Backend Development

What Is A Backend Development

Language?

Basically, when you make a static website dynamic, backend languages play an Important role. It makes that portion of your website which Stores, organize data,

sends, receives Information, let the users interact with your Website and makes a dynamic web Application.

What are the 3 parts of backend development?

**This typically includes three major parts:**

* The server. This is the computer that receives requests.
* The app. This is the application running on the server that listens for requests, retrieves information from the database, and sends a response.
* The database. Databases are used to organize and persist data.

Skill sets required to become a Backend Developer • Server side Backend engineer should know at least one server-side programming languages their framework like • JavaScript • Node.js • Express • Java(not same as java) • Spring • Python • Flask • C# • ASP.NET MVC • ASP.NET core

 Skill sets required to become a Backend Developer • Database Knowledge of various DBMS technology • MySQL • MongoDB • Oracle • SQLServer • Etc.

Skill sets required to become a Backend Developer • API(Application Programming Interface) Knowledge of web services or API. • Common format of API response is JSON { name: "John", age: 31, city: "New York“ }

* What is JavaScript - ES7 and where can we use it
* Getting all the tools for JavaScript
* Setting up VSCode and Hello World in Javascript
* Basics of variables and constants in javascript
* A talk about numbers in javascript
* Undefined and Boolean Values in Javascript
* If and Else statement in javascript
* Logical operations in javascript
* Var and let keyword - Scope in Javascript
* Arrays and Template literals in javascript
* 3 Must known methods for Arrays in javascript
* Introduction to functions in javascript
* Default parameters in javascript
* ForEach loop in javascript
* For loop in javascript
* === vs == in javascript
* Objects in javascript
* Function that takes object and gives object
* Methods and this keyword in javascript
* Username and Password checking basics in javascript
* Generating a random number in javascript
* Search from array of objects in javascript
* Moving forward in JS - Paths
* Arrow function basics in javascript
* Arrow function and this keyword in javascript
* Coercion in Javascript
* Try catch to avoid program freeze in javascript
* Maps and for of loop in javascript
* Setting up web structure of javascript
* What is DOM
* Where to keep js file and selecting elements
* Changing all texts using javascript
* Creating a new element in DOM
* Event listeners in javascript
* Form validation and simple application
* Advance form validation
* While loop and create text node
* Adding a todo with fade in
* CRUD in localstorage in javascript
* Parsing JSON and stringify
* Classes and objects in javascript
* Inheritance and method overriding
* Our very first project - web server

A web server -

* Configuration for HTML, CSS and JS files
* Finding a file on the server
* Handling server error response
* Finishing up node server project

Learn Express6 lectures

* What is expressJS and templating
* Installing express and details about packages file
* Creating routes using express
* GET, POST and DELETE requests - Postman
* Routing in express
* Server response and status code

Body parser, middleware and view engines4 lectures • 46min

* What is a middleware
* Applying bodyparser with express
* Serving static files and form data
* Using template engine

Multer - Upload a user profile photo4 lectures • 46min

* Overview of multer and documentation
* Setting up multer
* Configuring multer for uploads
* Change profile pic using multer

PassportJS - Facebook Authentication8 lectures • 1hr 8min

* Authentication in nodeJs
* Setting up Facebook app
* Installing dependencies
* Create all views
* Configuring middleware
* Configuring our routes
* http and https problem in facebook auth
* A demo on Heroku - Not a heroku tutorial
* Why we are using MongoDB
* Setting up an Amazon instance using MLab
* 03 Take time and read these NPM docs
* Design scalable folder structure
* Creating home route and setup

Move to Mongodb4 lectures • 36min

* Connect your project with mongoDB
* Creating auth API and a challenge
* Solution of challenge
* Creating a person schema

Move to MongoDB4 lectures • 32min

* Our first query in MongoDB
* Creating new object from Mongo model
* Generating salt and hash to save password
* Using postman for testing

Bigstack - login routes and tokens and User profiles11 lectures • 2hr 10min

* Setting up login route
* Validation of password in login route
* Creating a strategy using passport
* Creating tokens with information
* Fixing errors with profile route
* Creating model for UserProfile
* Creating route for profile
* Collecting user profile values
* Update the profile values and save them
* Debugging routes part 1
* Debugging application - part 2

Unique usernames and work role routes6 lectures • 1hr 2min

* Unique username and url based access
* Getting all users from database
* Deleting a user from database
* Workrole - Pushing array in database
* Testing array based routes
* Writing and testing delete route in array

[1](https://images.slideplayer.com/16/4895831/slides/slide_1.jpg) Introduction to Backend James Kahng

[2](https://images.slideplayer.com/16/4895831/slides/slide_2.jpg) Install Node.js

