Model Curriculum for Minor Degree Course in Advanced Web Development

2022

ALL INDIA COUNCIL FOR TECHNICAL EDUCATION

Nelson Mandela Marg, Vasant Kunj, New Delhi 110070

www.aicte-india.org

Model Curriculum for

Minor Degree Course

in

Advanced Web Development

ALL INDIA COUNCIL FOR TECHNICAL EDUCATION

NELSON MANDELA MARG, Vasant Kunj, New Delhi – 110070

www.aicte-india.org

MESSAGE

With a view to enhance the employability skills and impart deep knowledge in emerging areas which are usually not being covered in Undergraduate Degree credit framework, AICTE has come up with the concept of '**Minor Degree**' in emerging areas. The concept of Minor Degree is discussed in the Approval Process Handbook (APH) for the academic session 2020-21 issued by AICTE. Minor Degree will carry 18 to 20 credits in addition to the credits essential for obtaining the Undergraduate Degree in Major Discipline (i.e. 160 credits usually).

Keeping in mind the need for ~10Crore Web Development Engineers globally, AICTE with the help of industry-academia experts, has framed the curriculum for Minor Degree in Advanced Web Development. Courses have been designed after rigorous brainstorming and considering the inputs from experts and shall be continuously updated by LITE Teaching Fellows from Industry.

I gratefully acknowledge the time and efforts of all those who were involved in preparation of this curriculum especially, the contributions of the members of the Working Group consisting of Engineers Sanjay Vijayakumar, Hari Gopal, Suma Sundararajan Dr. Reena Singh Senior from Pupilfirst and Engineers Avishek Jana, Melson Zacharias and Vignesh Rajendran who are practising software engineering professionals.

I congratulate Vice Chancellors of JNTU(H), JNTU(K), Osmania University, Rashtrakant Tukadoji Maharaj Nagpur University, Savitribai Phule Pune University, Sankalchand Patel Univesity, Marwadi University & University of Engineering and Management Jaipur who have already approved the model curriculum for their students.

All Vice Chancellors and Heads of Institutions may also allocate time in the academic calendar for learners under their institutions to take this opportunity.

The well timed initiative to have this model curriculum addressing the need by Prof. M.P Poonia, Vice Chairman, Prof. Rajive Kumar, Member Secretary, AICTE is highly appreciated. I also appreciate the continuous effort put in coordinating the complete process of development of this curriculum by members of the Policy and Academic Planning Bureau of AICTE namely, Dr. Ramesh Unnikrishanan, Adviser–II; Dr. Pradeep Bhaskar, Assistant Director, Mr. Rakesh Kumar Pandit, Young Professionals and others.

(Prof. Anil D. Sahasrabudhe)
Chairman
All India Council for Technical Education

PREFACE

Dear Faculty,

This handbook is an outline of the approach to implementing the 20 credits Minor in Advanced Web Development Curriculum that is approved by All India Council for Technical Education (AICTE) under the National Educational Alliance for Technology (NEAT) programme of the Ministry of Education, Government of India.

This document is divided into three chapters:

Chapter 1. Key Approaches from NEP 2020 that are covered in Leadership in Teaching Excellence (LITE)

Chapter 2. Role of an Educator in Learner Centered Pedagogy

Chapter 3. Minor in Web Development Course Outline

Select faculty shall be professionally trained by industry coaches in technology tools, curriculum and continuous assessments methods that shall empower them with necessary skills and knowledge required to implement the program through a learner-centered-pedagogy.

Inputs from trained faculty would be taken to create the teaching-learning process and measured output from each batch such that a process of excellence is created at the institutes covering steps of faculty selection and training, student awareness, student selection, diagnostic assessments, formative assessments, summative assessments, identifying teaching assistants, guiding course graduation (Internship/Entrepreneurship) and feedback for improvements of the above steps.

This document shall also serve as a base document for discussion between key stakeholders on how to work together to achieve the goals laid out in the National Educational Policy 2020 in a holistic and planned manner.

Working Group for this Model Curriculum of Minor Degree for UG Degree Courses in Engineering & Technology

S.No	Name	Designation & Organization
1	Avishek Jana	Co-Founder & Principal Engineer, GEOGO Techsolutions Pvt. Ltd.
2	Melson J Zacharias	CTO, Perleybrook Labs LLC.
3	Vignesh Rajendran	Principal Engineer, Oracle
4	Sanjay Vijayakumar	Chairman & Cofounder, Pupilfirst
5	Hari Gopal	CTO, Pupilfirst
6	Suma Sundararajan	COO, Pupilfirst
7	Dr. Reena Singh	Senior Manager - Course Operations, Pupilfirst

Table of Contents

Course Coding Nomenclature:

- WD denotes that minor degree in "Advanced Web Development".
- 101, 201, 301, 401 are courses in order they have to be taken.
- The learner may choose to take multiple courses in the same semester.

