ls : command diya akta folder a ja kisu ase sob dekya jabe.

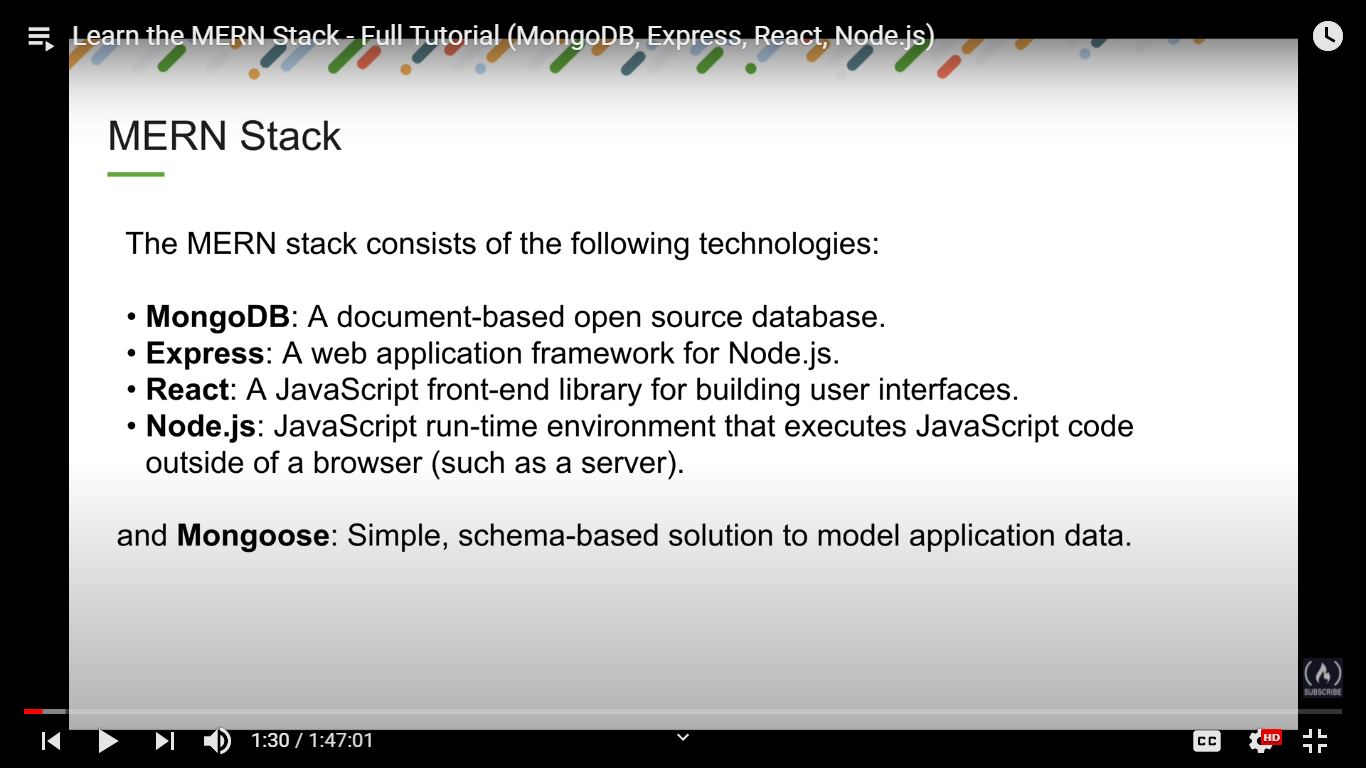
cat package.jon : command diya package.jon file ta dekha jabe.

