

**KANDULA SRINIVASA REDDY MEMORIAL COLLEGE OF ENGINEERING
(AUTONOMOUS)**

KADAPA- 516003. AP

(Approved by AICTE, Affiliated to JNTUA, Ananthapuramu, Accredited by NAAC)

(An ISO 9001-2008 Certified Institution)

DEPARTMENT OF ARTIFICIAL INTELLIGENCE & MACHINE LEARNING



VALUE ADDED COURSE

ON

“AngularJS & MongoDB”

Resource Person : Mr. Sunil J, Assistant Professor, Dept.of AIML,KSRMCE

Course Coordinator: Dr. K. Srinivasa Rao, Professor,Dept.of AIML,KSRMCE

Duration: 28/08/2023 to 23/09/2023

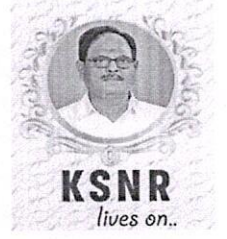


K.S.R.M. COLLEGE OF ENGINEERING (UGC-AUTONOMOUS)

Kadapa, Andhra Pradesh, India- 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution



Lr./KSRMCE/AIML/2023-24/

Date: 24-08-2023

To
The Principal,
KSRMCE,
Kadapa.

Respected Sir,

Sub: Permission to Conduct Value added Course on “Angular JS & Mongo DB”
28/08/2023 to 23/09/2023–Req- Reg.

The Department of Artificial Intelligence & Machine Learning is planning to offer a Value-Added Course on “Angular JS & Mongo DB” to B. Tech. students. The course will be conducted from 28/08/2023 to 23/09/2023. In this regard, I kindly request you to grant permission to conduct Value Added Course.

Thanking you sir,

Yours faithfully

(Dr. K. Srinivasa Rao, Professor in AIML)

forwarded to
Principal Sir,
KSRMCE
24/8/23

Permitted
U.S.S.M.M.Ty



K.S.R.M. COLLEGE OF ENGINEERING (UGC-AUTONOMOUS)

Kadapa, Andhra Pradesh, India- 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution



Cr./KSRMCE/AIML/2023-24/

Date:25/08/2023

Circular

The Department of Artificial Intelligence & Machine Learning is offering a Value Added Course on “**Angular JS & Mongo DB**” from **28/08/2023 to 23/09/2023** to B.Tech students. In this regard, interested students are requested to register their names for the Value Added Course with Course Coordinator.

For further information contact Course Coordinator.

Course Coordinator: Dr. K. Srinivasa Rao, Professor, Dept. of AIML.-KSRMCE.

Contact No: 8978571543

HOD

Dr. K. Srinivasa Rao, M.Tech., Ph.D.
Professor & HOD AIML
K.S.R.M. College of Engineering
(Autonomous)
KADAPA- 516 005. (A.P.)

Cc to:

IQAC-KSRMCE



K.S.R.M. COLLEGE OF ENGINEERING (UGC-AUTONOMOUS)

Kadapa, Andhra Pradesh, India- 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution



Date: 27-09-2023

DEPARTMENT OF ARTIFICIAL INTELLIGENCE & MACHINE LEARNING

REGISTRATION FORM

Value Added Course

On

“AngularJS & MongoDB” From 28/08/2023 to 23/09/2023

S. No	Roll Number	Full Name	Branch	Semester	Signature
1	219YIA03908	D. Vishnu Teja	AIML	V	Vishnu
2	219YIA3901	A. Mohammed	AIML	V	Am
3	219YIA3902	P. Pradeep	AI&ML	V	Pradeep
4	219YIA3931	Naciella Argun	AI&ML	V	Argun
5	219YIA3956	S. Sreenath	AI&ML	V	Sreenath
6	219YIA3910	D. Vishnu Teja	AI&ML	V	D. Vishnu
7	219YIA3904	A. Anith	AI&ML	V	Anith
8	219YIA3949	S. Mohammed Abdul Majeed	AI&ML	V	S. Mohammed
9	229YIA3903	K. Dhanush	AI&ML	V	Dhanush
10	219YIA3929	Moghal Junaid	AI&ML	V	Junaid
11	219YIA3916	G. Balaji Reddy	AI&ML	V	G. Balaji
12	219YIA3905	B. V. Ravi Kumar	AI&ML	V	B. V. Ravi
13	219YIA3966	Pragathi Gajula	AI&ML	V	Pragathi
14	219YIA3950	S. Mohammed Faheemullah	AI&ML	V	S. Mohammed
15	219YIA3921	K. UMESH	AI&ML	V	K. Umesh
16	219YIA3907	B. Sri Tejaswini	AI&ML	V	B. Sri Tejaswini
17	219YIA3908	C. Asha Deepthi Reddy	AI&ML	V	C. Asha Deepthi Reddy

18	21941A0580	M. Hemalatha	AI&CSE	V	M. Hemal
19	22941A3920	KATIGANDLA SOWMYA	AI&ML	III	Sowmya
20	21941A3920	K. Bageswara	AI&ML	V	K.B
21	21941A3943	P. Pochamma	AI&ML	V	P. Pochamma
22	21941A3954	Srigireddy Reddiah	AI&ML	V	S. Reddiah
23	21941A3922	K. Karthikan	AI&ML	V	K.P.
24	21941A3932	NAKKA UURU ANKARSH	AI&ML	V	UURU
25	21941A0512	V. Manasa	CSE	V	V. Manasa
26	21941A0484	L. Hemant Kumar	ECE	V	Hemant
27	21941A0494	M. Sunil Kumar	ECE	V	Sunil Kumar
28	21941A0496	H. Akhila	ECE	V	Akhila
29	21941A0536	D. Naga Mahindra	CSE	V	M
30	21941A0514	VARIKUTI SREELAKSHMI	CSE	V	V. Sree Lakshmi
31	22941A3926	Lakshmi vaishnavi	AI&ML	III	Lakshmi vaishnavi
32	22941A3933	N. Sridhar Reddy	AI&ML	V	Sridhar
33	22941A3928	MANDURU USAYSAI	AI&ML	III	Sai
34	21941A0483	H. Mallesh	ECE	V	Mallesh
35	21941A0538	D. Nanditha Reddy	CSE	V	M
36	21941A3955	S. M. GAFFAR ALI	AI&ML	V	
37	21941A3955	S. Hemant Kumar	AI&ML	V	
38	21941A0408	B. Poornima	ECE	V	Pul
39	21941A0409	B. Charan	ECE	V	Charan
40	21941A3917	Kakumani Hagika	AI&ML	V	Hagika
41	21941A3919	Kanchannreddy Jahnavi	AI&ML	V	Jahnavi
42	21941A0404	Anagiri Kanya	ECE	V	Kanya
43	21941A0405	APPARANDU CHARITIKA	ECE	V	Charitika
44	21941A3906	B. Sai Tejaswini	AI&ML	V	B. Tejaswini
45	21941A3907	C. Khamaleshwar	AI&ML	V	C. Khamaleshwar
46	21941A0488	M. Veera Brahmendra	ECE	V	Veera

47	21941A0539	D. pradeep	AIML	V	pradeep
48	21941A3830	myrthy meghana	AIMI	V	meghane
49	21941A0489	M Venkatesh	ECE	V	Venkat
50	21941A3957	S. Yezdan Ahamed	AIML	V	Yezdan
51	22941A3922	M Umadevi	AIML	III	Uma
52	21941A0478	K Chaitanya Sai	ECE	V	Chaitanya
53	22941A3926	L Vaishnavi	AIML	III	Vaishnavi
54	22941A3902	D. Nagalakshmi	AIML	V	D. Nagalakshmi
55	22041A3932	M. M. Gagni Baig	AI&ML	III	Gagni
56	21941A0582	M. Bindu Latha	CSE	V	Bindu
57	22941A0077	K. Chamundeswari	ECE	V	Chamundeswari
58	22941A0478	K. Chaitanya Sai	ECE	V	Chaitanya
59	21941A0581	M. SETHA RAJITH	CSE	V	Seetha
60	21941A0513	V. pradeep sreddy	CSE	V	Pradeep
61	21941A0515	V. Rajeswari	CSE	V	Rajeswari
62	22941A3927	M. Umadevi	AI&ML	III	Uma
63	21941A0407	A Bharu prakash	ECE	V	Bharu
64	21941A0495	M Mallikarjuna	ECE	V	Mallikarjuna
65	21941A0492	M Jeyaradhan	ECE	V	Jeyaradhan
66	22041A3931	meghna jaminia	AI&ML	III	Meghna
67	21941A0537	D pradhiraaj	CSE	V	pradhiraaj



Coordinator



HoD

Dr. K. SRINIVASA RAO, M.Tech., Ph.D.
Professor & HOD AIML
K.S.R.M. College of Engineering
(Autonomous)
KADAPA- 516 015. (A.P.)

 /ksrmce.ac.in

Follow Us:



/ksrmceofficial

Syllabus of Value Added Course

Course Name: AngularJS & MongoDB

Course Objectives:

1. To learn about basic features of Angular JS and Mongo DB
2. To create an Angular JS Applications using CSS, JavaScript and Mongo DB
3. To learn about connecting data sources using client server applications.

Course Outcomes:

1. Learners will be able to design web applications using Scripting languages as Mongo DB as a database
2. Develop the console and GUI applications using Java Script
3. You will also understand how to use the Node.js MongoDB driver for the same ends in order to manipulate your data directly from Node.js.

UNIT-I

Introduction to HTML & CSS:

HTML Basics, Elements, Attributes, Styles, Forms, Form Elements, Input Element Types, Input Attributes, File Paths, Script tag, HTML & XHTML, CSS Introduction, Syntax, Selectors, Styling, Pseudo class, Pseudo Elements, CSS Tables, CSS Box Models, CSS Opacity, CSS Navigation Bar, Dropdowns.

UNIT-II

Introduction to JavaScript & Working with Objects

JavaScript Statements, Keywords, Functions, JavaScript Programs, Operators, Functions, Function Types, Data Types, Primitive Type, Object Oriented Programming, Object Creation, Adding Methods of Objects, JavaScript Loops & Iteration, Adding Properties of Objects, JavaScript Conditional Parameters, Function Return Statements

UNIT-III

Angular JS Basics & Angular Expressions

What is Angular JS?, Why Angular JS?, Why MVC matters, MVC-The Angular JS way, Features of Angular JS ,Model-View-Controller, My First Angular JS app, All about Angular Expressions, How to use expressions, Angular vs JavaScript, Built-In Filters, Using Angular JS Filters, Creating Custom Filters

UNIT-IV

Introduction to NoSQL Database & CRUD Operation in MongoDB

What is NoSQL?, Difference between NoSQL and RDBM, Benefits of NoSQL, CRUD (Creating, Reading & Updating Data) Mongo Shell, Query Operators

Data Modeling & Storage Classes

Schema Design Pattern, Case Studies & Tradeoffs, Automatic Storage Class, Static Storage Class, External Storage Class, Register Storage Class

UNIT-V

Aggregation & MongoDB Replication

Aggregation Framework Goals, The Use Of The Pipeline, Comparison With SQL Facilities, Application Engineering Drivers, Impact Of Replication And Sharding On Design And Development.

Text Books/Reference Books:

1. Professional JavaScript for Web developers by Matt Frisbie by Wiley publications.
2. Building Browser extensions by Create modern extensions for chrome, Safari, Firefox and edge by Matt Frisbie by Wiley publications.
3. Angular Cook book: Discover 70 recipes that provide the solutions you need to know to face every challenge in angular 2 by Packt.