Chapter #1 - Key Approaches from NEP 2020 that are covered in Leadership in Teaching Excellence (LITE)

This chapter covers an overview of 10 salient goals of NEP 2020 that are being covered through the LITE[1] programme.

S.No	NEP 2020 Goal	Implementation in LITE
1	Training teachers in learner-centred pedagogy and using online teaching platforms and tools **(NEP 24.g)**	Students in this programme would be learning in a learner-centered model. Faculty would receive continuous support from professional industry experts to learn and deploy learner centered pedagogy
2	For achievement of learning outcomes, classroom transactions shall shift towards competency-based learning **(NEP 4.6)**	With real world competencies mapped into the curriculum, faculty would be able to work with industry experts and upskill their knowledge to be in sync with latest industry standards and techniques.
3	Training teachers in latest pedagogies for formative and adaptive assessments and implementing pedagogical plans based on competency based education. **(NEP 5.15)**	Faculty shall be equipped to perform diagnostic, formative and summative assessments using technology tools. This shall enable faculty to see the overall course progression of the class and spend more time on weaker students such that nearly all students in the class are able to achieve the course learning outcomes.
4	University Admissions for students with singular interest **(NEP 4.45)**	AICTE shall select CBSE schools into the LITE program. University faculty shall be connected with school teachers to identify gifted students who can have a direct admission to university programmes.
5	Continuous Professional Development of Faculty **(NEP 5.15)**	From the regular 5 day FDP programmes, AICTE, LITE has a continuous model of faculty development with professional support from industry all-round the year.
6	Focus on greater industry partnerships and innovation amongst student communities **(NEP 11.2)**	With deep industry integration through NEAT, faculty shall be able to guide students towards applying knowledge to practical use cases like an industry setting.

[1] https://www.aicte-india.org/sites/default/files/fdc/AICTE-LITE%20Programme.pdf

7	Creating Optimal Learning Environments for high quality learning outcomes **(NEP 12.1)**	Faculty shall receive support to set up learning environments right from creating awareness, student selection, diagnostic assessments, formative assessments, summative assessments, identifying teaching assistants, guiding course graduation(Internship/Entrepreneurship) and feedback for improvements of the above steps.
8	Classrooms shall have access to the latest educational technology that enables better learning outcomes. **(NEP 13.2)**	AICTE has selected the open-source tooling built by pupilfirst.org after careful multi-step selection process and demonstrating proven learning outcomes for students. Faculty shall have the ability to learn about the tooling in depth and enhance student experience through more inputs.
9	Research in Educational Technology for improving teaching-learning-evaluation process and increasing access to education **(NEP 23.1)**	Institutions and Faculty who are part of LITE after training shall form part of a Industry-Academia Research Group to identify new areas of research and themes to be submitted to AICTE for support.
10	Creating pilot studies for digital education, training teachers to be effective online educators and creating tools for blended models of learning **(NEP 24.1, 24.3, 24.4)**	AICTE has designed the LITE programme to start with 50 institutions and faculty who shall demonstrate a new model of teaching-learning to the other 10,000 affiliated institutions.

Chapter #2 - Role of an Educator in Learner Centered Pedagogy

In the 21st century knowledge economy, the role of a faculty in a technology-enabled teaching learning environment has slowly transformed from theory lecturers to that of an active educator and facilitator for students to learn how to learn on their own. Faculty shall help shape and smoothen the learning curve of students.

This chapter will talk about key responsibilities of educators and suggest the pedagogical approach that can be followed for better learning outcomes for students.

Role of an Educator

From teaching to facilitating (experts take care of dynamic curriculums, teachers faciliate in the learning process)

1. Preparing the mindset of students
2. Preparatory groundwork according to needs and current understanding level of students
3. Facilitating throughout the curriculum
4. Professional self-development
5. Mentoring and Guidance

Teaching Pedagogy

1. Discovery, discussion and analysis based learning (Taking inspiration, thinking alone and thinking together)
2. Peer assisted learning to enable critical thinking
3. Classroom as place to play, experiment, and reflect (Creating and Sharing)

Open Source tools for the Educator

The Web Development courses are hosted online in the open-source teaching-learning platform, Pupilfirst LMS which is developed keeping all learners of the classroom in mind, and by taking the feedback from teachers and students from universities into consideration. The tool supports quality teaching and learning via many inbuilt processes. A few of them are listed below -

- **Community** - This is a place where Students, Teachers, Coaches, Teaching Assistants, Student-graduates (from previous batches) come together to build a developer ecosystem that supports peer-learning. This acts as a repository of questions /discussions from previous batches of students, that help a new student clarify their doubts with ease.