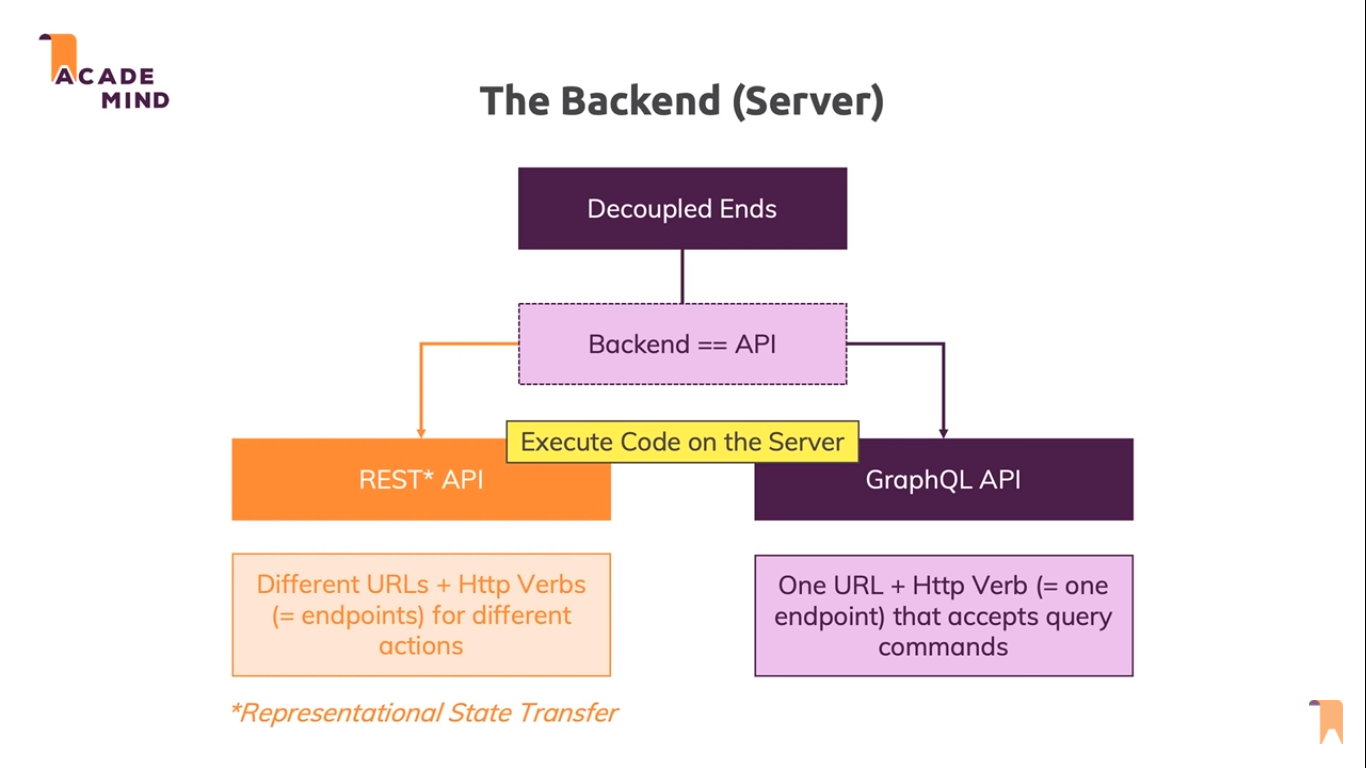
touch app.js : command diya new file create hobe (here app.js is file name).

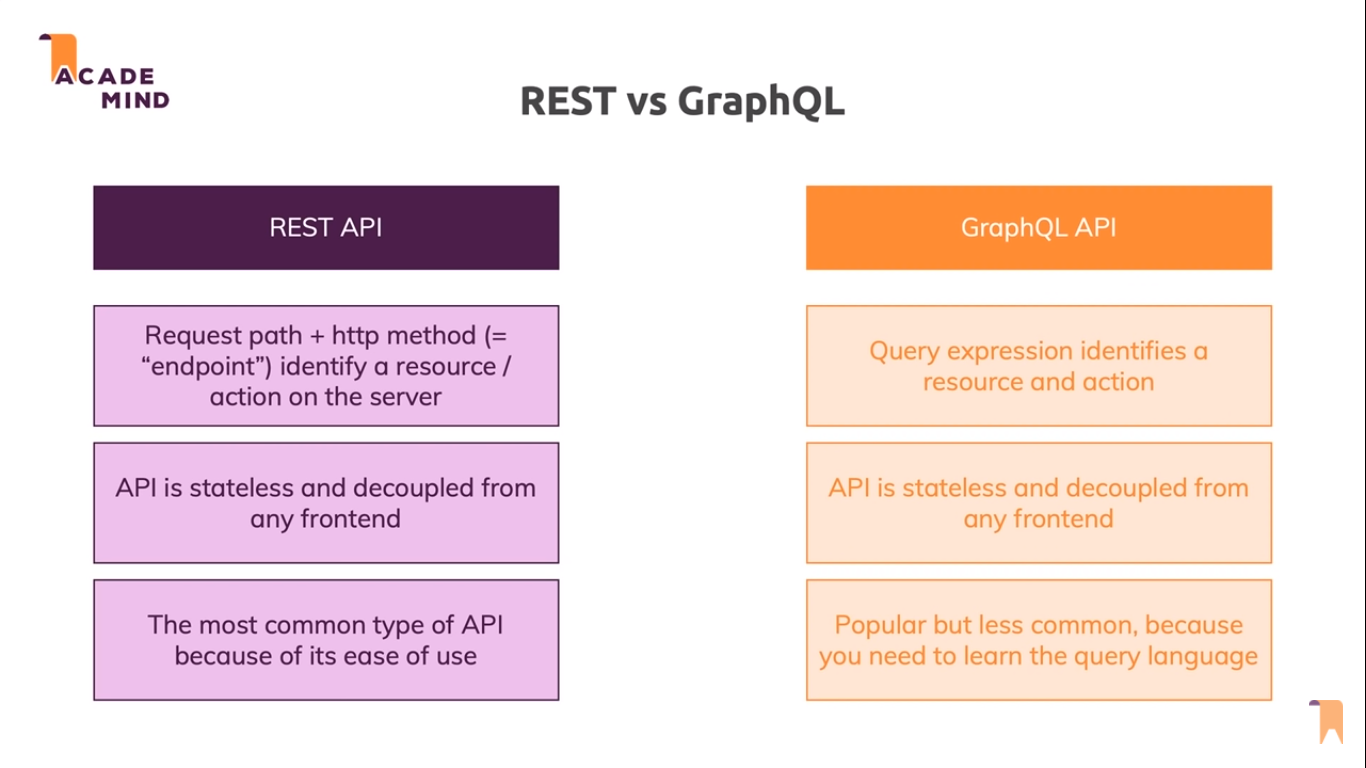
mkdir newFolder: newFolder ai name akti directory/folder create hobe.