[3](https://images.slideplayer.com/16/4895831/slides/slide_3.jpg) “End”s Frontend: What you see and interact with as a user HTML, CSS, Javascript Backend The engine running the operation (databases, servers, sessions) Express.js, Ruby on Rails, PHP

[4](https://images.slideplayer.com/16/4895831/slides/slide_4.jpg) Stacks Frontend and backend “stacked” on each other Traditional LAMP Linux Apache (serves front-end HTML, CSS, JS) MySQL PHP Ruby On Rails “full-stack” framework Ruby as backend language

[5](https://images.slideplayer.com/16/4895831/slides/slide_5.jpg) Stacks MEAN MongoDB Express.js AngularJS Node.js Company stacks

[6](https://images.slideplayer.com/16/4895831/slides/slide_6.jpg) Common components in the backend Server Routing API (REST) Database Caching Sessions

[7](https://images.slideplayer.com/16/4895831/slides/slide_7.jpg) Express.js example

[8](https://images.slideplayer.com/16/4895831/slides/slide_8.jpg) Databases Useful web applications know something about their users Ex. Facebook photos, Twitter tweets, Blizzard subscriptions Databases are a systematic way to store and request these kinds of data

[9](https://images.slideplayer.com/16/4895831/slides/slide_9.jpg) Databases The most popular database style for a long time has been the relational database management system (RDBMS) Build databases using tables that have strict structures called “schema” All databases that use SQL queries fall under this category Recently, NoSQL has been rising in popularity Database rankings

[10](https://images.slideplayer.com/16/4895831/slides/slide_10.jpg) What is NoSQL? The traditional SQL database uses “schema” or a skeleton to define our column types rigidly E.g. this database holds the names, addresses, and phone numbers of users NoSQL is any database that doesn’t follow this relational structure Popular databases in the category are MongoDB and Redis

[11](https://images.slideplayer.com/16/4895831/slides/slide_11.jpg) MongoDB Document-based NoSQL database The storage style is similar to JSON objects meaning it is possible to have stored data in these document objects (like API responses) This allows for flexible, fast data retrieval

[12](https://images.slideplayer.com/16/4895831/slides/slide_12.jpg) MongoDB example

[13](https://images.slideplayer.com/16/4895831/slides/slide_13.jpg) Other concerns for a backend developer Caching Store recently-accessed data on the server to minimize load on server People use Redis for this (key-value pair database) Rest APIs Standardized way of interacting with the server’s functionalities GET, POST, PUT, DELETE Sessions Authentication (OAuth)

**Title: Learn node.js & Be an expert by building 10 projects**  
  
  
**1**  
Learn Nodejs by Building 10 projects  
**2**  
What is Nodejs

* An Open source, Cross platform, Event Based and Non-blocking framework used to develop server side and networking applications
* Nodejs applications are written in JavaScript and can be run with its own runtime in Windows,  
  Linux, OS X
* It is not a web server but it is an alternative way to run your code on computer

**3**  
Why Nodejs  
**4**  
Type of Application you can built with Nodejs

* Messaging middleware
* Web Application frameworks
* Servers for HTML5 multi player games, or  
  streaming audio/video
* Cross-platform programs
* Static file servers

**5**  
Popularly used at  
**6**  
Where Can I learn Nodejs  
**7**  
Learn Nodejs by building projects from

* Eduonix learning Solutions is the premier  
  training and skill development organization which  
  provide easy to understand online courses for  
  everyday people.
* This extensive course from the world class  
  trainers of Eduonix highlights practical  
  application of Nodejs technology and will teach  
  you to develop real world applications using the  
  same
* This course will help you to learn technologies  
  like HTML/CSS Frontend technologies, Nodejs NPM,  
  NoSQL database, Column Databases, Database ORM,  
  Express Framework, Asynchronous programming,  
  Karken layer, Drywall user management, Bcrypt  
  encryption, Socket IO and REST APIs

**8**  
Watch Intro video of this course  
**9**  
Projects you will learn in this course

1. Simple Web Server
2. Basic Express Website
3. User Login System
4. Node Blog Systems
5. Community Events
6. BookStore
7. Chat IO
8. Doctor Directory
9. Portfolio App
10. Elearning System

**10**  
What will you get from this course

* Over 86 lectures and 18.5 hours of content!
* Create applications using Nodejs
* Build Scalable and lightweight web  
  applications
* Learn the use of technologies such as  
  Cassandra and MongoDB
* Understand the coding principles behind  
  practical web applications
* Use the Express Framework to build web  
  applications
* Learn the concepts of network programming

