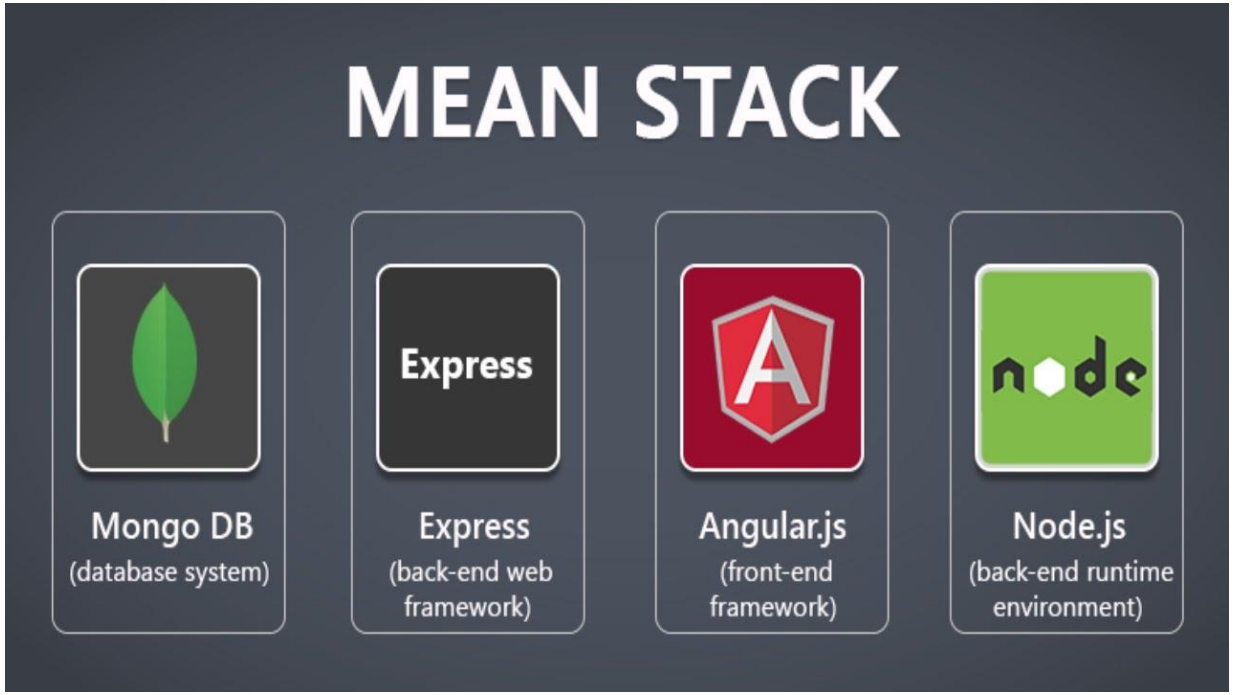

Overview

ANURADHA SENGALVARAYAN

Objectives

01

Overview of Web application

02

Introduction to MEAN stack

03

Describe MEAN hotel project

Web application overview

- Web application is nothing but client-sever computer program that has the following components
  - Client side program that lives on the browser and responds to user inputs
  - Server side program that lives on the Server and responds to HTTP requests
  - Database that persists application data and returns the data on request

Client side programs

- Parsed by user's web browser
- Responsibilities
  - ✓ Present information sent by Sever in appropriate format/layout
  - ✓ React to user interactions
  - ✓ Validate data input by users and send HTTP request to the server
- Uses HTML, CSS and Javascript

Server side programs

Responsibilities

- Listen to HTTP requests and invoke appropriate sections of the code (Routing)
- Retrieve data from database and return to client
- Process data received from client and store in database
- Authenticate users credentials

Languages/Frameworks

- PHP
- Python
- Ruby
- C#
- Javascript (Node.js)

Databases

- Specialized application to store and retrieve data in disks
- Can store structured and unstructured data
- Two types
  - Relational database that store structured data in the form of tables and rows
  - Document oriented NOSQL databases that can store structured and unstructured data
- Some databases
  - Relational
    - Oracle
    - SQL Server
    - Teradata
  - Non relational
    - MongoDB
    - IBM Domino
    - Hbase
    - Cassandra