K.S.R.M. COLLEGE OF ENGINEERING (UGC-AUTONOMOUS)

Kadapa, Andhra Pradesh, India- 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution



SCHEDULE

Department of Artificial Intelligence & Machine Learning

Value Added Course

On

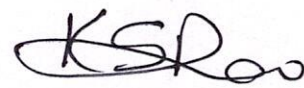
“AngularJS & MongoDB” From 28/08/2023 to 23/09/2023

Date	Timing	Resource Person	Topic to be covered
28/08/2023	4 PM to 6 PM	Mr. Sunil J	HTML Basics, Elements, Attributes, Styles, Forms, Form Elements
29/08/2023	4 PM to 6 PM	Mr. Sunil J.	Input Element Types, Input Attributes, File Paths, Script tag, HTML & XHTML
30/08/2023	4 PM to 6 PM	Mr. Sunil J	CSS Introduction, Syntax, Selectors, Styling, Pseudo class, Pseudo Elements, CSS Tables
31/08/2023	4 PM to 6 PM	Mr. Sunil J	CSS Box Models, CSS Opacity, CSS Navigation Bar, Dropdowns.
01/09/2023	4 PM to 6 PM	Mr. Sunil J	JavaScript Statements, Keywords, Functions, JavaScript Programs, Operators, Functions
02/09/2023	4 PM to 6 PM	Mr. Sunil J	Function Types, Data Types, Primitive Type, Object Oriented Programming
03/09/2023	4 PM to 6 PM	Mr. Sunil J	Object Creation, Adding Methods of Objects ,JavaScript Loops & Iteration
04/09/2023	4 PM to 6 PM	Mr. Sunil J	Adding Properties of Objects, JavaScript Conditional Parameters, Function Return Statements
05/09/2023	4 PM to 6 PM	Mr. Sunil J	What is Angular JS?, Why Angular JS?, Why MVC matters, MVC-The Angular JS way
06/09/2023	4 PM to 6 PM	Mr. Sunil J	Features of Angular JS ,Model-View-Controller, My First Angular JS app, All about Angular Expressions
08/09/2023	4 PM to 6 PM	Mr. Sunil J	, How to use expressions, Angular vs JavaScript, Built-In Filters, Using Angular

			JS Filters, Creating Custom Filters
09/09/2023	4 PM to 6 PM	Mr. Sunil J	What in NoSQL?, Difference between NoSQL and RDBM
11/09/2023	4 PM to 6 PM	Mr. Sunil J	Benefits of NoSQL, CRUD (Creating, Reading & Updating Data) Mongo Shell, Query Operators
12/09/2023	4 PM to 6 PM	Mr. Sunil J	Schema Design Pattern, Case Studies & Tradeoffs
13/09/2023	4 PM to 6 PM	Mr. Sunil J	Automatic Storage Class, Static Storage Class
14/09/2023	4 PM to 6 PM	Mr. Sunil J	External Storage Class, Register Storage Class
15/09/2023	4 PM to 6 PM	Mr. Sunil J	Aggregation Framework Goals, The Use Of The Pipeline
16/09/2023	4 PM to 6 PM	Mr. Sunil J	Comparison With SQL Facilities, Application Engineering Drivers
19/09/2023	4 PM to 6 PM	Mr. Sunil J	Impact Of Replication And Shading On Design And Development.
20/09/2023	4 PM to 6 PM	Mr. Sunil J	Sample Projects Implementation
21/09/2023	4 PM to 6 PM	Mr. Sunil J	Sample Projects Implementation
22/09/2023	4 PM to 6 PM	Mr. Sunil J	Sample Projects Implementation
23/09/2023	4 PM to 6 PM	Mr. Sunil J	Sample Projects Implementation


Resource Person(s)


Coordinator(s)


HoD

Dr. K. SRINIVASA RAO, M.Tech., Ph.D.
Professor & HOD AIML
K.S.R.M. College of Engineering
(Autonomous)
KADAPA- 516 005. (A.P.)

		UMA DEVI(W)	A	une	une	une	une	une	une	une	A	une	une	une	une	une	une	une	A	une	une	une	une	une	une
65	229Y1A3928	MANDURU VIJAY SAI	UJAY SAI	UJAY	UJAY	UJAY	UJAY	UJAY	UJAY	UJAY	UJAY	UJAY	UJAY	UJAY	UJAY	UJAY	UJAY	UJAY	UJAY	UJAY	UJAY	UJAY	UJAY	UJAY	UJAY
66	229Y1A3931	MERUVA YAMINI(W)	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
67	229Y1A3932	MUGHAL MOHAMMED GHANI BAIG	Ciba	Ciba	Ciba	Ciba	Ciba	Ciba	Ciba	Ciba	Ciba	Ciba	Ciba	Ciba	Ciba	Ciba	Ciba	Ciba	Ciba	Ciba	Ciba	Ciba	Ciba	Ciba	Ciba


Coordinator(s)


HoD



KSRM
COLLEGE OF ENGINEERING

(UGC - Autonomous)
Kadapa, Andhra Pradesh, India- 516 005
Approved by AICTE, New Delhi & Affiliated to JNTUA, Anantapuramu.



KSNR
lives on..

VALUE ADDED COURSE ON

AngularJS & MongoDB



AI&ML



Python Programming Lab
(AI202)



28.08.2023
to
23.09.2023

Resource Person

Mr. Sunil J

Assistant professor, Department of AI&ML

Coordinator

Dr. K. Srinivasa Rao

professor, Department of AI&ML

Dr. V.S.S. Morthy
(Principal)

Dr. Kandula Chandra Obul Reddy
(MD, KGI)

Smt. K. Rajeswari
(Correspondent, Secretary, Treasurer)

Sri K. Madan Mohan Reddy
(Vice - Chairman)

Sri K. Raja Mohan Reddy
(Chairman)

ksrmceofficial

www.ksrmce.ac.in

8143731980, 8575697569



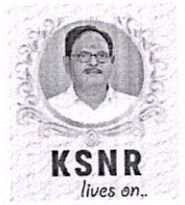
K.S.R.M. COLLEGE OF ENGINEERING

(UGC-AUTONOMOUS)

Kadapa, Andhra Pradesh, India- 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution



Report on Value Added Course on “Angular JS & Mongo DB” From 28/08/2023 to 23/09/2023

Target Group	:	B. Tech Students
Details of Participants	:	67 Students
Co-coordinator(s)	:	Dr. K. Srinivasa Rao
Resource Person(s)	:	Mr. Sunil J
Organizing Department	:	Artificial Intelligence & Machine Learning
Venue	:	Python Programming lab(AI-302)

Description:

The Department of Artificial Intelligence & Machine Learning conducted a Value Added Course on “Angular JS & Mongo DB” from 28/08/2023 to 23/09/2023. The course Resource Persons is Mr. Sunil J, Assistant Professor Department Artificial Intelligence & Machine Learning, KSRMCE.

The main objective of this course is to introduce about the Angularjs and its proficient framework for developing Rich Internet Applications (RIA). It enables developers to create client-side applications using JavaScript in a structured Model-View-Controller (MVC) pattern. AngularJS applications are compatible with multiple web browsers.

HTML is an effective declarative language for creating static documents. The framework lacks sufficient support for application development, leading to the need for workarounds to achieve desired browser behavior when building web applications.

Benefits of using AngularJS Framework

No prerequisite required: AngularJS is designed to be compatible with HTML, CSS, and JavaScript. Learning a new scripting language is unnecessary. Also, HTML, CSS, and JavaScript are relatively simple to pick up, even if you don't know them already.

Simple to expand: because of several built-in features, HTML's functionality may be increased by coupling a particular behavior. It is customized since one may add their own directives to it.

Excellent MVC: To implement the Model-View-Controller (MVC) architecture, many frameworks need the code to be broken up into separate MVC components (Model, View, and Controller). The process is fully automated in Angular. Angular helps programmers save time by organizing their code.

Simple for Testing: Angular is written in the dynamically typed JavaScript language. The expressive potential of angular is immense. However, Angular does not come with a compiler. Therefore, a solid test code for it must be written. Unit testing will be much simpler because it has a dependency injection built right in. Angular is compatible with both unit tests and system tests.



/ksrmce.ac.in

Follow Us:



/ksrmceofficial

About MongoDB:

Fully Managed Database Service: MongoDB Atlas takes care of time-consuming and costly administration tasks so you can get the database resources you need, when you need them.

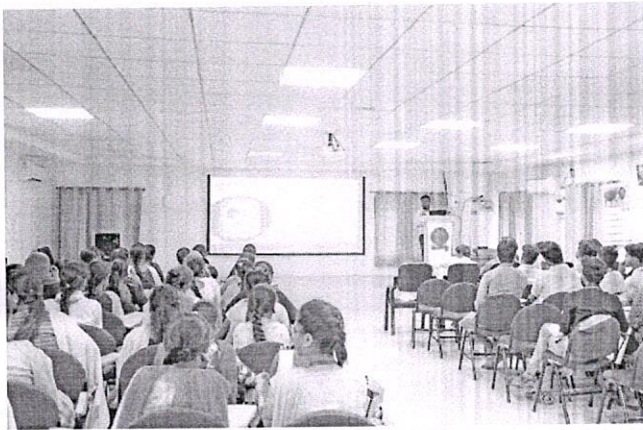
Automated Deployments: Infrastructure provisioning, setup, and deployment is fully automated with MongoDB Atlas. Select a cloud provider, region, instance size, memory, and additional configurations in the Cluster Builder or via the API and be on your way.

Simple Configuration Changes: When requirements and workloads change, MongoDB Atlas makes it easy to make post-deployment modifications to the database. Scale up, add more storage, configure cross-region clusters, create read-only or analytics nodes, and more with the click of a button.

Continuous Improvements: Patches and minor version upgrades are applied automatically so you can take advantage of the latest updates and features. For major version upgrades, you choose when you want them to happen. Atlas makes it easy to spin up environments to test compatibility or try out new features.

Photos

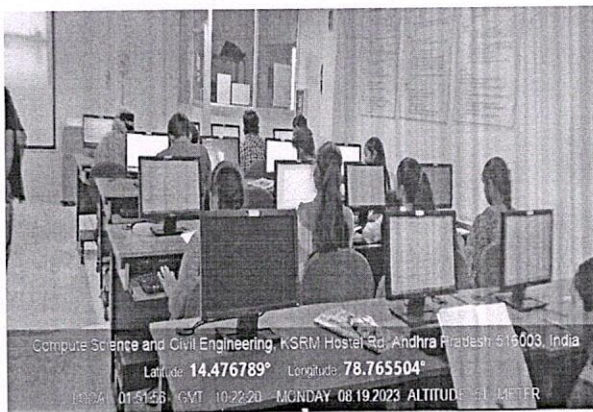
The pictures taken during the course are given below:



Inaugural function of the course




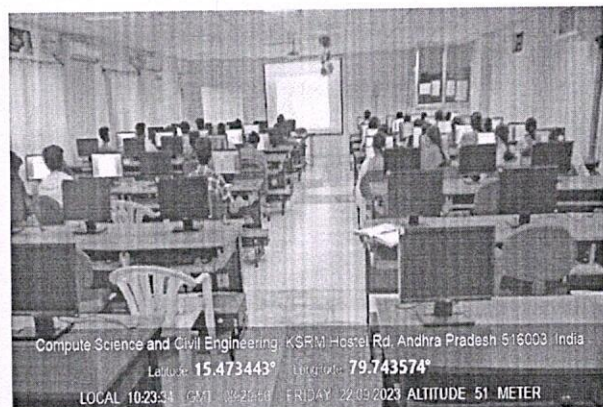
Certificate Distribution by Dr. K. Srinivasa Rao, HoD, AIML



Students while Practicing

Coordinator(s)

 /ksrmce.ac.in



Students while Practicing

HoD

Dr. K. SRINIVASA RAO, M.Tech., Ph.D.
Professor & HOD AIML

K.S.R.M. College of Engineering
(Autonomous)

KADAPA- 516 005. (A.P.)

Follow Us:



K.S.R.M. COLLEGE OF ENGINEERING (AUTONOMOUS), KADAPA-516003
DEPARTMENT OF ARTIFICIAL INTELLIGENCE & MACHINE LEARNING
VALUE ADDED CERTIFICATE COURSE ON
“Angular JS & Mongo DB” FROM 28/08/2023 to 23/09/2023

ASSESSMENT TEST

Roll Number: _____ **Name of the Student:** _____

Time: 20 Min (Objective Questions) **Max.Marks: 20**

Note: Answer the following Questions and each question carries **one** mark.

1. AngularJS is perfect for? (a)
(a) SPAs (b) MPAs (c) DPAs (d) CPAs
2. Which of the following directive is used to bind the application data to the HTML view in AngularJS?
(a) ng-app directive (b) ng-model directive (c) ng-bind directive (d) ng-init directive (c)
3. Which of the following syntax is used to create a module in AngularJS? (c)
(a) var myModule= angular.module();(b) var myModule= new Module();
(c) module("app", []);(d) None of the above
4. Which of the following is used to share data between controller and view in AngularJS?
(a) using Model (b) using services (c) using factory (d) using \$scope (b)
5. Which of the following is not a valid AngularJS filter? (c)
(a) lowercase (b) orderby (c) email (d) currency
6. A module created by using the AngularJS function is called? (b)
(a) module() (b) module() (c) mod() (d) angular module()
7. Which of the following components can be injected as a dependency in AngularJS (d)
(a) Value (b) Factory (c) Constant (d) Application Module
8. AngularJS applications are a mix of which of the following technologies (b)
(a) HTML and PHP (b) HTML and JavaScript (c) HTML and TypeScript (d) PHP and JavaScript
9. Which of the following statement is true in the case of \$routeProvider? (a)
(a) It is a service. (b) It is a module. (c) It is a component. (d) None of the above
10. Which of the following statement justify the working of AngularJS? (c)
(a) module is primarily used to create application modules.
(b) module is used to create AngularJS modules along with its dependent modules.
(c) Both A and B
(d) None of the above

11. Which is the default mode in which the explain() command runs? (a)
(a) queryPlanner(b) executionStats(c) allPlansExecution(d) customExecutionStats
12. Which of the following commands removes a single document that matches the condition that Author is Joe? (d)
(a) db.posts.removeOne({ Author : "Joe" }, 1)(b) db.posts.remove({ Author : "Joe" }, 1)
(C)db.posts.remove({ Author : "Joe" }, {justOne: true})(d) Both b and c
13. What is the equivalent command in MongoDB for the following SQL query? (a)
SELECT * FROM posts WHERE author like "%john%"
(a) db.posts.find({ author: /john/ })(b)db.posts.find({ author: {\$like: /john/} })
(c) db.posts.find({ \$like: {author: /john/} })(d) db.posts.find({ author: /^john^/ })
14. Which of the following best describes MongoDB? (c)
(a) Relational database (b) Spreadsheet program (c) Document-based NoSQL database (d) Graph database
15. In MongoDB, a record is equivalent to a: (c)
(a) Row (b) Table (c) Document (d) Database
16. Which of the following is the default port for MongoDB? (a)
(a) 27017 (b) 8080 (c) 3306 (d) 5432
17. Which MongoDB command is used to display the database you are currently using? (d)
(a) show currentDatabase (b) show db (c) use db (d) db
18. What is the BSON in MongoDB? (d)
(a) A database engine (b) A query language (c) A backup tool (d) Binary representation of JSON
19. What format does MongoDB use for its queries? (c)
(a) SQL (b) XML (C) BSON (d) XQuery
20. Which MongoDB function is used to limit the number of results returned? (c)
(a) db.<collection_name>.count()
(b) db.<collection_name>.skip()
(c) db.<collection_name>.limit()
(d) db.<collection_name>.restrict()

K.S.R.M. COLLEGE OF ENGINEERING (AUTONOMOUS), KADAPA-516003
DEPARTMENT OF ARTIFICIAL INTELLIGENCE & MACHINE LEARNING
VALUE ADDED CERTIFICATE COURSE ON
“Angular JS & Mongo DB” FROM 28/08/2023 to 23/09/2023

ASSESSMENT TEST

Roll Number: 219Y1A3933 **Name of the Student:** N. Srividya Reddy

Time: 20 Min **(Objective Questions)** **Max.Marks: 20**

Note: Answer the following Questions and each question carries **one** mark.