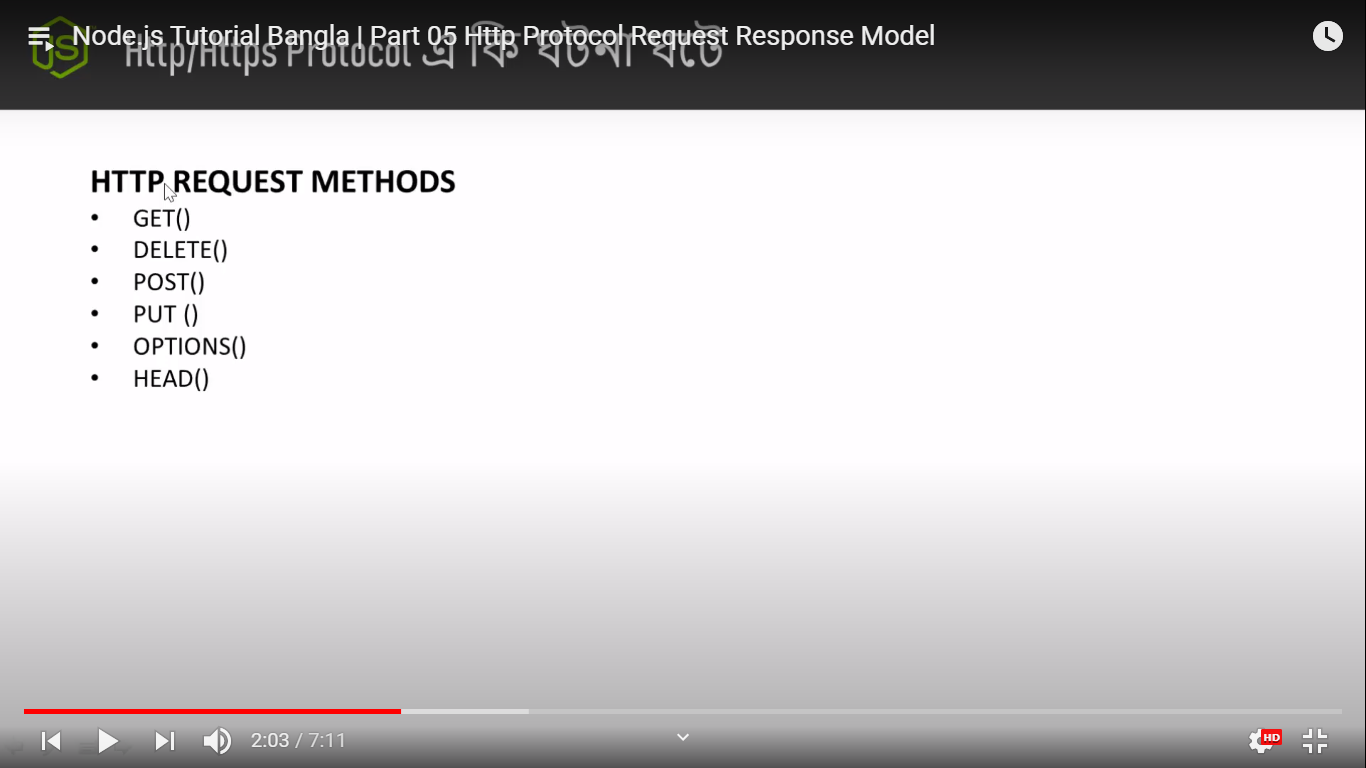
rm app.js : remove app.js.

