ready to go.

(Refer Slide Time: 25:00)



User Acceptance Testing

- Deploy final system
- Tested by restricted set of users - pilot
- "Beta" testing
  - Beta- software: pre-production

And finally, the last form of testing is what is called user acceptance testing, where you deploy the final system, and before actually getting it used in the field, there would be typically a restricted set of users sometimes called the pilot test cases, who might be doing something called beta testing. So, beta version of software is what is usually called a preproduction version, meaning that, it is not yet ready for the public to use, but it can be used by people, especially people who are willing to sort of put up with some problems with it once in a while.

And the whole idea of a beta test is that a person who is participating in a beta test is using the software expecting it to work properly, but if there is a problem, they would at least be able to give you intelligent feedback and not just say look the thing did not work, they should be able to tell you look, when I tried this, under these circumstances, this broke, that is very useful for a developer, because unlike something which just sort of says it gave me an error, they are able to give you sort of pointed feedback saying this is what seems to have gone wrong, can you go and fix it. So, once that is also there, you have sort of user acceptance, and at that stage you can then sort of release it to production. And that is usually the final stage of a software deployment.