1. AngularJS is perfect for? (a) ✓
(a) SPAs (b) MPAs (c) DPAs (d) CPAs
2. Which of the following directive is used to bind the application data to the HTML view in AngularJS? (B) ✗
(a) ng-app directive (b) ng-model directive (c) ng-bind directive (d) ng-init directive
3. Which of the following syntax is used to create a module in AngularJS? (B) ✗
(a) var myModule= angular.module();(b) var myModule= new Module();
(c) module("app", []);(d) None of the above
4. Which of the following is used to share data between controller and view in AngularJS? (B) ✓
(a) using Model (b) using services (c) using factory (d) using \$scope
5. Which of the following is not a valid AngularJS filter? (B) ✓
(a) lowercase (b) orderby (c) email (d) currency
6. A module created by using the AngularJS function is called? (C) ✓
(a) module() (b) module() (c) mod() (d) angular module()
7. Which of the following components can be injected as a dependency in AngularJS (D) ✓
(a) Value (b) Factory (c) Constant (d) Application Module
8. AngularJS applications are a mix of which of the following technologies (B) ✓
(a) HTML and PHP (b) HTML and JavaScript (c) HTML and TypeScript (d) PHP and JavaScript
9. Which of the following statement is true in the case of \$routeProvider? (A) ✓
(a) It is a service. (b) It is a module. (c) It is a component. (d) None of the above
10. Which of the following statement justify the working of AngularJS? (C) ✓
(a) module is primarily used to create application modules.
(b) module is used to create AngularJS modules along with its dependent modules.
(c) Both A and B
(d) None of the above

11. Which is the default mode in which the explain() command runs? (A) ✓
(a) queryPlanner(b) executionStats(c) allPlansExecution(d) customExecutionStats
12. Which of the following commands removes a single document that matches the condition that Author is Joe? (D) ✓
(a) db.posts.removeOne({ Author : "Joe" }, 1)(b) db.posts.remove({ Author : "Joe" }, 1)
(C)db.posts.remove({ Author : "Joe" }, {justOne: true})(d) Both b and c
13. What is the equivalent command in MongoDB for the following SQL query? (A) ✓
SELECT * FROM posts WHERE author like "%john%"
(a) db.posts.find({ author: /john/ })(b)db.posts.find({ author: {\$like: /john/} })
(c) db.posts.find({ \$like: {author: /john/} })(d) db.posts.find({ author: /^john^/ })
14. Which of the following best describes MongoDB? (C) ✓
(a) Relational database (b) Spreadsheet program (c) Document-based NoSQL database (d) Graph database
15. In MongoDB, a record is equivalent to a: (C) ✓
(a) Row (b) Table (c) Document (d) Database
16. Which of the following is the default port for MongoDB? (A) ✓
(a) 27017 (b) 8080 (c) 3306 (d) 5432
17. Which MongoDB command is used to display the database you are currently using? (D) ✓
(a) show currentDatabase (b) show db (c) use db (d) db
18. What is the BSON in MongoDB? (B) ✓
(a) A database engine (b) A query language(c) A backup tool (d) Binary representation of JSON
19. What format does MongoDB use for its queries? (C) ✓
(a) SQL (b) XML (C) BSON (d) XQuery
20. Which MongoDB function is used to limit the number of results returned? (E) ✓
(a) db.<collection_name>.count()
(b) db.<collection_name>.skip()
(c) db.<collection_name>.limit()
(d) db.<collection_name>.restrict()

K.S.R.M. COLLEGE OF ENGINEERING (AUTONOMOUS), KADAPA-516003
DEPARTMENT OF ARTIFICIAL INTELLIGENCE & MACHINE LEARNING
VALUE ADDED CERTIFICATE COURSE ON
"Angular JS & Mongo DB" FROM 28/08/2023 to 23/09/2023

ASSESSMENT TEST

Roll Number: 21941A0409 Name of the Student: B. Charan

Time: 20 Min (Objective Questions) Max.Marks: 20

Note: Answer the following Questions and each question carries **one** mark.

1. AngularJS is perfect for? (a) ✓
(a) SPAs (b) MPAs (c) DPAs (d) CPAs
2. Which of the following directive is used to bind the application data to the HTML view in AngularJS? (c) ✓
(a) ng-app directive (b) ng-model directive (c) ng-bind directive (d) ng-init directive
3. Which of the following syntax is used to create a module in AngularJS? (c) ✓
(a) var myModule= angular.module();(b) var myModule= new Module();
(c) module("app", []);(d) None of the above
4. Which of the following is used to share data between controller and view in AngularJS? (b) ✓
(a) using Model (b) using services (c) using factory (d) using \$scope
5. Which of the following is not a valid AngularJS filter? ()
(a) lowercase (b) orderby (c) email (d) currency
6. A module created by using the AngularJS function is called? (b) ✓
(a) module() (b) module() (c) mod() (d) angular module()
7. Which of the following components can be injected as a dependency in AngularJS (d) ✓
(a) Value (b) Factory (c) Constant (d) Application Module
8. AngularJS applications are a mix of which of the following technologies (a) ✗
(a) HTML and PHP (b) HTML and JavaScript (c) HTML and TypeScript (d) PHP and JavaScript
9. Which of the following statement is true in the case of \$routeProvider? (b) ✗
(a) It is a service. (b) It is a module. (c) It is a component. (d) None of the above
10. Which of the following statement justify the working of AngularJS? (c) ✓
(a) module is primarily used to create application modules.
(b) module is used to create AngularJS modules along with its dependent modules.
(c) Both A and B
(d) None of the above

11. Which is the default mode in which the explain() command runs? (a) ✓
- (a) queryPlanner(b) executionStats(c) allPlansExecution(d) customExecutionStats
12. Which of the following commands removes a single document that matches the condition that Author is Joe? (b) ✓
- (a) db.posts.removeOne({ Author : "Joe" }, 1)(b) db.posts.remove({ Author : "Joe" }, 1)
(C)db.posts.remove({ Author : "Joe" }, {justOne: true})(d) Both b and c
13. What is the equivalent command in MongoDB for the following SQL query? (c) ✓
- SELECT * FROM posts WHERE author like "%john%"
- (a) db.posts.find({ author: /john/ })(b)db.posts.find({ author: {\$like: /john/} })
(c) db.posts.find({ \$like: {author: /john/} })(d) db.posts.find({ author: /^john^/ })
14. Which of the following best describes MongoDB? (a) ✓
- (a) Relational database (b) Spreadsheet program (c) Document-based NoSQL database (d)Graph database
15. In MongoDB, a record is equivalent to a: (a) ✓
- (a) Row (b) Table (c) Document (d) Database
16. Which of the following is the default port for MongoDB? () ✗
- (a) 27017 (b) 8080 (c) 3306 (d)5432
17. Which MongoDB command is used to display the database you are currently using? (d) ✓
- (a) show currentDatabase (b) show db (c) use db (d) db
18. What is the BSON in MongoDB? (d) ✓
- (a) A database engine (b)A query language(c) A backup tool (d) Binary representation of JSON
19. What format does MongoDB use for its queries? (e) ✓
- (a) SQL (b) XML (C) BSON (d) XQuery
20. Which MongoDB function is used to limit the number of results returned? (c) ✓
- (a) db.<collection_name>.count()
(b) db.<collection_name>.skip()
(c) db.<collection_name>.limit()
(d) db.<collection_name>.restrict()

K.S.R.M. COLLEGE OF ENGINEERING (AUTONOMOUS), KADAPA-516003
DEPARTMENT OF ARTIFICIAL INTELLIGENCE & MACHINE LEARNING
VALUE ADDED CERTIFICATE COURSE ON
“Angular JS & Mongo DB” FROM 28/08/2023 to 23/09/2023

04/20 44

ASSESSMENT TEST

Roll Number: 22941A3932 **Name of the Student:** M. Ghani Baig

Time: 20 Min **(Objective Questions)** **Max.Marks: 20**

Note: Answer the following Questions and each question carries **one** mark.

1. AngularJS is perfect for?
(a) SPAs (b) MPAs (c) DPAs (d) CPAs (d) ✓
2. Which of the following directive is used to bind the application data to the HTML view in AngularJS?
(a) ng-app directive (b) ng-model directive (c) ng-bind directive (d) ng-init directive (c) ✓
3. Which of the following syntax is used to create a module in AngularJS?
(a) var myModule= angular.module();(b) var myModule= new Module();
(c) module("app", []);(d) None of the above (a) ✓
4. Which of the following is used to share data between controller and view in AngularJS?
(a) using Model (b) using services (c) using factory (d) using \$scope (b) ✓
5. Which of the following is not a valid AngularJS filter?
(a) lowercase (b) orderby (c) email (d) currency (a) ✓
6. A module created by using the AngularJS function is called?
(a) module() (b) module() (c) mod() (d) angular module() (d) ✓
7. Which of the following components can be injected as a dependency in AngularJS
(a) Value (b) Factory (c) Constant (d) Application Module (b) ✓
8. AngularJS applications are a mix of which of the following technologies
(a) HTML and PHP (b) HTML and JavaScript (c) HTML and TypeScript (d) PHP and JavaScript (c) ✓
9. Which of the following statement is true in the case of \$routeProvider?
(a) It is a service. (b) It is a module. (c) It is a component. (d) None of the above (d) ✓
10. Which of the following statement justify the working of AngularJS?
(a) module is primarily used to create application modules.
(b) module is used to create AngularJS modules along with its dependent modules.
(c) Both A and B (c) ✓
(d) None of the above

11. Which is the default mode in which the `explain()` command runs? (c) ✓
- (a) `queryPlanner`(b) `executionStats`(c) `allPlansExecution`(d) `customExecutionStats`
12. Which of the following commands removes a single document that matches the condition that Author is Joe? (a) ✓
- (a) `db.posts.removeOne({ Author : "Joe" }, 1)`(b) `db.posts.remove({ Author : "Joe" }, 1)`
(c) `db.posts.remove({ Author : "Joe" }, {justOne: true})`(d) Both b and c
13. What is the equivalent command in MongoDB for the following SQL query? (c) ✓
- `SELECT * FROM posts WHERE author like "%john%"`
- (a) `db.posts.find({ author: /john/ })`(b) `db.posts.find({ author: {$like: /john/} })`
(c) `db.posts.find({ $like: {author: /john/} })`(d) `db.posts.find({ author: /^john^/ })`
14. Which of the following best describes MongoDB? (b) ✓
- (a) Relational database (b) Spreadsheet program (c) Document-based NoSQL database (d) Graph database
15. In MongoDB, a record is equivalent to a: (d) ✓
- (a) Row (b) Table (c) Document (d) Database
16. Which of the following is the default port for MongoDB? (a) ✓
- (a) 27017 (b) 8080 (c) 3306 (d) 5432
17. Which MongoDB command is used to display the database you are currently using? (a) ✓
- (a) `show currentDatabase` (b) `show db` (c) `use db` (d) `db`
18. What is the BSON in MongoDB? (d) ✓
- (a) A database engine (b) A query language (c) A backup tool (d) Binary representation of JSON
19. What format does MongoDB use for its queries? (d) ✓
- (a) SQL (b) XML (c) BSON (d) XQuery
20. Which MongoDB function is used to limit the number of results returned? (c) ✓
- (a) `db.<collection_name>.count()`
(b) `db.<collection_name>.skip()`
(c) `db.<collection_name>.limit()`
(d) `db.<collection_name>.restrict()`

K.S.R.M. COLLEGE OF ENGINEERING (AUTONOMOUS), KADAPA-516003
DEPARTMENT OF ARTIFICIAL INTELLIGENCE & MACHINE LEARNING
VALUE ADDED COURSE ON
“AngularJS & MongoDB” FROM 28/08/2023 to 23/09/2023