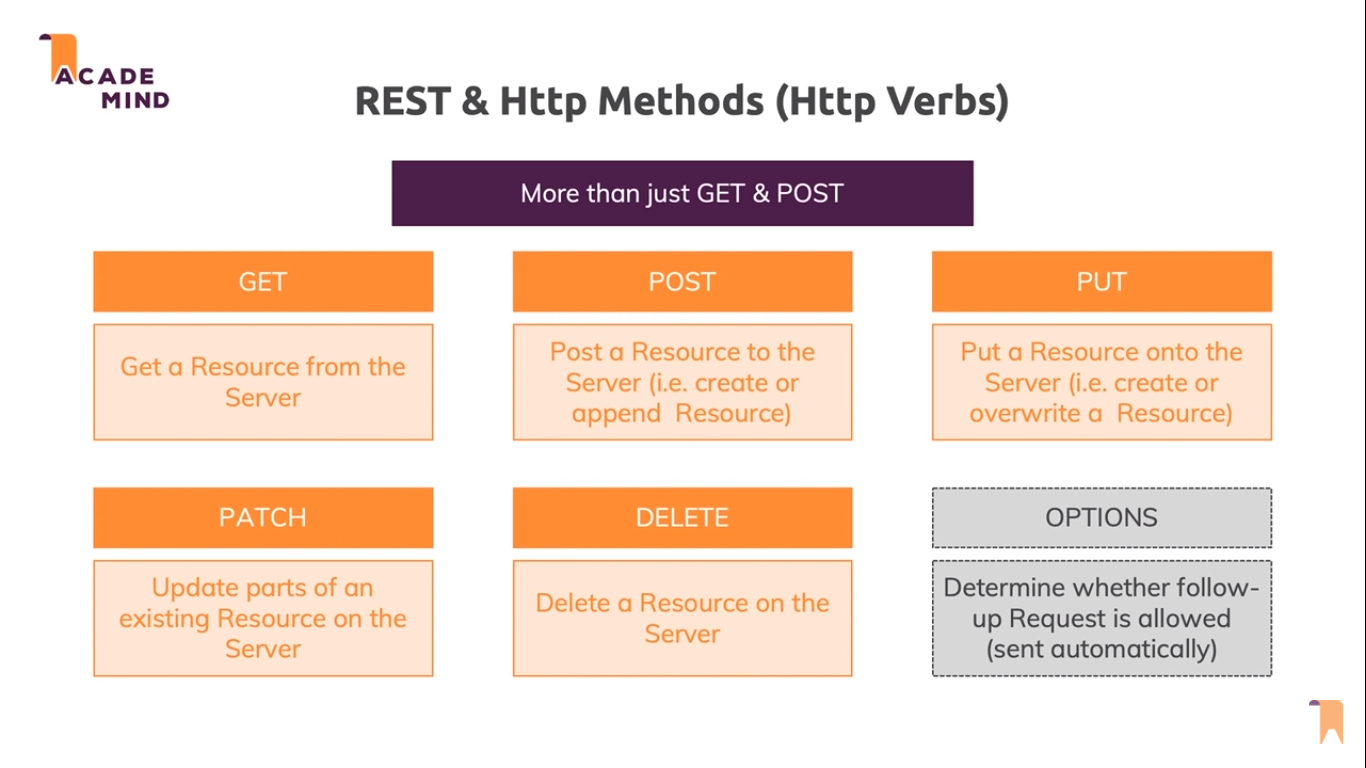
code app.js: app.js will be opened.