- **Dynamic Content** - The course content on the platform is written by Industry experts gets revised very frequently based on feedback from students on concepts/topics and updates in technology so that it is always of high relevance and quality.
- **Review Checklist** - Giving an elaborate, qualitative feedback to students is very important in their learning. The platform supports creation of custom quality templates of review feedback by the Coach/TAs, based on the quality and features implemented by a student in the submission, that can be re-used by the educator to give high-quality template feedback to the students.

Chapter #3 - Minor in Advanced Web Development Course Outline

In this chapter, we shall discuss the details of the web development courses and skills gained by the student by the time they complete the course. The course curriculum is set in a manner to engage and drive the students to develop competencies required by the modern web development industry so that they are ready to contribute to a codebase running in a production environment when they graduate from the course.

Definition of Credit

1 Hour of Lecture (L) / Week	**1 Credit**
1 Hour of Tutorial (T)/ Week	**1 Credit**
2 Hours of Practical (P) /Week	**1 Credit**

The Minor degree programme is divided into four courses:

1. Web Development 101 - Getting started with JavaScript
2. Web Development 201 - Server-side programming with Node.js
3. Web Development 301 - Front-end development with React & TypeScript
4. Web Development 401 - Getting ready for production

These four courses add up to 20 Credits. Students may choose to replace WD 401 with an internship as well.

Minor Degree in Advanced Web Development

Course Structure						
S.No	**Course Code**	**Title**	**L**	**T**	**P**	**Credits**
1	WD101	**Web Development 101** Getting started with JavaScript	0	1	0	1
2	WD201	**Web Development 201** Server-side Programming with Node.js	0	6	0	6

3	WD301	**Web Development 301** Front-end development with React & TypeScript	0	6	0	6
4	WD401	**Web Development 401** Getting ready for production	0	0	10-14	5-7
TOTAL			**0**	**13**	**10-14**	**18-20**

Head-start to industry for students

A student of any branch, who has completed second semester, is eligible for this advanced minor degree programme. The institutes, their affiliating university may plan the implementation details along with Pupilfirst.

The course work may be spread over 2-3 semesters till the degree is completed.

Colleges and their respective affiliating universities may plan the implementation details together with AICTE LITE faculty coordinators.

AICTE Leadership in Teaching Excellence (LITE) - Minor Degree in Advanced Web Development

Detailed Syllabus

Course Code	:	Web Development 101
Course Title	:	Getting Started with JavaScript
Number of Credits	:	1 (L: 0; T: 1; P: 0)
Course Category	:	Web Development (WD)

Course Objective:

This course is meant for students who do not have prior programming experience, or have a light background, and are looking to build a robust foundation for computational thinking.

They'll learn to deconstruct what software applications do, and reason about the essence of computation as transformation of data from one shape to another.

Practically, they will be able to set up a development environment, be introduced to HTML & CSS, and learn to program in a functional subset of JavaScript.

By the end of the course, they will build an Online Registration form that runs on the browser, with the ability to store and retrieve submissions using browser native web storage. They will also be able to create and deploy a simple and basic website to the internet.

Prerequisites:

- Students should have access to a computer with a modern OS (Windows 10 or above, Ubuntu 20.04 and above, macOS 10.15 and above).
- Students should have computer literacy. It includes skills like the ability to browse the internet and find information, ability to use software applications like word processors and spreadsheets, and be comfortable with user-level operating system concepts like CPU, memory, disks, and files and folders.

Course Contents:

Module 1: Welcome to the course
This module introduces students to the World Wide Web. Students are also guided through setting up a development environment on their computer. Students are taught to set up Visual Studio Code as their editor and to use Prettier and ESLint extensions for code formatting and code quality respectively.

Module 2: Let's create our own websites!
In this module students learn how to develop a simple website using HTML. They experiment with some useful HTML tags, learn how to look inside websites. The students deploy the website they develop and share it over the Internet.