AWARD LIST

S.NO	Roll Number	Name of the Student	Marks Obtained
1	219Y1A3904	AVULA AMITH	12
2	219Y1A3905	BHUMIREDDY VENKATA RAVI KUMAR REDDY	10
3	219Y1A3906	BORANAPU SAI TEJASWINI (W)	15
4	219Y1A3907	CHANDRAGIRI KHAMALESHWAR	11
5	219Y1A3908	CHAPPIDI ASHA DEEPTHI REDDY (W)	18
6	219Y1A3909	CHENNA VENKATESH	17
7	219Y1A3910	DAPPELLA VISHNU TEJA	16
8	219Y1A3916	GUMMALLA BALAJI REDDY	07
9	219Y1A3917	KAKUMANI HARIKA (W)	11
10	219Y1A3918	KANCHAMREDDY JAHNAVI REDDY (W)	09
11	219Y1A3920	KAYAPATI KARTHIKEYA · BABRUVAHANA	12
12	219Y1A3921	KENGERI UMESH	14
13	219Y1A3922	KRUPAKARAN KARTHIKAN	11
14	219Y1A3929	MOGHAL JUNAID BAIG	18
15	219Y1A3930	MURTHY MEGHANA (W)	16
16	219Y1A3931	NAGELLA ARJUN	09
17	219Y1A3932	NAKKA GURU AAKARSH	10
18	219Y1A3933	NALIPI SRIDHAR REDDY	18
19	219Y1A3941	PASUPULA KALYAN CHAKRAVARTHI	15
20	219Y1A3942	PATIL PRADEEP	09
21	219Y1A3943	POCHIREDDY POCHAMMA (W)	11
22	219Y1A3944	PRAGATHI GAJULA (W)	17
23	219Y1A3948	SHAIK KATTUBADI IMSHAD (W)	10
24	219Y1A3949	SHAIK MOHAMMED ABDUL MOIZE	16
25	219Y1A3950	SHAIK MOHAMMED FAHEEMULLAH	11
26	219Y1A3951	SHAIK MOHAMMED GAFFAR ALI	15
27	219Y1A3954	SIRIGIREDDY REDDAIAH	12
28	219Y1A3955	SOMISETTY HEMANTH KUMAR	10
29	219Y1A3956	SURYA SREENATH	12
30	219Y1A3957	SYED YEZDAN AHAMED	08
31	229Y5A3902	DERANGULA NAGALAKSHMI(W)	12
32	229Y5A3903	KOTLO DHANUSH	16
33	229Y5A3905	VADDE HEMANTHKUMAR	17
34	219Y1A0404	ANNAGIRI KAVYA (W)	10
35	219Y1A0405	APPAKONDU CHANDRIKA (W)	17
36	219Y1A0406	ARAVEEDU TRIVIKRAM	07
37	219Y1A0407	AVULA BHANU PRAKASH	10
38	219Y1A0408	B POORNIMA (W)	16
39	219Y1A0409	BADUGU CHARAN	13

40	219Y1A0476	KOVVURI AKHILA (W)	10
41	219Y1A0477	KOYAVALLU CHAMUNDESWARI (W)	18
42	219Y1A0478	KUNDA CHAITANYA SAI	12
43	219Y1A0483	KURUVA MALLESH	17
44	219Y1A0484	LINGAM HEMANTH KUMAR	10
45	219Y1A0488	MALLE VEERA BRAHMENDRA	09
46	219Y1A0489	MALLE VENKATADRI	16
47	219Y1A0492	MANGALI JANARDHAN	12
48	219Y1A0494	MANJULA VENKATA SUNIL KUMAR	07
49	219Y1A0495	MANUPATI MALLIKARJUNA	09
50	219Y1A0536	DERANGULA VENKATA NAGA MAHINDRA	10
51	219Y1A0537	DERANGULA VENKATA PRITHVIRAJ ARYAN	15
52	219Y1A0538	DHANIREDDY NANDITHA REDDY (W)	12
53	219Y1A0539	DIDDEKUNTA VENKATA PRADEEP KUMAR	16
54	219Y1A05B0	MITTAMANUPALLE HEMALATHA (W)	12
55	219Y1A05B1	MOGALI SESA ROHITH	10
56	219Y1A05B2	MOPURU BINDHU LATHA (W)	12
57	219Y1A05I2	VANKARA MANASA (W)	09
58	219Y1A05I3	VARA PRADEEP REDDY	12
59	219Y1A05I4	VARIKUTI SREELAKSHMI (W)	18
60	219Y1A05I5	VUSUVANDLA RAJESWARI (W)	10
61	229Y1A3920	KATIGANDLA SOWMYA(W)	11
62	229Y1A3921	KODATHALA BHAVANA(W)	12
63	229Y1A3926	LAKSHMI VAISHNAVI(W)	17
64	229Y1A3927	MAJJARI UMA DEVI(W)	10
65	229Y1A3928	MANDURU VIJAY SAI	12
66	229Y1A3931	MERUVA YAMINI(W)	17
67	229Y1A3932	MUGHAL MOHAMMED GHANI BAIG	04


Coordinator


HoD

3920 KATIGANDLA SOWMYA
3921 KODATHALA BHAVANA
3926 LAKSHMI VAISHNAVI
3927 MAJJARI UMA DEVI
3928 MANDURU VIJAY SAI
3931 MERUVA YAMINI
3932 MUGHAL MOHAMMED GHANI BAIG



K.S.R.M. COLLEGE OF ENGINEERING

(UGC - Autonomous)

Kadapa, Andhra Pradesh, India- 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.



KSNR
lives on..

Certificate of Completion

This to certify that Mr/Mrs. A. Chandrika Bearing

the Roll Number 2194A0405 has Successfully Completed Value Added

Course on "AngularJS & MongoDB" from 28th August 2023 to 23rd September 2023,

Organized by Department of AIML, KSRMCE, Kadapa.

KSRao

Coordinator

KSRao

HOD

K.S.R. MURTHY, M.Tech., Ph.D.
Professor & HOD AIML
K.S.R.M. College of Engineering
(Autonomous)
KADAPA-516003 (A.P.)

V. S. S. Murthy

Principal



K.S.R.M. COLLEGE OF ENGINEERING

(UGC - Autonomous)

Kadapa, Andhra Pradesh, India- 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

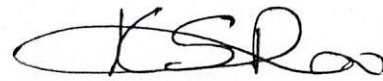


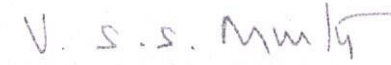
KSNR
lives on..

Certificate of Completion

This to certify that Mr/Mrs. B. Poorhima Bearing
the Roll Number 21941A0408 has Successfully Completed Value Added
Course on "AngularJS & MongoDB" from 28th August 2023 to 23rd September 2023,
Organized by Department of AIML, KSRMCE, Kadapa.


Coordinator


HOD
Dr. K. SRINIVASA RAO, M.Tech., Ph.D.
Professor & HOD AIML
K.S.R.M. College of Engineering
(Autonomous)
KADAPA-516 003 (A.P.)


Principal



K.S.R.M. COLLEGE OF ENGINEERING

(UGC - Autonomous)

Kadapa, Andhra Pradesh, India- 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.



KSNR
lives on..

Certificate of Completion

This to certify that Mr/Mrs. C ASHA DEEPTHI REDDY Bearing

the Roll Number 219Y1A3908 has Successfully Completed Value Added

Course on "AngularJS & MongoDB" from 28th August 2023 to 23rd September 2023,

Organized by Department of AIML, KSRMCE, Kadapa.

KSRao

Coordinator

KSRao

HOD


Dr. K. SRINIVASA RAO, M.Tech., Ph.D.
Professor & HOD AIML
K.S.R.M. College of Engineering
(Autonomous)

KADAPA- 516 005.

V. S. S. Murthy

Principal

Feedback form on Value Added Course "AngularJS & MongoDB" from **28/08/2023** to **23/09/2023**

 sunil.j@ksrmce.ac.in (not shared) [Switch account](#)



* Required

Roll Number *

Your answer

Name of the Student *

Your answer

The objectives of the Value Added Course were met (Objective) *

- Excellent
- Good
- Satisfactory
- Poor



The content of the course was organized and easy to follow (Delivery) *

- Excellent
- Good
- Satisfactory
- Poor

The Resource Persons were well prepared and able to answer any question (Interaction) *

- Excellent
- Good
- Satisfactory
- Poor

The exercises/role play were helpful and relevant (Syllabus Coverage) *

- Excellent
- Good
- Satisfactory
- Poor



The Value Added Course satisfy my expectation as a value added Programme (Course Satisfaction) *

- Excellent
- Satisfactory
- Good
- Poor

Any Issues

Your answer

Submit

Clear form

Never submit passwords through Google Forms.

This form was created inside of KSRM College of Engineering. [Report Abuse](#)

Google Forms



K.S.R.M. COLLEGE OF ENGINEERING

Demartment of Artificial Intelligence & Machine Learning, Feedback report on Value Added Course on Angular JS & MongoDB

Timestamp	Name	Roll Number	Email	Branch	The objectives of the Value Added Course were met (Objective)	The content of the course was organized and easy to follow (Delivery) *	The Resource Persons were well prepared and able to answer any question (Interaction)	The exercises/role play were helpful and relevant (Syllabus Coverage) *	The Value Added Course satisfy my expectation as a value added Programme (Course Satisfaction)	Any Issues
09/24/2023 14:12:00	ATTAR MOHAMMED SUFIYAN	219Y1A3901	1A3901@ksrmce	AIML	Excellent	Good	Satisfactory	Good	Good	No
09/24/2023 14:13:33	AVULA AMITH	219Y1A3902	1A3902@ksrmce	AIML	Good	Satisfactory	Good	Excellent	Excellent	
09/24/2023 14:16:23	BHUMIREDDY VENKATA RAVI KUMAR REDDY	219Y1A3903	1A3903@ksrmce	AIML	Excellent	Good	Good	Satisfactory	Good	
09/24/2023 14:16:30	BORANAPU SAI TEJASWINI (W)	219Y1A3904	1A3904@ksrmce	AIML	Good	Good	Excellent	Good	Good	
09/24/2023 14:17:34	CHANDRAGIRI KHAMALESHWAR	219Y1A3905	1A3905@ksrmce	AIML	Satisfactory	Good	Excellent	Excellent	Good	
09/24/2023 14:17:35	CHAPPIDI ASHA DEEPTHI REDDY (W)	219Y1A3906	1A3906@ksrmce	AIML	Excellent	Good	Excellent	Good	Good	
09/24/2023 14:17:40	CHENNA VENKATESH	219Y1A3907	1A3907@ksrmce	AIML	Excellent	Excellent	Good	Satisfactory	Good	
09/24/2023 14:18:20	DAPPELLA VISHNU TEJA	219Y1A3908	1A3908@ksrmce	AIML	Excellent	Satisfactory	Good	Excellent	Satisfactory	

09/24/2023 14:18:22	DEEPAK REDDY SIRIGIREDDY	219Y1A3909	1A3909@ksrmce	AIML	Excellent	Satisfactory	Excellent	Excellent	Satisfactory	
09/24/2023 14:18:27	DODLA SANTHOSH REDDY	219Y1A3910	1A3910@ksrmce	AIML	Satisfactory	Good	Satisfactory	Good	Satisfactory	
09/24/2023 15:19:40	ATTAR MOHAMMED SUFİYAN	219Y1A3911	1A3911@ksrmce	AIML	Excellent	Good	Satisfactory	Good	Good	
09/24/2023 15:20:30	AVULA AMITH	219Y1A3912	1A3912@ksrmce	AIML	Good	Good	Good	Excellent	Good	
15:41:42	MANCHALA CANCA YATISH	219Y1A3919	1A3919@ksrmce	AIML	Good	Good	Satisfactory	Excellent	Excellent	
09/24/2023 16:20:22	KAYAPATI KARTHIKEYA BABRUVAHANA	219Y1A3920	1A3920@ksrmce	AIML	Excellent	Good	Satisfactory	Good	Good	
09/24/2023 16:21:44	KENGERI UMESH	219Y1A3921	1A3921@ksrmce	AIML	Satisfactory	Satisfactory	Good	Good	Good	
09/24/2023 16:32:23	KRUPAKARAN KARTHIKAN	219Y1A3922	1A3922@ksrmce	AIML	Satisfactory	Excellent	Good	Excellent	Good	
09/24/2023 16:32:46	KURUVA MADHUKRISHN A	219Y1A3923	1A3923@ksrmce	AIML	Good	Good	Satisfactory	Satisfactory	Good	
09/24/2023 16:40:20	MALIREDDY SAI CHARAN REDDY	219Y1A3924	1A3924@ksrmce	AIML	Good	Satisfactory	Good	Good	Good	
09/24/2023 16:44:32	MANCHALA RANJITH KUMAR	219Y1A3925	1A3925@ksrmce	AIML	Good	Good	Excellent	Excellent	Excellent	
09/24/2023 16:46:49	NALLAMALLA MEGHANA (W)	219Y1A3934	1A3934@ksrmce	AIML	Good	Good	Satisfactory	Excellent	Excellent	
09/24/2023 17:41:74	NAYANAGARI VASAVI (W)	219Y1A3935	1A3935@ksrmce	AIML	Good	Good	Satisfactory	Excellent	Excellent	