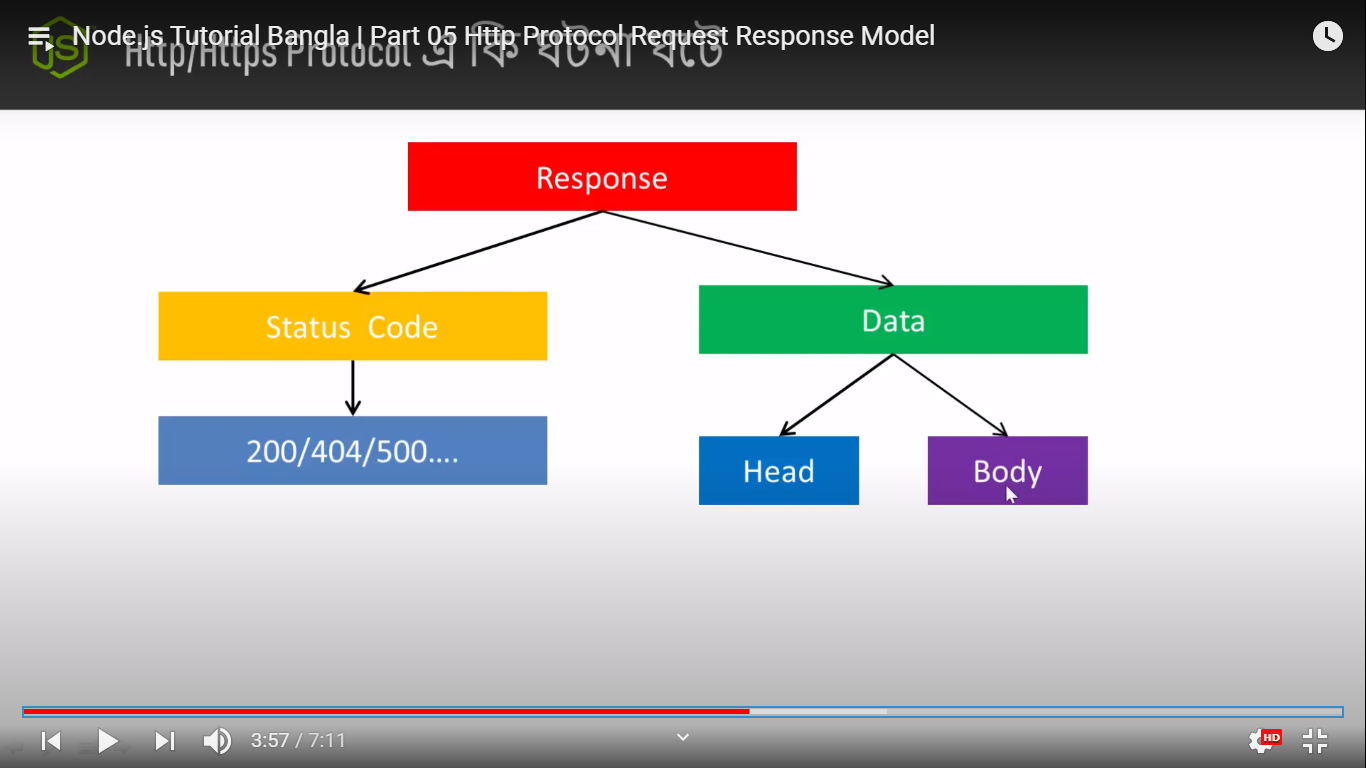


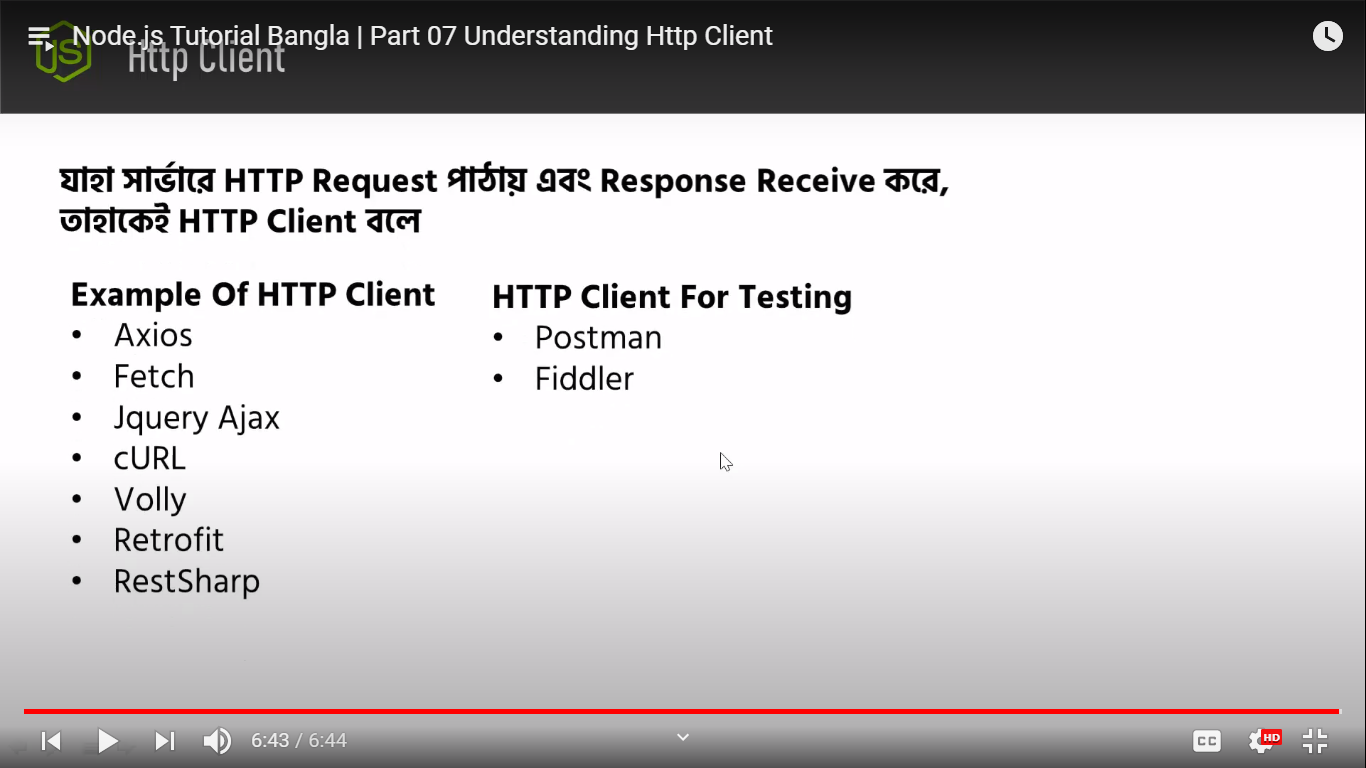


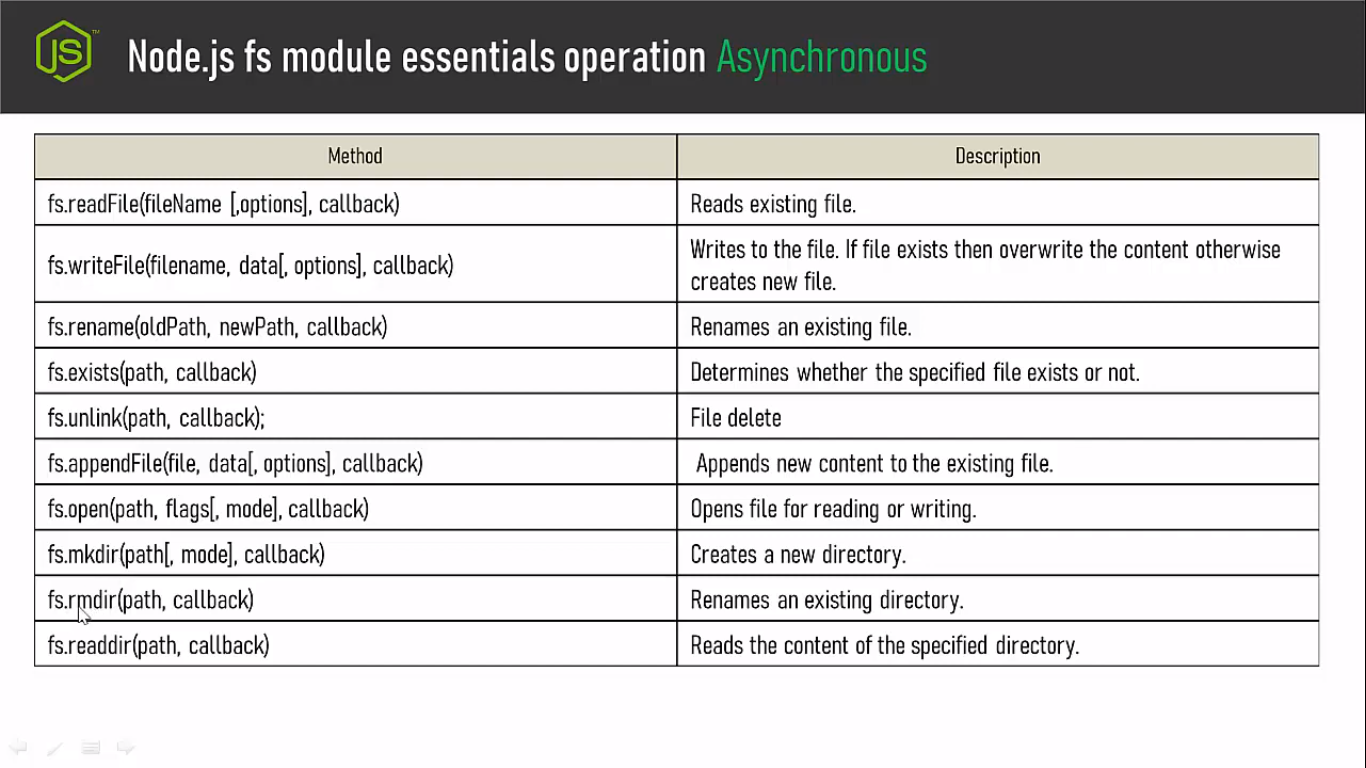


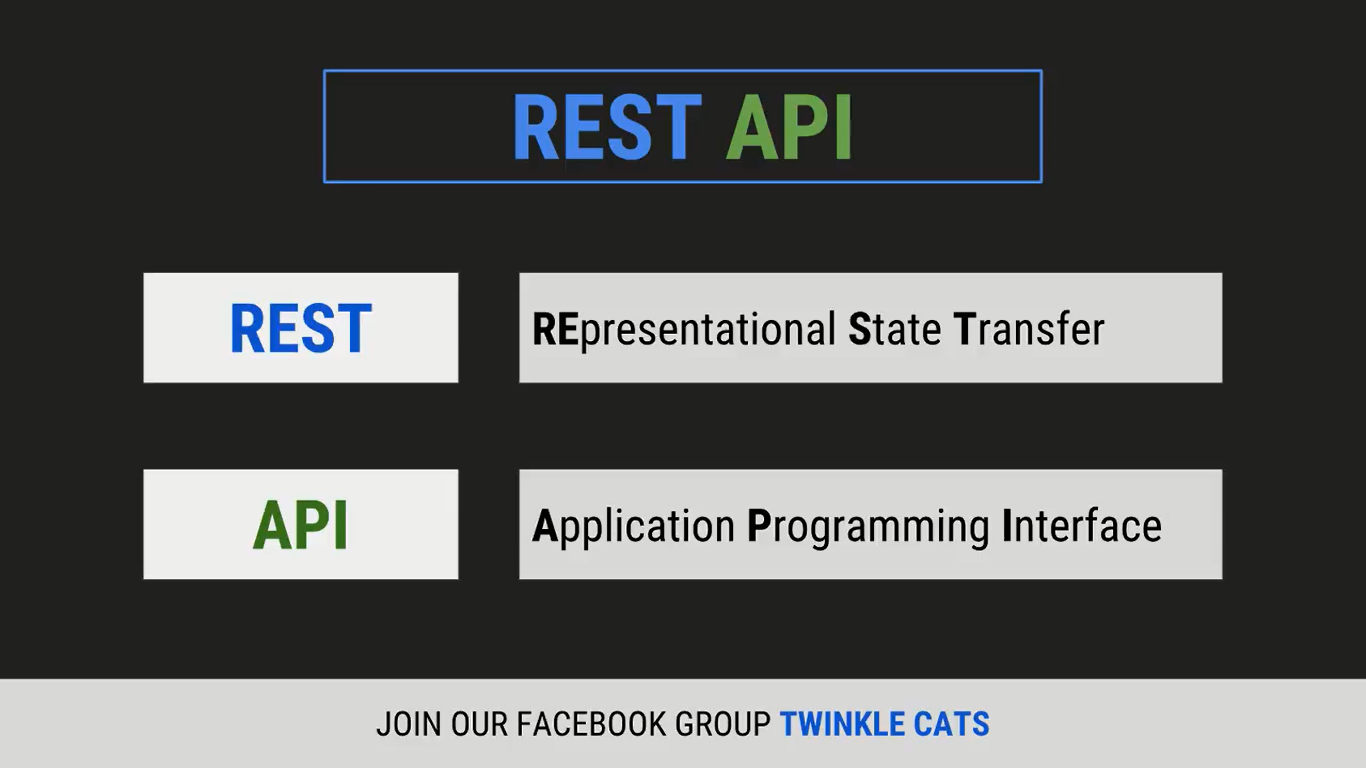












await : The await keyword lets you wait for the promise to resolve. Once the promise is resolved, it returns the parameter passed into the then call.

await blocks JavaScript from executing the next line of code until a promise resolves. This may have the unintended consequence of slowing down code execution.

If you don’t need await, you don’t need to use an async function.

জাভস্ক্রিপ্ট এ এই নতুন async আর await এর পরিচয় করিয়ে দেওয়া হয়েছে। যেখানেই আপনি এরকম অ্যাসিনক্রোনাস কোডকে সিনক্রোনাস আচরণ করাতে চান সেখানেই এগুলো ব্যবহার করতে পারবেন। তবে এখানে কিছু নিয়ম কানুন আছে। আপনাকে async এই কীওয়ার্ডটা ব্যবহার করতে হবে ফাংশনের সাথে। আপনি যে কোডগুলোকে সিনক্রোনাস আচরণ করাতে চাচ্ছেন সেগুলো সবগুলো একটা ফাংশনের ভিতরে ঢুকিয়ে সেই ফাংশনের নামের আগে জাস্ট এভাবে async কীওয়ার্ডটা লাগিয়ে দিবেনঃ

Asynchronous functions always return promises.