Module 3: Basic Introduction to HTML and CSS
This module gives some basic introduction of HTML and CSS. Students learn how to put together a web page that contains HTML, CSS, and JavaScript.

Module 4: Style Matters
This module teaches students how to style web pages using CSS. Students also learn how to use Tailwind CSS to add custom styling to their webpages.

Module 5: Working with JavaScript data types
In this module students are introduced to different data types - Number, Boolean and String. They carry out various operations on these data types to understand the difference between them and also can decide the suitability of a data type given a task or operation.

Module 6 - Working with JavaScript data structures
This module teaches students how to iterate with arrays using *forEach* method and generate an HTML list from an array. Students perform various transformations on an array using the map method and are introduced to filtering of arrays.

Students are also introduced to objects in JavaScript. They learn how to create objects, add and access properties of objects and perform various operations on them.

Module 7 - Functions - code we can call multiple times
This module teaches students how to use functions to modularize the codebase. Students learn how to return values from a function and also how to treat functions as values, by passing them as arguments.

Module 8 - Create a form with validations
In this module, students learn about HTML form element and form data. They learn how to create a user form, add validations, store and retrieve data.

Students develop and deploy their personal website that includes the form they have built with additional validations and display the data submitted by users on the website.

Text/Reference Books (if any):

This course does not require students to use physical textbooks. Instead, original course material (videos, text and images) has been prepared for students to go through and is open-sourced under Creative Commons Attribution-ShareAlike 4.0 International License © Freshworks Inc. & Pupilfirst Pvt. Ltd.

This course material may include some third-party content with a compatible license, and external links for additional reading on the Internet. Students are also taught how to search for information on their own.

Course Outcomes:

By completing the WD 101 course, students will gain a foundation in programming and computational thinking, and be introduced to the field of web development.

Specifically, they will learn how to:

- Set up a development environment.
- Create and style basic web pages.
- Transform data with JavaScript.
- Use computational abstractions.
- Work with the HTML Forms.
- Work on native HTML Form Validations.
- Understand Web Storage for saving and retrieving data.

****

Course Code	:	Web Development 201
Course Title	:	Server-side programming with Node.js
Number of Credits	:	6 (L: 0; T: 6; P: 0)
Course Category	:	Web Development (WD)

Course Objective:

The objective of this course is to teach students how to build web applications using the Express.js framework, with focus on industry-practices like functional programming, object-oriented design, programming style guides, security, and version control.

Prerequisites:

Students should have completed *Web Development 101,* before beginning this course. Students should have access to a computer with a modern OS (Windows 10 or above, Ubuntu 20.04 and above, macOS 10.15 and above).

Overview:

Through the course, the student will work up to build a To-do Management application using Express.js, PostgreSQL, HTML, and CSS. The app will be hosted on the cloud using Heroku.

They will then independently work on a capstone project which will be a microcosm of a production web application and the challenges and trade-offs that come with it.

Being an industry-led course, the students will also be exposed to professional practices like code reviews, code quality, and version control (git). They'll have access to a Web Development Community where they are encouraged to ask well-crafted and specific questions, a valuable skill in a professional setting.

Course Content:

Module 1 - Introduction to Node.js
In this module students are introduced to Node.js - they learn how to install it and write programs on it and use Node.js REPL. Students also start using GitHub and learn how to collaborate on code with others using the git tool.

Module 2 - Working with NPM
This module is an introduction to Node.js package manager for students where they start writing custom NPM modules. They also explore and use built-in modules of Node.js

Module 3 - Node.js deep dive
In this module students start building their first application and learn how to use closure to emulate private methods.

Module 4 - Testing

In this module students are introduced to testing. They start writing tests for their application, learn how to use Jest to run the tests and pre-commit hooks to run the tests automatically before each commit.

Module 5 - Databases and Sequelize
In this module students get to learn about databases and set up a PostgreSQL database. They learn how to connect to a database from a Node.js application and then work on the database by creating Sequelize models to manipulate data.

Module 6 - Backend Web development with Express.js
In this module, students develop their first application and connect it to the PostgreSQL database on their machine, and begin learning the basics of the CRUD pattern by building some additional features to the application that they're working on.

Module 7 - Add User Interface for To-do Application
This module teaches students how to create interfaces for their application. They also practice converting a given visual design into working HTML and CSS.

Module 8 - EJS Templating
This module teaches touches upon the basics of the MVC pattern, instructing student how to render dynamic data inside their HTML pages using EJS templates. This module also lets the student practice how to deploy their work to a remote server.