09/24/2023 17:43:51	OBULAREDDY GARI MANASA (W)	219Y1A3936	1A3936@ksrmce	AIML	Excellent	Good	Good	Good	Satisfactory	
09/24/2023 17:41:52	P LIKITHA (W)	219Y1A3937	1A3937@ksrmce	AIML	Excellent	Good	Good	Excellent	Satisfactory	
09/24/2023 17:43:35	P SOWJANYA (W)	219Y1A3938	1A3938@ksrmce	AIML	Excellent	Good	Satisfactory	Excellent	Good	
09/24/2023 17:51:54	PALLETI RAM SAI PRAVALLIKA (W)	219Y1A3939	1A3939@ksrmce	AIML	Excellent	Satisfactory	Good		Good	
09/24/2023 18:41:55	PASALA NAGA PRIYANKA (W)	219Y1A3940	1A3940@ksrmce	AIML	Good	Good	Good	Good	Good	
09/24/2023 18:41:55	SHAIK MOHAMMED ABDUL MOIZE	219Y1A3949	1A3949@ksrmce	AIML	Excellent	Excellent	Good	Good	Good	
09/24/2023 18:41:55	SHAIK MOHAMMED FAHEEMULLAH	219Y1A3950	1A3950@ksrmce	AIML	Good	Excellent		Excellent	Excellent	
09/24/2023 18:41:58	SHAIK MOHAMMED GAFFAR ALI	219Y1A3951	1A3951@ksrmce	AIML	Good	Satisfactory	Good	Good	Good	
09/24/2023 18:44:59	SHAIK MOHAMMED SAAD	219Y1A3952	1A3952@ksrmce	AIML	Good	Good	Good	Satisfactory	Good	
09/24/2023 18:44:59	SHAIK SHUAIB	219Y1A3953	1A3953@ksrmce	AIML	Good	Good	Satisfactory	Good	Satisfactory	
09/24/2023 18:45:30	SIRIGIREDDY REDDAIAH	219Y1A3954	1A3954@ksrmce	AIML	Excellent	Excellent	Excellent	Satisfactory	Excellent	
09/24/2023 18:45:42	SURYA SREENATH	219Y1A3956	1A3956@ksrmce	AIML	Good	Excellent	Excellent	Good	Good	
09/24/2023 18:46:20	SYED YEZDAN AHAMED	219Y1A3957	1A3957@ksrmce	AIML	Good	Good	Good	Satisfactory	Good	

09/24/2023 18:46:21	THADIGOTLA NITHYA LAVANYA (W)	219Y1A3958	1A3958@ksrmce	AIML	Good	Good	Excellent	Good	Excellent	
09/24/2023 18:46:21	V KULADEEP	219Y1A3960	1A3960@ksrmce	AIML	Good	Good	Good	Good	Excellent	
09/24/2023 18:47:34	VALLEPU AJAY	219Y1A3961	1A3961@ksrmce	AIML	Excellent	Good	Good	Satisfactory	Excellent	
09/24/2023 18:47:37	VALLURU RUCHITHA (W)	219Y1A3962	1A3962@ksrmce	AIML	Good	Good	Excellent	Good	Good	
09/24/2023 18:47:42	VANKADHARA GURU JAHNAVI (W)	219Y1A3963	1A3963@ksrmce	AIML	Good	Good	Excellent	Satisfactory	Excellent	
09/24/2023 18:47:50	YADATI LAKSHMISRAV ANI (W)	219Y1A3964	1A3964@ksrmce	AIML	Excellent	Satisfactory	Excellent	Satisfactory	Excellent	
09/24/2023 19:32:50	YERUGUDIPAD U CHENNA KESAVA	219Y1A3965	1A3965@ksrmce	AIML	Good	Satisfactory	Excellent	Satisfactory	Good	
09/24/2023 19:40:21	ACHHAMMAGA RI SUNEETHA (W)	219Y1A0501	1A0501@ksrmce	CSE	Excellent	Excellent	Good	Good	Excellent	
09/24/2023 19:50:50	AEGUVAGADD A RANGANADH	219Y1A0502	1A0502@ksrmce	CSE	Good	Good	Good	Excellent	Good	
09/24/2023 19:54:21	ANKAIAHGARI SUBHADRA (W)	219Y1A0503	1A0503@ksrmce	CSE	Good	Good	Good	Good	Good	
09/24/2023 19:60:74	ANNAREDDY BINDUSREE (W)	219Y1A0504	1A0504@ksrmce	CSE	Excellent	Good	Excellent	Excellent	Good	
09/24/2023 19:61:75	BALAGANI NAVEEN	219Y1A0509	1A0509@ksrmce	CSE	Satisfactory	Excellent	Excellent	Satisfactory	Good	
09/24/2023 19:61:75	BATHALA NAGARAJU	219Y1A0510	1A0510@ksrmce	CSE	Excellent	Excellent	Excellent	Excellent	Excellent	
09/24/2023 19:61:75	CHEMUDURU SATHYA CHAKRADHAR	219Y1A0520	1A0520@ksrmce	CSE	Excellent	Good	Good	Good	Satisfactory	

09/24/2023 20:01:21	CHIMALAPENT A CHINTU	219Y1A0521	1A0521@ksrmce	CSE	Excellent	Satisfactory	Good	Good	Good	
09/24/2023 20:01:21	CHINNANNAGA RI PULLAREDDY	219Y1A0522	1A0522@ksrmce	CSE	Excellent	Good	Excellent	Good	Excellent	
10/25/2023 14:12:00	CHINNEGOWLL A DEVENDRA PRASAD	219Y1A0523	1A0523@ksrmce	CSE	Satisfactory	Excellent	Excellent	Satisfactory	Good	
14:13:33	KARAKONTA LARI KRISHNA KASA	219Y1A0577	1A0577@ksrmce	CSE	Excellent	Excellent	Excellent	Excellent	Excellent	
10/25/2023 14:16:23	UMAMAHESWA RI (W)	219Y1A0578	1A0578@ksrmce	CSE	Excellent	Good	Good	Good	Satisfactory	
10/25/2023 14:16:30	KASUNURU SURYA KANTHAM (W)	219Y1A0579	1A0579@ksrmce	CSE	Excellent	Satisfactory	Good	Good	Good	
10/25/2023 14:17:34	KUDAMALA RAVITEJA	219Y1A0589	1A0589@ksrmce	CSE	Excellent	Good	Excellent	Good	Excellent	
10/25/2023 14:17:35	KUMMETHA SHAHEEN (W)	219Y1A0591	1A0591@ksrmce	CSE	Excellent	Good	Good	Good	Satisfactory	
10/25/2023 14:17:40	KURRA SIVA NANDINI (W)	219Y1A0592	1A0592@ksrmce	CSE	Excellent	Satisfactory	Good	Good	Good	
10/25/2023 14:18:20	MANDLA LAKSHMI BHAVANI (W)	219Y1A05A7	1A05A7@ksrmce	CSE	Excellent	Good	Excellent	Good	Excellent	
10/25/2023 14:18:22	MANNAJI MOHAMMED ASIF	219Y1A05A8	1A05A8@ksrmce	CSE	Excellent	Good	Good	Good	Satisfactory	
10/25/2023 14:18:27	MARTHANI VINILA (W)	219Y1A05A9	1A05A9@ksrmce	CSE	Excellent	Satisfactory	Good	Good	Good	
10/25/2023 14:18:28	MITTAMANUPA LLE HEMALATHA (W)	219Y1A05B0	1A05B0@ksrmce	CSE	Excellent	Good	Excellent	Good	Excellent	
10/25/2023 14:18:28	BARREMUKALA ESWAR	229Y1A3904	229Y1A3904@k srmce.ac.in	AIML	Excellent	Good	Good	Good	Satisfactory	

10/25/2023 14:18:28	BOYA POOJITHA(W)	229Y1A3905	<a href="mailto:229Y1A3905@k
srmce.ac.in">229Y1A3905@k srmce.ac.in	AIML	Excellent	Satisfactory	Good	Good	Good	
10/25/2023 14:18:28	C SAI SRI LAKSHMI(W)	229Y1A3906	<a href="mailto:229Y1A3906@k
srmce.ac.in">229Y1A3906@k srmce.ac.in	AIML	Excellent	Good	Excellent	Good	Excellent	
10/25/2023 14:18:28	HARI SREE MUNELLA(W)	229Y1A3916	<a href="mailto:229Y1A3916@k
srmce.ac.in">229Y1A3916@k srmce.ac.in	AIML	Excellent	Good	Good	Good	Satisfactory	
10/25/2023 14:18:28	KALINGIRI BHARGAVA SIDDARTHA	229Y1A3917	<a href="mailto:229Y1A3917@k
srmce.ac.in">229Y1A3917@k srmce.ac.in	AIML	Excellent	Satisfactory	Good	Good	Good	
10/25/2023 14:18:28	KAMBOJI GANGA BHAVANA(W)	229Y1A3918	<a href="mailto:229Y1A3918@k
srmce.ac.in">229Y1A3918@k srmce.ac.in	AIML	Excellent	Good	Excellent	Good	Excellent	

KSRao
co-ordinator

KSRao
HOD

Dr. K. SRINIVASA RAO, M.Tech., Ph.D,
Professor & HOD AIML
K.S.R.M. College of Engineering
(Autonomous)
KADAPA- 516 005. (A.P.)

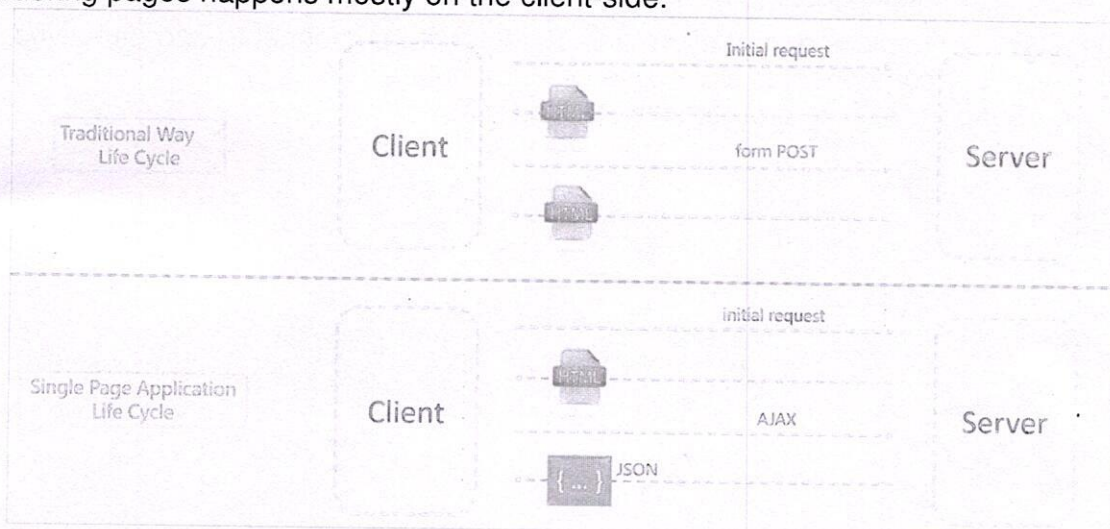
SCS1620 User Interface Technologies

Unit V – AngularJS and ReactJS

Introduction to Angular 4.0 - Needs & Evolution – Features – Setup and Configuration – Components and Modules – Templates – Change Detection - Directives – Data Binding - Pipes – Nested Components. Template Driven Forms - Model Driven Forms or Reactive Forms - Custom Validators. Introduction to ReactJS- React Components- Build a simple React component- React internals – Component inter communication- Component composition- Component styling

Introduction

Angular is a client-side JavaScript framework that was specifically designed to help developers build SPAs (Single Page Applications) in accordance with best practices for web development. Single-page application (or SPA) are applications that are accessed via a web browser like other websites but offer more dynamic interactions resembling native mobile and desktop apps. The most notable difference between a regular website and SPA is the reduced amount of page refreshes. SPAs have a heavier usage of AJAX- a way to communicate with back-end servers without doing a full page refresh to get data loaded into our application. As a result, the process of rendering pages happens mostly on the client-side.



Single Page Application vs Traditional Web Application

Angular is a TypeScript-based open-source front-end web application platform led by the Angular Team at Google and by a community of individuals and corporations. Angular is a complete rewrite from the same team that built AngularJS.