MEAN stack - overview

- MEAN is a free and open-source JavaScript software stack for building dynamic web applications
- The components of the MEAN stack are as follows
  - ✓ MongoDB, a NoSQL database
  - ✓ Express.js, a web application framework that runs on Node.js
  - ✓ Angular.js or Angular, JavaScript MVC frameworks that run in browser
  - ✓ Node.js, an execution environment for event-driven server-side applications

Node.js

- Node.js is an open-source, cross-platform JavaScript run-time environment for executing JavaScript code on the server-side
- Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient
- Node.js' package ecosystem, NPM, is the largest ecosystem of open source libraries in the world

Express.js

- Express.js is an open-source web application framework for Node.js, designed for building web applications and APIs
- Express provides various features that make web application development fast and easy which otherwise takes more time using only Node.js

Angular

- AngularJS is a JavaScript-based open-source front-end web application framework
- Simplifies building Single Page Applications by extending HTML attributes with directives and binding data to HTML with expressions
- Provides framework for developing client side applications using MVC architecture
- Event handling, DOM manipulation, Form validation and AJAX calls can be made much more easily with Angular

MongoDB

- MongoDB is an open-source cross-platform document-oriented database program
- Classified as a NoSQL database program, MongoDB uses JSON-like documents and provides high performance, high availability, and easy scalability
- Some terminology
  - Database is container for collections. Each database gets its own set of files on the file system
  - Collection is a group of MongoDB documents. It is the equivalent of an RDBMS table
  - A document is a set of key-value pairs and have dynamic schema, meaning documents in the same collection do not need to have the same set of fields or structure

MEAN Hotel Project - overview

- Single Page Application that has the following features
  - ✓ Register new users
  - ✓ Authenticate existing user credentials
  - ✓ Display list of Hotels
  - ✓ Display details of selected Hotel
  - ✓ Allow users to add and modify reviews for any Hotel
- Following slides describe some key components

app.js

Server side component

Main program that starts the server and is responsible for the following

1) Invokes db.js which uses Mongoose module to establish connection to MongoDB
2) Listens for any HTTP requests on specified port
3) Invokes appropriate routes by using index.js
4) Serves client side objects – HTML, CSS, Javascripts and Images stored under public folder
5) Contains middleware like bodyParser

index.js

Server side component

Handles all routing on the server side

ROUTE	METHOD	INVOKE
/hotels	GET	ctrlUsers.authenticate ctrlHotels.hotelsGetAll
	POST	ctrlHotels.hotelsAddOne
/hotels/:hotelId	GET	ctrlHotels.hotelsGetOne
	PUT	ctrlHotels.hotelsUpdateOne
/hotels/:hotelId/reviews	GET	ctrlReviews.reviewsGetAll
	POST	ctrlUsers.authenticate ctrlReviews.reviewsAddOne
/hotels/:hotelId/reviews/:reviewId	GET	ctrlReviews.reviewsGetOne
	PUT	ctrlReviews.reviewsUpdateOne
/users/register	POST	ctrlUsers.register
/users/login	POST	ctrlUsers.login

controllers

Server side component

Controllers do the actual work of retrieving data from the request, interacting with the database to retrieve and store data using Mongoose and sending information back to the client via response

Following controllers are developed

hotels.controllers.js

reviews.controllers.js

users.controllers.js

DB modules

Server side component

Separate modules are created that handle connection to MongoDB and also to define the Models for the collections used in the app

- db.js connects to the database on server start up and closes connection when the app is shut down
- hotels.models.js defines the data model for the hotels collection
- users.model.js defines the data model for users collection

Client side components

index.html : Shell page that gets loaded initially and contains angular directives and links to the angular app “meanhotel”

app.js: Main angular file where module is defined and routes are set up

controllers: separate controllers are defined for each route, which use corresponding factories corresponding factories to make AJAX calls to the server to retrieve data

Run time control flow - example

Prerequisites:
App.js is running
MongoDB is running

User selects MeanHotel home page in browser

Server serves Index.html, CSS and Javascript files

Browser loads Index.html and javascript files. Since there is reference to angular module, angular handles routing as defined in app.js

Per navigation directive, navigation-directive.html is loaded and upon supplying user/password, authorization interceptor is invoked, which uses auth-factory.js to make AJAX calls to authenticate

User clicks Hotels link in the page

Hotels Controller is invoked, which invokes corresponding Factory to make AJAX call to server and retrieves data

template URL hotels.html is populated with Hotel data by angular and rendered by the browser