Module 9 - HTML forms to save and accept user inputs
This module teaches students how to accept user input on their application via form element in HTML. Students also explore more of the CRUD pattern, moving onto creation of resources using forms, deletion of existing resources, and learn about Cross Site Request Forgery (CSRF) and how authenticity tokens can be used to prevent such attacks. Students are also introduced to APIs.

Module 10 - User Authentication and final wrap-up
In this module students dig deeper into Sequelize association, migration and validation. They build a functional user sign-up page, learn about password storage and play around with browser cookies, sessions, user authentication, and related best practices. They also learn to display one-off flash messages.

Text/Reference Books (if any):

This course does not require students to use physical textbooks. Instead, original course material (videos, text and images) has been prepared for students to go through and is open-sourced under Creative Commons Attribution-ShareAlike 4.0 International License © Freshworks Inc. & Pupilfirst Pvt. Ltd.

This course material may include some third-party content with a compatible license, and external links for additional reading on the Internet. Students are also taught how to search for information on their own.

Course Outcomes:

By the end of the course the students will be able to:

- Build web applications using Express.js.
- Manipulate data using both imperative and functional programming techniques.
- Model real-world systems using object-oriented design
- Write HTML & CSS to create elegant web pages
- Build database applications using Sequelize.

The students would have built fundamental first-principles based knowledge about web development and the practical chops to use them to build real-world software. They would also have learnt what it is to work in a professional software environment, helping build a strong foundation for their transition to the industry as competent professionals.

****

Course Code	:	Web Development 301
Course Title	:	Front-end development with React & TypeScript
Number of Credits	:	6 (L: 0; T: 6; P: 0)
Course Category	:	Web Development (WD)

Course objective

The course aims at training students on the following fronts:

- Understand the basic architecture of front end applications and create web applications using React TypeScript front-end stack.

- Interaction between a client-side application and server-side app via an API.
- Industry practices for state management and usage of static types.
- Best practices with regard to the development of a modern client-side application.
- Learn to build TypeScript projects from scratch to scale.

Prerequisites

Students should have completed *Web Development 201,* before beginning this course. Students should have access to a computer with a modern OS (Windows 10 or above, Ubuntu 20.04 and above, macOS 10.15 and above).

Course outline

Module 1: React fundamentals

This module introduces students to development using TypeScript by setting up a development environment, introducing them to the TypeScript programming language and the React framework, and demonstrates some of the basic concepts that underpin the use of React for building dynamic reactive user interfaces.

Module 2: State management

This module introduces students to the *Hooks* feature of React, on the usage of callback functions and how to use them to build dynamic components that maintain an internal state. This module also demonstrates state management by building a form and accepting user input.

Module 3: A deeper dive into React Hooks

This module discusses the common pitfalls of state management, introduces in-browser persistent storage, demonstrates additional standard hooks and the creation and use of custom hooks.

Module 4: Client-side routing

This module covers the concept of client-side routing as a separate behaviour from server-side route management. It demonstrates the various aspects of client-side routing such as the use of path parameters, query parameters, programmatic navigation and the operation of links and URLs that are handled client-side.

Module 5: Types in depth and Variants

This module takes a deeper dive into TypeScript's type system, demonstrating concepts such as function types, custom-defined types, generics, and union types. It also instructs the student why the "any" type should be avoided in practice, and finishes up with a demonstration of TypeScript's type inference behaviour.

Module 6: Modelling and managing complex states

This module teaches students how to manage complex states using the state reducer pattern, and then demonstrates the pattern by implementing it using React's useReducer hook.

Module 7: APIs and state modelling

Through this module, students are introduced to using APIs to interface their client-side code with the server-side, how to model types to allow this interaction to take place, how to maintain a session with the backend, and how to work with pageable APIs.

Module 8: Best practices and npm packages

This module covers the best practices of front-end development, including the importance of accessibility and WAI-ARIA standards, and use of third-party packages from the NodeJS ecosystem.

Module 9: Production React Apps

This final module focuses on production-specific optimizations of a React application, best practices for its build & deployment process, and the configuration of a progressive web app.

Text/Reference Books (if any):

This course does not require students to use physical textbooks. Instead, original course material (videos, text and images) has been prepared for students to go through and is open-sourced under Creative Commons Attribution-ShareAlike 4.0 International License.

This course material may include some third-party content with a compatible license, and external links for additional reading on the Internet. Students are also taught how to search for information on their own.