- The architecture of an Angular application is different from AngularJS. The main building blocks for Angular are modules, components, templates, metadata, data binding, directives, services and dependency injection. We will be looking at it in a while.
- Angular was a complete rewrite of AngularJS.
- Angular does not have a concept of “scope” or controllers instead, it uses a hierarchy of components as its main architectural concept.
- Angular has a simpler expression syntax, focusing on “[]” for property binding, and “()” for event binding

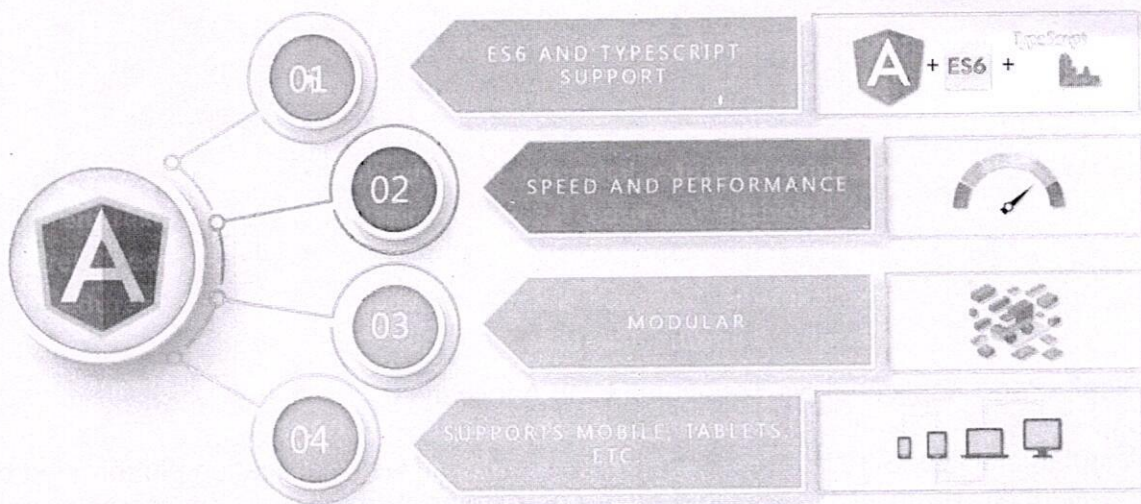
- **Mobile development** – Desktop development is much easier when mobile performance issues are handled first. Thus, Angular first handles mobile development.
- **Modularity** – Angular follows modularity. Similar functionalities are kept together in same modules. This gives Angular a lighter & faster core.

Angular recommends the use of Microsoft's TypeScript language, which introduces the following features:

- Class-based Object Oriented Programming
- Static Typing

TypeScript is a superset of ECMAScript 6 (ES6) and is backward compatible with ECMAScript 5. Angular also includes the benefits of ES6:

- Iterators
- For/Of loops
- Reflection
- Improved dependency injection – bindings make it possible for dependencies to be named
- Dynamic loading
- Asynchronous template compilation
- Simpler Routing
- Replacing controllers and \$scope with components and directives – a component is a directive with a template
- Support reactive programming using RxJS



Features of Angular

Cross Platform

- **Progressive web apps:** It uses modern web platform capabilities to deliver an app-like experience. It gives high performance, offline, and zero-step installation. So, working with Angular is pretty much easy.
- **Native:** Builds native mobile apps with strategies using Ionic Framework, NativeScript, and React Native.
- **Desktop:** Create desktop-installed apps across Mac, Windows, and Linux using the same Angular methods you've learned for the web plus.

Speed and Performance

- **Code generation:** Angular turns your templates into code that's highly optimized for JavaScript virtual machines, giving you all the benefits of hand-written code with the productivity of a framework.
- **Universal:** Any technology can be used with Angular for serving the application like node.js, .NET, PHP and other servers.
- **Code splitting:** Angular apps load quickly with the new Component Router, which delivers automatic code-splitting, so users only load code required to render the view they request.

Productivity

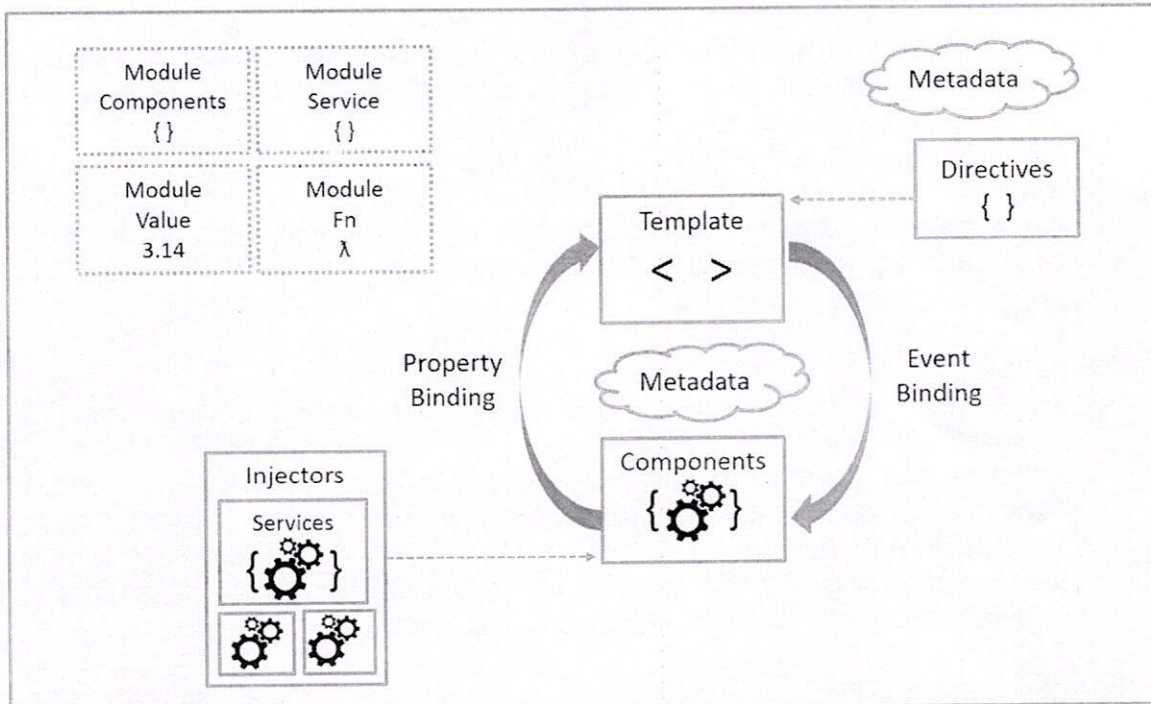
- **Templates:** Quickly create UI views with simple and powerful template syntax.
- **Angular CLI:** Command line tools: Can easily and quickly start building components, adding components, testing them, and then, instantly deploy them using Angular CLI.
- **IDEs:** Get intelligent code completion, instant errors, and other feedback in popular editors and IDEs like Microsoft's VS Code.

Full Development Story

- **Testing:** With Karma for unit tests, you can identify your mistake on the fly and Protractor makes your scenario tests run faster and in a stable manner.

Building Blocks of Angular

- The main building blocks of Angular are:
- Modules
- Components
- Templates
- Metadata
- Data binding
- Directives
- Services
- Dependency injection



AngularJS Architecture

Modules

Angular apps are modular and to maintain modularity, there exists *Angular modules* or *NgModules*. Every Angular app contains at least one Angular module, i.e. the root module. Generally, it is named as *AppModule*. The *root module* can be the only module in a small application. Most of the apps have multiple modules. A module is a cohesive block of code with a related set of capabilities which have a specific application domain or a workflow. Any angular module is a class with `@NgModule` decorator.

Decorators are functions that modify JavaScript classes. Decorators are basically used for attaching metadata to classes so that, it knows the configuration of those classes and how they should work. *NgModule* is a decorator function that takes metadata object whose properties describe the module. The properties are:

- **declarations:** The classes that are related to views and it belong to this module. There are three classes of Angular that can contain view: components, directives and pipes. We will talk about them in a while.
- **exports:** The classes that should be accessible to the components of other modules.
- **imports:** Modules whose classes are needed by the component of this module.
- **providers:** Services present in one of the modules which is to be used in the other modules or components. Once a service is included in the providers it becomes accessible in all parts of that application
- **bootstrap:** The *root component* which is the main view of the application. This root module only has this property and it indicates the component that is to be bootstrapped.

Template of root module (i.e. `src/app/app.module.ts`):

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
```

```

@NgModule({
  imports:[ BrowserModule ],
  providers: [ BookList ],
  declarations: [ AppComponent ],
  exports: [],
  bootstrap: [ AppComponent ]
})
export class AppModule { }

```

A root module generally doesn't *export* its class because as root module is the one which imports other modules & components to use them. The *AppModule* is *bootstrapped* in a *main.ts* file, where the bootstrap module is specified and inside the bootstrap module, contains the bootstrap component.

```

import { enableProdMode } from '@angular/core';
import { platformBrowserDynamic } from '@angular/platform-browser-dynamic';
import { AppModule } from './app/app.module';
import { environment } from './environments/environment';
if (environment.production) {
  enableProdMode();
}
platformBrowserDynamic().bootstrapModule(AppModule);

```

Angular libraries

Angular gives a collection of JavaScript modules (library modules) which provide various functionalities.

Each Angular library has @angular prefix, like

- @angular/core,
- @angular/compiler,
- @angular/compiler-cli,
- @angular/http,
- @angular/router.

using the npm package manager the libraries can be installed and import parts of them with JavaScript import statements.

Example:

```
import { Component } from '@angular/core';
```

Components

A *component* controls one or more section on the screen called a *view*. For example, for a movie list application, you can have components like App Component (*the bootstrapped component*), Movielist Component, Movie Description Component, etc. Inside the component, component's application logic is defined i.e. how does it support the view—inside a class.

The class interacts with the view through an API of properties and methods.

Every app has a main component which is bootstrapped inside the main module, i.e AppComponent.

```
import { Component } from '@angular/core';
@Component({
```

```

    selector:'app-root',
    templateUrl:'./app.component.html',
    styleUrls: ['./app.component.css']
  })
  export class AppComponent{
    title = 'app works!';
  }

```

Templates

A template is a form of HTML tags that tells Angular about how to render the component.

A template looks like regular HTML, except for a few differences.

Example

```

<app-navbar></app-navbar>
<div class ="container">
  <flash-messages></flash-messages>
  <router-outlet></router-outlet>
</div>

```

Metadata

Metadata tells Angular how to process a class.

To inform Angular that MovieList Component is a component, metadata is attached to the class.

In TypeScript, attach metadata by using a decorator.

```

import { Component, OnInit } from '@angular/core';
@Component({
  selector: 'app-movies',
  templateUrl: './movies.component.html',
  styleUrls: ['./movies.component.css']
})

```

Here is the @Component decorator, which identifies the class immediately below it as a component class. The @Component decorator takes the required configuration object which Angular needs to create and present the component and its view.

The most important configurations of @Component decorator are:

- *selector*: Selector tells Angular to create and insert an instance of this component where it finds <app-movies> tag. For example, if an app's HTML contains <app-movies></app-movies>, then Angular inserts an instance of the MovieListComponent view between those tags.
- *templateUrl*: It contains the path of this component's HTML template.
- *providers*: An array of **dependency injection providers** for services that the component requires. This is one way to tell Angular that the component's constructor requires a *MovieService* to get the list of movies to display.

The metadata in the @Component tells Angular where to get the major building blocks to be specified for the component. *The template, metadata, and component together describe a view.*

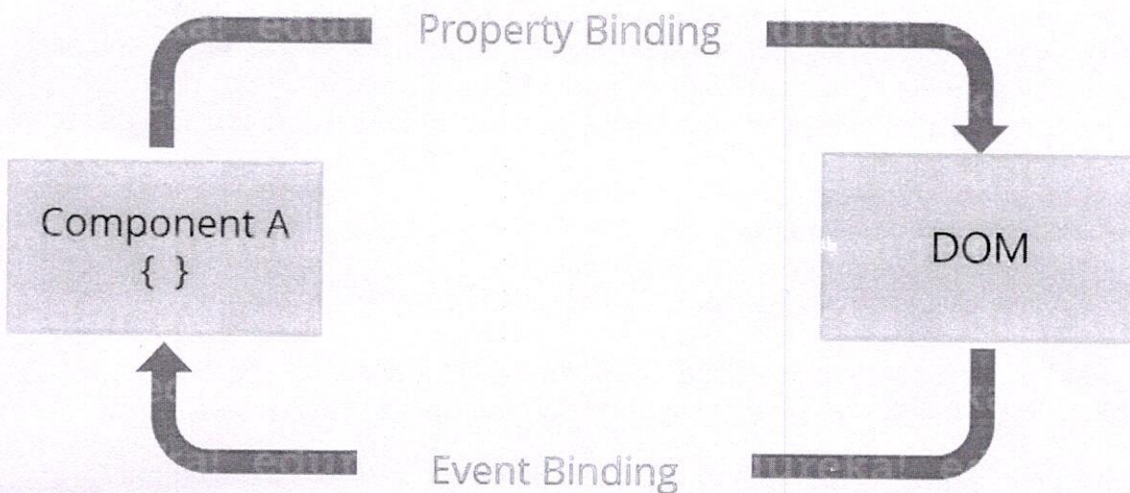
Data Binding

If framework is not used, data values are to be pushed into the HTML controls and turn user responses into some actions and value updates.

Writing such push/pull logic is tedious, error-prone, and a nightmare to read. Angular supports data binding, a mechanism for coordinating parts of a template with parts of a component.

Add binding markup to the template HTML and inform Angular how to connect both sides.

Each form has a direction — to the DOM, from the DOM, or in both directions.



```
<li> {{movie.name}}</li>
<movie-detail [movie]="selectedMovie"></movie-detail>
<li (click)="selectMovie(Movie)"></li>
```

- The `{{movie.name}}` *interpolation* displays the component's name property value within the `` element.
- The `[movie]` *property binding* passes the value of `selectedMovie` from the parent `MovieListComponent` to the `movie` property of the child `MovieDetailComponent`.
- The `(click)` *event binding* calls the component's `selectMovie` method when the user clicks a movie's name.

Two-way data binding is an important part as it combines property and event binding in a single notation, using the `ngModel` directive.