জাভাস্ক্রিপ্ট এর অ্যাসিনক্রোনাস আচরণ সম্পর্কে আমরা জানি। রিমোট কোনো সার্ভার থেকে বা একটু সময় লাগে এমন কোনো অপারেশন শেষ করার জন্যে জাভাস্ক্রিপ্ট অপেক্ষা না করে বরং পরের অপারেশনে চলে যায়। এধরনের অপারেশন হচ্ছে অ্যাসিনক্রোনাস অপারেশন। এখন এই অপারেশন ফেলে অন্য অপারেশনে চলে গেলেও জাভাস্ক্রিপ্ট ঠিকই এটাকেও ট্র্যাক করে রাখে। আর এইজন্যেই জাভাস্ক্রিপ্ট প্রমিস(Promise) ব্যবহার করে। এখন প্রমিসের কাজ হচ্ছে এ ধরনের অ্যাসিনক্রোনাস অপারেশনকে হ্যান্ডল করা। আমাদের বেশীর ভাগ ক্ষেত্রে রিটার্ণ হওয়া প্রমিস .then() আর .catch() দিয়েই হ্যান্ডল করতে হয়।

A **promise** is an object that will return a value in future. Because of this “in future” thing, Promises are well suited for asynchronous JavaScript operations.

static files are files that clients download as they are from the server. Create a new directory, **public**. Express, by default does not allow you to serve static files. You need to enable it using the following built-in middleware.

Note that the root route is now set to your public dir, so all static files you load will be considering public as root.

Now, run the above code using node server.js command and point your browser to *http://localhost:5000/myImage.jpg* and it will display myImage.jpg from the public folder (public folder should have myImage.jpg).

A URL (Uniform Resource Locator) is the address of a resource in the world wide web. URL encoding is a mechanism for translating unprintable or special characters to a universally accepted format by web servers and browsers.

URL Encoding converts reserved, unsafe, and non-ASCII characters in URLs to a format that is universally accepted and understood by all web browsers and servers. URL Encoding is the process of converting string into valid URL format.

**bcrypt** is a [password-hashing function](https://en.wikipedia.org/wiki/Password-hashing_function)

The **res.status()** function set the HTTP status for the response. It is a chainable alias of Node’s response.statusCode.

**Syntax:**

res.status( code )

**Parameter:** This function accepts single parameter **code** that holds the HTTP status code.

**Returns:** It retuns an Object.

// equivalent to res.status(200).send('Everything is OK')

/ / equivalent to res.status(400). Send (' Bad Request ')

// equivalent to res.status(403).send('Forbidden')

// equivalent to res.status(404).send('Not Found')

// equivalent to res.status(500).send('Internal Server Error')

http.STATUS\_CODES

{ '100': 'Continue',

'101': 'Switching Protocols',

'102': 'Processing',

'200': 'OK',

'201': 'Created',

'202': 'Accepted',

'203': 'Non-Authoritative Information',

'204': 'No Content',

'205': 'Reset Content',

'206': 'Partial Content',

'207': 'Multi-Status',

'300': 'Multiple Choices',

'301': 'Moved Permanently',

'302': 'Moved Temporarily',

'303': 'See Other',

'304': 'Not Modified',

'305': 'Use Proxy',

'307': 'Temporary Redirect',

'400': 'Bad Request',

'401': 'Unauthorized',

'402': 'Payment Required',

'403': 'Forbidden',

'404': 'Not Found',

'405': 'Method Not Allowed',

'406': 'Not Acceptable',

'407': 'Proxy Authentication Required',

'408': 'Request Time-out',

'409': 'Conflict',

'410': 'Gone',

'411': 'Length Required',

'412': 'Precondition Failed',

'413': 'Request Entity Too Large',

'414': 'Request-URI Too Large',

'415': 'Unsupported Media Type',

'416': 'Requested Range Not Satisfiable',

'417': 'Expectation Failed',

'418': 'I\'m a teapot',

'422': 'Unprocessable Entity',

'423': 'Locked',

'424': 'Failed Dependency',

'425': 'Unordered Collection',

'426': 'Upgrade Required',

'428': 'Precondition Required',

'429': 'Too Many Requests',

'431': 'Request Header Fields Too Large',

'500': 'Internal Server Error',

'501': 'Not Implemented',

'502': 'Bad Gateway',

'503': 'Service Unavailable',

'504': 'Gateway Time-out',

'505': 'HTTP Version Not Supported',

'506': 'Variant Also Negotiates',

'507': 'Insufficient Storage',

'509': 'Bandwidth Limit Exceeded',

'510': 'Not Extended',

'511': 'Network Authentication Required' }

Fetch():

এই অ্যাপিআই এর কাজ হচ্ছে বাইরের রিমোট কোনো সার্ভার থেকে রিসোর্স আনা।