Course outcome

By the end of the course the students will:

- Be able to create Single Page Web Applications (SPA) using React, Typescript and TailwindCSS.
- Have a solid understanding of static types, and know how to port untyped JavaScript to TypeScript.
- Learn typed state management that is inline with a backend data model.

****

Course Code	:	Web Development 401
Course Title	:	Getting ready for production
Number of Credits	:	7 (L: 0; T: 0; P: 14)
Course Category	:	Web Development (WD)

Course objective

The objective of WD401 is to allow the student to learn more about production-ready deployments.

This can be achieved either via the WD401 course material or through an internship at a company. To complete WD401 by following Pupilfirst's course material, students will need to deploy an application of their choice that integrates learnings from earlier courses, and also work through the material of the course, integrating the new production-readiness concepts that are presented here.

The second option is for the student to gain an internship at a company where they get to work on an application that tests a similar skill-set.

Prerequisites

Students should have completed *Web Development 301,* before beginning this course. Students should have access to a computer with a modern OS (Windows 10 or above, Ubuntu 20.04 and above, macOS 10.15 and above).

Course outline

Module 1: Workflow using pull-requests

This module acts as an advanced guide to the usage of git in development teams, where the norm is to develop on branches, perform peer-reviews, and to re-work based on reviews before merging. Since this cycle is most often performed using online tooling that uses pull requests to achieve this workflow, students are taught how to open a pull request, make changes, submit work for review and then update code based on review.

Module 2: JS Bundling - integration of JS into non-JS backends

This module covers the history of why "bundling" as a process exists for the JS ecosystem, the most common bundling tools, and the general methodology. This module also covers the new "import maps" feature that allows for similar capability without the use of a bundling tool.

Module 3: Compile to JS languages - options & approaches

This module covers the reason why languages that compile to JS exist, the different purposes that they serve, and demonstrate a few of the most popular options and the differences between each.

Module 4: Testing

This module covers the importance of testing, the different approaches to testing such as unit testing, integration testing, and hybrid testing. It should also cover popular libraries that are used to help with testing, and also common pitfalls in the practice of testing and how to avoid them.

Module 5: CI/CD - Continuous integration & delivery

This module teaches students about modern development processes that enable teams to release changes quickly and often, by leading them through the process of setting up an automated system that detects changes to code to run tests and then linking that to the deployment of code that passes its test suite to a remote server.

Module 6: Application environments

This module teaches students about the different environments in which an application is expected to run. This module explains the differences between the environments that a student has already operated in - development, testing & production, and also introduces the concept of a staging environment which acts as a gateway to the production environment.

Module 7: Containerization

This module covers the field of containerization - where complex applications are packaged to run in isolated spaces called containers. The approach for covering this topic involves the use of the popular Docker (OCI) standard, teaching students how to build a Docker image for their web application, and how to deploy this image to different targets.

Module 8: Internationalisation and localisation

This module covers i18n, teaching students the basics of setting up their web applications to support users who prefer or require a language different from the default language of the app, and/or live in a timezone that is different from the default. This module also covers L10n, teaching students how to use the i18n framework to customise their web application for another locale.

Module 9: Error logging & debugging

This module covers the practice of logging and notification of runtime errors that occur on a deployed application. This module also covers the process that is followed to detect the source of a bug, and how testing can be used to ensure a fix and to prevent recurrences.

Text/Reference Books (if any):

This course does not require students to use physical textbooks. Instead, original course material (videos, text and images) has been prepared for students to go through and is open-sourced under Creative Commons Attribution-ShareAlike 4.0 International License.

This course material may include some third-party content with a compatible license, and external links for additional reading on the Internet. Students are also taught how to search for information on their own.

Course outcome

By the end of the course the students will:

- Be able to bundle a codebase with non-trivial JS dependencies and code.
- Know how to differentiate between popular JS flavours and pick one that is suitable for a task.
- Understand why testing is important, what TDD is, and be able to write both unit and integration tests for Rails applications that use JS in the front-end.
- Be able to set up a CI/CD pipeline for a server-side application, ensuring the code reaches production automatically after tests pass.

- Know how to organise & communicate development work using pull requests.
- Be aware of container-based deployments, be able to build a Docker image for their web application and then deploy that image to a web server.
- Know how to set up a web application to support localization.
- Set up error-logging for their web application to capture runtime errors - both in the back-end and in the front-end. They'll also know how to write tests that replicate errors before implementing a fix to prevent regressions.

ALL INDIA COUNCIL FOR TECHNICAL EDUCATION
Nelson Mandela Marg, Vasant Kunj, New Delhi 110070
www.aicte-india.org