```
<input [(ngModel)]="movie.name">
```

In two-way binding, a data property value flows to the input box from the component as with property binding. The user's changes also flow back to the component, resetting the property to the latest value, as with event binding. Angular processes all data bindings once per JavaScript event cycle, from the root of the application component tree through all child components.

Data binding plays an important role in communication between a template and its component. Data binding is also important for communication between parent and child components.

Directives

- Angular templates are dynamic. When Angular renders them, it transforms the DOM according to the instructions given by directives.
- A directive is a class with a `@Directive` decorator. A component is a directive-with-a-template; a `@Component` decorator is actually a `@Directive` decorator extended with template-oriented features.
- While a component is technically a directive, components are so distinctive and central to Angular applications that this architectural overview separates components from directives.
- Two other kinds of directives exist: structural and attribute directives.
- Directive tends to appear within an element tag as attributes do, sometimes by name but more often as the target of an assignment or a binding.
- **Structural** directives alter layout by adding, removing, and replacing elements in DOM.

Two built-in structural Directives

```
<li *ngFor="let movie of movies"></li>  
<movie-detail *ngIf="selectedMovie"></movie-detail>
```

*ngFor tells Angular to retrieve one `` per movie in the movies

*ngIf includes the MovieDetail component only if a selected movie exists.

Attribute directives alter the appearance or behavior of an existing element.

In templates, attributes are like regular HTML attributes.

The `ngModel` directive, which implements two-way data binding, is an example of an attribute directive.

`ngModel` modifies the behavior of an existing element by setting its display value property and responding to change events.

```
<input [(ngModel)]="movie.name">
```

Angular has a few more directives that either alter the layout structure (for example, `ngSwitch`) or modify aspects of DOM elements and components (for example, `ngStyle` and `ngClass`).

Custom directives can also be created.

Services

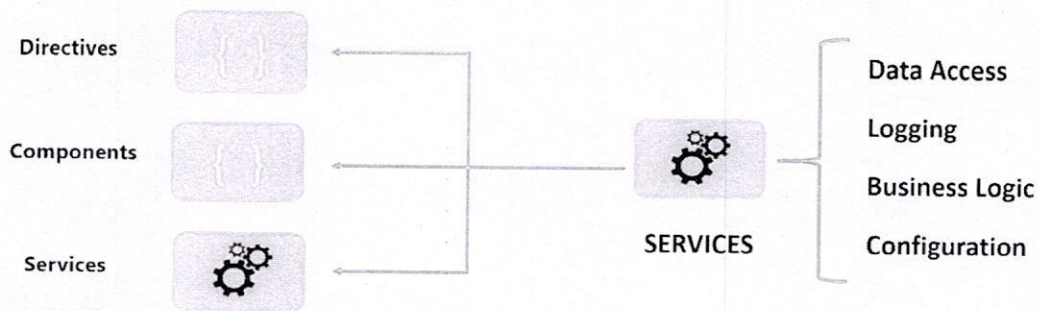
Service is a broad category encompassing any value, function, or feature that the application is in need of.

A service is typically a class with a well-defined purpose.

Anything can be a service.

Examples include:

- logging service
- data service
- message bus
- tax calculator
- application configuration



Angular has no definition of a service.
 There is no service base class, and no place to register a service.
 Yet services are fundamental to any Angular application.
 Components are the consumers of services.

Environmental Setup

Nodejs
 NPM (Will be installed along with NodeJs)

To install angularjs
`npm install -g @angular/cli`

To check the installation versions run the following in command line mode.

```
node -v
npm -v
ng -v
```

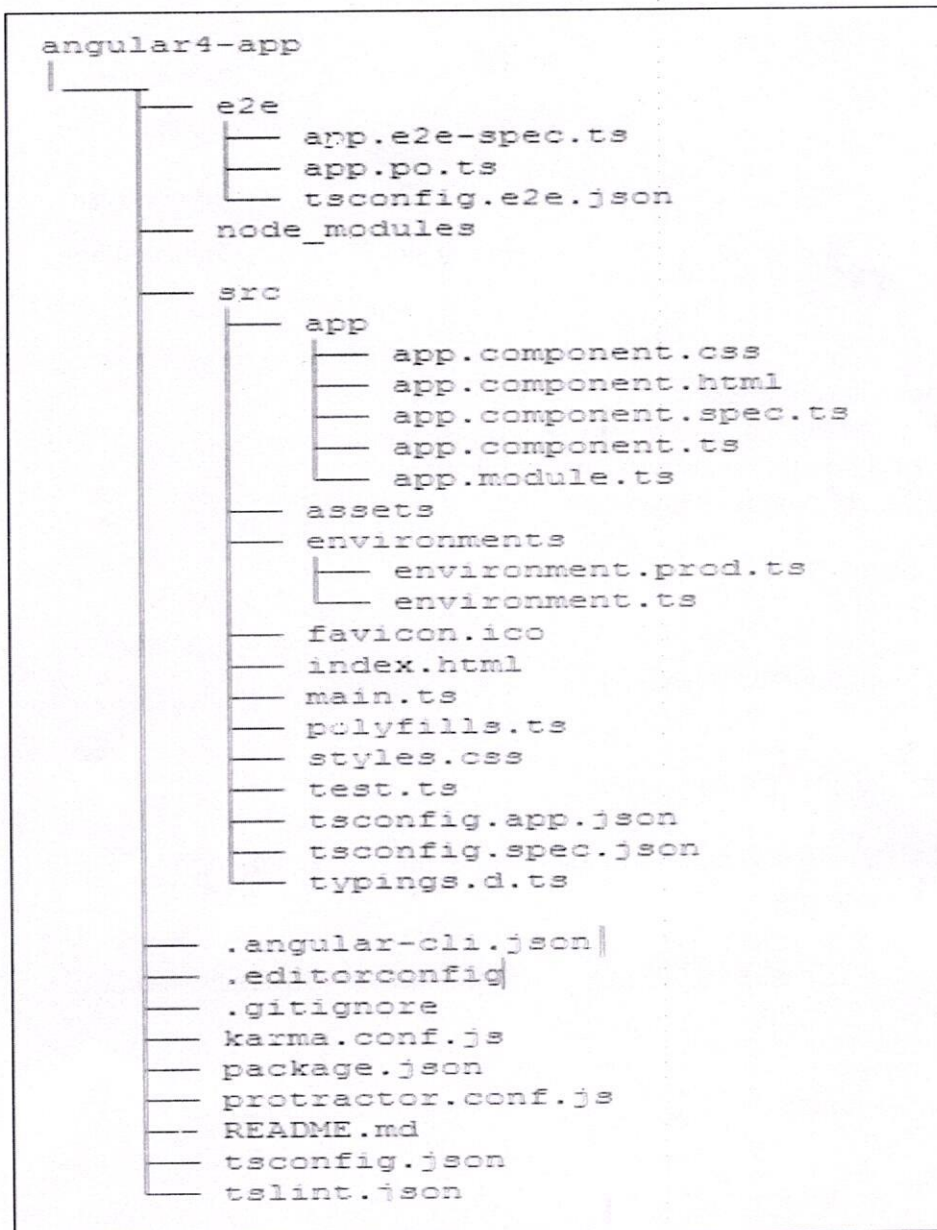
To create a new project:
`ng new angular <appName>`

To start the server
`ng serve`

By default the web servers gets started in the port 4200

To test, launch the browser:
`localhost:4200/`

The project directory structure of an angularjs is as follows:



The Angular 4 app folder has the following **folder structure** –

- **e2e** – end to end test folder. Mainly e2e is used for integration testing and helps ensure the application works fine.
- **node_modules** – The npm package installed is node_modules. You can open the folder and see the packages available.
- **src** – This folder is where we will work on the project using Angular 4.

The Angular 4 app folder has the following **file structure** –

- **.angular-cli.json** – It basically holds the project name, version of cli, etc.
- **.editorconfig** – This is the config file for the editor.
- **.gitignore** – A .gitignore file should be committed into the repository, in order to share the ignore rules with any other users that clone the repository.
- **karma.conf.js** – This is used for unit testing via the protractor. All the information required for the project is provided in karma.conf.js file.
- **package.json** – The package.json file tells which libraries will be installed into node_modules when you run npm install.

- **protractor.conf.js** – This is the testing configuration required for the application.
- **tsconfig.json** – This basically contains the compiler options required during compilation.
- **tslint.json** – This is the config file with rules to be considered while compiling.

The **src folder** is the main folder, which **internally has a different file structure**.
app

It contains the files described below. These files are installed by angular-cli by default.

- **app.module.ts** – If you open the file, you will see that the code has reference to different libraries, which are imported. Angular-cli has used these default libraries for the import – angular/core, platform-browser. The names itself explain the usage of the libraries.

They are imported and saved into variables such as **declarations, imports, providers, and bootstrap**.

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { AppComponent } from './app.component';
```

```
@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
```

```
export class AppModule { }
```

declarations – In declarations, the reference to the components is stored. The AppComponent is the default component that is created whenever a new project is initiated. We will learn about creating new components in a different section.

imports – This will have the modules imported as shown above. At present, BrowserModule is part of the imports which is imported from @angular/platform-browser.

providers – This will have reference to the services created. The service will be discussed in a subsequent chapter.

bootstrap – This has reference to the default component created, i.e., AppComponent.

- **app.component.css** – You can write your css structure over here. Right now, we have added the background color to the div as shown below.

```
.divdetails{
  background-color: #ccc;
}
```

- **app.component.html** – The html code will be available in this file.

```
<!--The content below is only a placeholder and can be replaced.-->
<div class = "divdetails">
```

```

<div style = "text-align:center">
  <h1>
    Welcome to {{title}}!
  </h1>
  <img width = "300" src =
"data:image/svg+xml;base64,PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNv...">
</div>
<h2>Here are some links to help you start: </h2>
<ul>
  <li>
    <h2>
      <a target = "_blank" href="https://angular.io/tutorial">Tour of Heroes</a>
    </h2>
  </li>
  <li>
    <h2>
      <a target = "_blank" href = "https://github.com/angular/angular-cli/wiki">
        CLI Documentation
      </a>
    </h2>
  </li>
  <li>
    <h2>
      <a target="_blank" href="http://angularjs.blogspot.ca/">Angular blog</a>
    </h2>
  </li>
</ul>
</div>

```

This is the default html code currently available with the project creation.

- **app.component.spec.ts** – These are automatically generated files which contain unit tests for source component.
- **app.component.ts** – The class for the component is defined over here. You can do the processing of the html structure in the .ts file. The processing will include activities such as connecting to the database, interacting with other components, routing, services, etc.

The structure of the file is as follows –

```
import { Component } from '@angular/core';
```

```

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'app';
}

```

Assets

You can save your images, js files in this folder.

Chapter 1: Getting started with MongoDB

Remarks

- Data in the world started to grow tremendously after mobile application came in the market. This huge amount of data became almost impossible to handle with traditional relational database - SQL. NoSQL databases are introduced to handle those data where much more flexibility came like variable number of columns for each data.
- MongoDB is one of the leading NoSQL databases. Each collection contains a number of JSON documents. Any data model that can be expressed in a JSON document can be easily stored in MongoDB.
- MongoDB is a server-client database. Server usually runs with the binary file `mongod` and client runs with `mongo`.
- There is no join operation in MongoDB prior to v.3.2, for various philosophical and pragmatic reasons. But Mongo shell supports javascript, so if `$lookup` is not available, one can simulate join operations on documents in javascript before inserting.
- To run an instance in production environment, it's strongly advised to follow the [Operations Checklist](#).

Versions

Version	Release Date
3.4	2016-11-29
3.2	2015-12-08
3.0	2015-03-03
2.6	2014-04-08
2.4	2013-03-19
2.2	2012-08-29
2.0	2011-09-12
1.8	2011-03-16
1.6	2010-08-31
1.4	2010-03-25
1.2	2009-12-10

Examples

Installation

To install MongoDB, follow the steps below:

- **For Mac OS:**

- There are two options for Mac OS: manual install or [homebrew](#).
- **Installing with homebrew:**
 - Type the following command into the terminal:

```
$ brew install mongodb
```

- **Installing manually:**

- Download the latest release [here](#). Make sure that you are downloading the appropriate file, specially check whether your operating system type is 32-bit or 64-bit. The downloaded file is in format `tgz`.
- Go to the directory where this file is downloaded. Then type the following command:

```
$ tar xvf mongodb-osx-xyz.tgz
```

Instead of `xyz`, there would be some version and system type information. The extracted folder would be same name as the `tgz` file. Inside the folder, there would be a subfolder named `bin` which would contain several binary file along with `mongod` and `mongo`.

- By default server keeps data in folder `/data/db`. So, we have to create that directory and then run the server having the following commands:

```
$ sudo bash
# mkdir -p /data/db
# chmod 777 /data
# chmod 777 /data/db
# exit
```

- To start the server, the following command should be given from the current location:

```
$ ./mongod
```

It would start the server on port 27017 by default.

- To start the client, a new terminal should be opened having the same directory as before. Then the following command would start the client and connect to the

server.

```
$ ./mongo
```

By default it connects to the `test` database. If you see the line like `connecting to: test`. Then you have successfully installed MongoDB. Congrats! Now, you can test `Hello World` to be more confident.

- **For Windows:**

- Download the latest release [here](#). Make sure that you are downloading the appropriate file, specially check whether your operating system type is 32-bit or 64-bit.
- The downloaded binary file has extension `exe`. Run it. It will prompt an installation wizard.
- Click **Next**.
- **Accept** the licence agreement and click **Next**.
- Select **Complete** Installation.
- Click on **Install**. It might prompt a window for asking administrator's permission. Click **Yes**.
- After installation click on **Finish**.
- Now, the `mongodb` is installed on the path `C:/Program Files/MongoDB/Server/3.2/bin`. Instead of version 3.2, there could be some other version for your case. The path name would be changed accordingly.
- `bin` directory contain several binary file along with `mongod` and `mongo`. To run it from other folder, you could add the path in system path. To do it:
 - Right click on **My Computer** and select **Properties**.
 - Click on **Advanced system setting** on the left pane.
 - Click on **Environment Variables...** under the **Advanced** tab.
 - Select **Path** from **System variables** section and click on **Edit...**
 - Before Windows 10, append a semi-colon and paste the path given above. From Windows 10, there is a **New** button to add new path.
 - Click **OKs** to save changes.
- Now, create a folder named `data` having a sub-folder named `db` where you want to run the server.
- Start command prompt from their. Either changing the path in cmd or clicking on **Open command window here** which would be visible after right clicking on the empty space of the folder GUI pressing the Shift and Ctrl key together.
- Write the command to start the server:

```
> mongod
```

It would start the server on port 27017 by default.

- Open another command prompt and type the following to start client:

```
> mongo
```

- By default it connects to the `test` database. If you see the line like `connecting to: test.` Then you have successfully installed MongoDB. Congrats! Now, you can test `Hello World` to be more confident.

- **For Linux:** Almost same as Mac OS except some equivalent command is needed.

- For Debian-based distros (using `apt-get`):

- Import MongoDB Repository key.

```
$ sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv EA312927
gpg: Total number processed: 1\
gpg:             imported: 1 (RSA: 1)
```

- Add repository to package list on **Ubuntu 16.04**.

```
$ echo "deb http://repo.mongodb.org/apt/ubuntu xenial/mongodb-org/3.2
multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-3.2.list
```

- on **Ubuntu 14.04**.

```
$ echo "deb http://repo.mongodb.org/apt/ubuntu trusty/mongodb-org/3.2
multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-3.2.list
```

- Update package list.

```
$ sudo apt-get update
```

- Install MongoDB.

```
$ sudo apt-get install mongodb-org
```

- For Red Hat based distros (using `yum`):

- use a text editor which you prefer.

```
$ vi /etc/yum.repos.d/mongodb-org-3.4.repo
```

- Paste following text.

```
[mongodb-org-3.4]
```

```
name=MongoDB Repository
baseurl=https://repo.mongodb.org/yum/redhat/$releasever/mongodb-
org/3.4/x86_64/
gpgcheck=1
enabled=1
gpgkey=https://www.mongodb.org/static/pgp/server-3.4.asc
```

- Update package list.

```
$ sudo yum update
```

- Install MongoDB

```
$ sudo yum install mongodb-org
```

Hello World

After installation process, the following lines should be entered in mongo shell (client terminal).

```
> db.world.insert({ "speech" : "Hello World!" });
> cur = db.world.find();x=cur.next();print(x["speech"]);
```

Hello World!

Explanation:

- In the first line, we have inserted a { key : value } paired document in the default database `test` and in the collection named `world`.
- In the second line we retrieve the data we have just inserted. The retrieved data is kept in a javascript variable named `cur`. Then by the `next()` function, we retrieved the first and only document and kept it in another js variable named `x`. Then printed the value of the document providing the key.

Complementary Terms

SQL Terms	MongoDB Terms
Database	Database
Table	Collection
Entity / Row	Document
Column	Key / Field
Table Join	Embedded Documents
Primary Key	Primary Key (Default key <code>_id</code> provided by mongodb itself)

Execution of a JavaScript file in MongoDB

```
./mongo localhost:27017/mydb myjsfile.js
```

Explanation: This operation executes the `myjsfile.js` script in a `mongo` shell that connects to the `mydb` database on the `mongod` instance accessible via the `localhost` interface on port `27017`. `localhost:27017` is not mandatory as this is the default port `mongodb` uses.

Also, you can run a `.js` file from within `mongo` console.

```
>load("myjsfile.js")
```

Making the output of find readable in shell

We add three records to our collection test as:

```
> db.test.insert({"key":"value1","key2":"Val2","key3":"val3"})
WriteResult({ "nInserted" : 1 })
> db.test.insert({"key":"value2","key2":"Val21","key3":"val31"})
WriteResult({ "nInserted" : 1 })
> db.test.insert({"key":"value3","key2":"Val22","key3":"val33"})
WriteResult({ "nInserted" : 1 })
```

If we see them via `find`, they will look very ugly.

```
> db.test.find()
{ "_id" : ObjectId("5790c5cecae25b3d38c3c7ae"), "key" : "value1", "key2" : "Val2", "key3" : "val3" }
{ "_id" : ObjectId("5790c5d9cae25b3d38c3c7af"), "key" : "value2", "key2" : "Val21", "key3" : "val31" }
{ "_id" : ObjectId("5790c5e9cae25b3d38c3c7b0"), "key" : "value3", "key2" : "Val22", "key3" : "val33" }
```

To work around this and make them readable, use the `pretty()` function.

```
> db.test.find().pretty()
{
  "_id" : ObjectId("5790c5cecae25b3d38c3c7ae"),
  "key" : "value1",
  "key2" : "Val2",
  "key3" : "val3"
}
{
  "_id" : ObjectId("5790c5d9cae25b3d38c3c7af"),
  "key" : "value2",
  "key2" : "Val21",
  "key3" : "val31"
}
{
  "_id" : ObjectId("5790c5e9cae25b3d38c3c7b0"),
  "key" : "value3",
  "key2" : "Val22",
  "key3" : "val33"
}
```

```
}  
>
```

Basic commands on mongo shell

Show all available databases:

```
show dbs;
```

Select a particular database to access, e.g. `mydb`. This will create `mydb` if it does not already exist:

```
use mydb;
```

Show all collections in the database (be sure to select one first, see above):

```
show collections;
```

Show all functions that can be used with the database:

```
db.mydb.help();
```

To check your currently selected database, use the command `db`

```
> db  
mydb
```

`db.dropDatabase()` command is used to drop a existing database.

```
db.dropDatabase()
```

Read Getting started with MongoDB online: <https://riptutorial.com/mongodb/topic/691/getting-started-with-mongodb>

Chapter 2: 2dsphere Index

Examples

Create a 2dsphere Index

`db.collection.createIndex()` method is used to create a `2dsphere` index. The blueprint of a `2dsphere` index :

```
db.collection.createIndex( { <location field> : "2dsphere" } )
```

Here, the `location field` is the key and `2dsphere` is the type of the index. In the following example we are going to create a `2dsphere` index in the `places` collection.

```
db.places.insert(
{
  loc : { type: "Point", coordinates: [ -73.97, 40.77 ] },
  name: "Central Park",
  category : "Parks"
})
```

The following operation will create `2dsphere` index on the `loc` field of `places` collection.

```
db.places.createIndex( { loc : "2dsphere" } )
```

Read 2dsphere Index online: <https://riptutorial.com/mongodb/topic/6632/2dsphere-index>

Chapter 3: Aggregation

Introduction

Aggregations operations process data records and return computed results. Aggregation operations group values from multiple documents together, and can perform a variety of operations on the grouped data to return a single result. MongoDB provides three ways to perform aggregation: the aggregation pipeline, the map-reduce function, and single purpose aggregation methods.

From Mongo manual <https://docs.mongodb.com/manual/aggregation/>

Syntax

- `db.collection.aggregate(pipeline, options)`

Parameters

Parameter	Details
<code>pipeline</code>	array(A sequence of data aggregation operations or stages)
<code>options</code>	document(optional, available only if pipeline present as an array)

Remarks

Aggregation framework in MongoDB is used to achieve common `GROUP BY` functionality of SQL.

Consider the following insertions in collection named `transactions` for every example.

```
> db.transactions.insert({ cr_dr : "D", amount : 100, fee : 2});
> db.transactions.insert({ cr_dr : "C", amount : 100, fee : 2});
> db.transactions.insert({ cr_dr : "C", amount : 10, fee : 2});
> db.transactions.insert({ cr_dr : "D", amount : 100, fee : 4});
> db.transactions.insert({ cr_dr : "D", amount : 10, fee : 2});
> db.transactions.insert({ cr_dr : "C", amount : 10, fee : 4});
> db.transactions.insert({ cr_dr : "D", amount : 100, fee : 2});
```

Examples

Count

How do you get the number of Debit and Credit transactions? One way to do it is by using `count()` function as below.

```
> db.transactions.count({cr_dr : "D"});
```

or

```
> db.transactions.find({cr_dr : "D"}).length();
```

But what if you do not know the possible values of `cr_dr` upfront. Here Aggregation framework comes to play. See the below Aggregate query.

```
> db.transactions.aggregate([
  {
    $group : {
      _id : '$cr_dr', // group by type of transaction
      // Add 1 for each document to the count for this type of transaction
      count : {$sum : 1}
    }
  }
]);
```

And the result is

```
{
  "_id" : "C",
  "count" : 3
}
{
  "_id" : "D",
  "count" : 5
}
```

Sum

How to get the summation of `amount`? See the below aggregate query.

```
> db.transactions.aggregate([
  {
    $group : {
      _id : '$cr_dr',
      count : {$sum : 1}, //counts the number
      totalAmount : {$sum : '$amount'} //sums the amount
    }
  }
]);
```

And the result is

```
{
  "_id" : "C",
  "count" : 3.0,
  "totalAmount" : 10.0
}
```



```

    "totalAmount" : 120.0
  }
  {
    "_id" : "D",
    "count" : 5.0,
    "totalAmount" : 410.0
  }
}

```

Another version that sums amount and fee.

```

> db.transactions.aggregate(
  [
    {
      $group : {
        _id : '$scr_dr',
        count : {$sum : 1},
        totalAmount : {$sum : { $sum : ['$amount', '$fee']}}
      }
    }
  ]
);

```

And the result is

```

{
  "_id" : "C",
  "count" : 3.0,
  "totalAmount" : 128.0
}
{
  "_id" : "D",
  "count" : 5.0,
  "totalAmount" : 422.0
}

```

Average

How to get the average amount of debit and credit transactions?

```

> db.transactions.aggregate(
  [
    {
      $group : {
        _id : '$scr_dr', // group by type of transaction (debit or credit)
        count : {$sum : 1}, // number of transaction for each type
        totalAmount : {$sum : { $sum : ['$amount', '$fee']}}, // sum
        averageAmount : {$avg : { $sum : ['$amount', '$fee']}} // average
      }
    }
  ]
);

```

The result is

```

{

```

```

    "_id" : "C", // Amounts for credit transactions
    "count" : 3.0,
    "totalAmount" : 128.0,
    "averageAmount" : 40.0
  }
  {
    "_id" : "D", // Amounts for debit transactions
    "count" : 5.0,
    "totalAmount" : 422.0,
    "averageAmount" : 82.0
  }
}

```

Operations with arrays.

When you want to work with the data entries in arrays you first need to **unwind** the array. The unwind operation creates a document for each entry in the array. When you have lot's of documents with large arrays you will see an explosion in number of documents.

```

{ "_id" : 1, "item" : "myItem1", sizes: [ "S", "M", "L" ] }
{ "_id" : 2, "item" : "myItem2", sizes: [ "XS", "M", "XL" ] }

db.inventory.aggregate( [ { $unwind : "$sizes" } ] )

```

An important notice is that when a document doesn't contain the array it will be lost. From mongo 3.2 and up there are is an unwind option "preserveNullAndEmptyArrays" added. This option makes sure the document is preserved when the array is missing.

```

{ "_id" : 1, "item" : "myItem1", sizes: [ "S", "M", "L" ] }
{ "_id" : 2, "item" : "myItem2", sizes: [ "XS", "M", "XL" ] }
{ "_id" : 3, "item" : "myItem3" }

db.inventory.aggregate( [ { $unwind : { path: "$sizes", includeArrayIndex: "arrayIndex" } } ] )

```

Match

How to write a query to get all departments where average age of employees making less than or \$70000 is greather than or equal to 35?

In order to that we need to write a query to match employees that have a salary that is less than or equal to \$70000. Then add the aggregate stage to group the employees by the department. Then add an accumulator with a field named e.g. average_age to find the average age per department using the \$avg accumulator and below the existing \$match and \$group aggregates add another \$match aggregate so that we're only retrieving results with an average_age that is greather than or equal to 35.

```

db.employees.aggregate([
  {"$match": {"salary": {"$lte": 70000}}},
  {"$group": {"_id": "$dept",
    "average_age": {"$avg": "$age"}
  }
},
  {"$match": {"average_age": {"$gte": 35}}}
]

```