1. What is Backend Development. Backend is the server side of the website. It includes activity, like writing APIs, creating libraries, and working with system components. It ensures data integrity and consistency. Involves scripting and writing code to communicate with the database.
2. Who is a Backend Developer Ensures data integrity and consistency Writes code communicate with the database Builds an effective website architecture

Backend Developer skills Data Structure and Algorithms • Stacks • Queues • Trees and Graphs • Sorting Programming languages and Frameworks • Python --> Django • PHP --> Laravel • JS --> Nodejs • Java --> Spring Version Control • Git • Subversion • Concurrent Versions System • Mercurial Database Management • MySQL • PostgreSQL • MongoDB • Oracle Cloud Hosting Platforms • Amazon Web Services (AWS) • Microsoft Azure • Google Cloud Platform (GCP) • Alibaba Cloud

ROLES • Understanding of the performance needs and goals of the websites. • Development and management of APIs • Programming and Framework Utilization • Knowledge of accessibility and security compliance • Writing, coding and testing development solutions • Organization of system logic RESPONSIBILITIES

The backend is the server-side of the website. θ It stores and arranges data, and also makes sure everything on the client-side of the website works fine. θ It is the part of the website that you cannot see and interact with. θ Activities, like writing APIs, creating libraries, and working with system components without user interfaces or even systems of scientific programming, are also included in the backend. Backend

back-end developer is the one working behind-the-scenes, on the infrastructure and databases.

Back-end web development, also called server-side development, alludes to the behind-the-scenes activities that take place when an action is performed on a website. This action could be logging in to one’s account or purchasing a watch from an online store. So a backend developer trains his/ her sights on databases, scripting and the architecture of websites. The code written by such a back-end developer helps in passing on the database information to the browser.

The following are the main back-end development goals:

The main tasks of back-end development are all data-centric. So there’s accessing the data that users ask for through the web browser or app, combining and transforming it, and returning the data.

### 1. *Accessing the data requested by users*

### 2. Combining and transforming the data

### 3.***Returning the data***

4.

**Responsibilities of a Backend Developer**

●All of the data in the platform that requires storage, forms the part of the backend of the website. The main

components of the backend include: an application, a database and a server. The existence of the user facing

side of the website is dependent on the back-end. This technology is built and managed by the backend

developer.

●The key languages that a backend developer must be proficient in, include PHP, Python, Ruby, Java and .NET.

Other tools that backend developers use, include: MySQL, Oracle and SQL.  These are used to serve the data

back to the frontend code. Backend developers also need knowledge of specific PHP frameworks - some of the

common being, Zend, Symfony and Cake PHP. Knowledge of version control software like CVS , GIT or SVN is

equally important along with Linux, used for development and deployment purposes.

●Based on these tools and techniques a backend developer has to develop a code that’s reliable and efficient.

The backend developer has access to other team members to understand the goals of the website and come up

with effective solutions.

Introduction to Backend James Kahng

[2](https://images.slideplayer.com/16/4895831/slides/slide_2.jpg) Install Node.js

[3](https://images.slideplayer.com/16/4895831/slides/slide_3.jpg) “End”s Frontend: What you see and interact with as a user HTML, CSS, Javascript Backend The engine running the operation (databases, servers, sessions) Express.js, Ruby on Rails, PHP

[4](https://images.slideplayer.com/16/4895831/slides/slide_4.jpg) Stacks Frontend and backend “stacked” on each other Traditional LAMP Linux Apache (serves front-end HTML, CSS, JS) MySQL PHP Ruby On Rails “full-stack” framework Ruby as backend language

[5](https://images.slideplayer.com/16/4895831/slides/slide_5.jpg) Stacks MEAN MongoDB Express.js AngularJS Node.js Company stacks

[6](https://images.slideplayer.com/16/4895831/slides/slide_6.jpg) Common components in the backend Server Routing API (REST) Database Caching Sessions

[7](https://images.slideplayer.com/16/4895831/slides/slide_7.jpg) Express.js example

[8](https://images.slideplayer.com/16/4895831/slides/slide_8.jpg) Databases Useful web applications know something about their users Ex. Facebook photos, Twitter tweets, Blizzard subscriptions Databases are a systematic way to store and request these kinds of data

[9](https://images.slideplayer.com/16/4895831/slides/slide_9.jpg) Databases The most popular database style for a long time has been the relational database management system (RDBMS) Build databases using tables that have strict structures called “schema” All databases that use SQL queries fall under this category Recently, NoSQL has been rising in popularity Database rankings

[10](https://images.slideplayer.com/16/4895831/slides/slide_10.jpg) What is NoSQL? The traditional SQL database uses “schema” or a skeleton to define our column types rigidly E.g. this database holds the names, addresses, and phone numbers of users NoSQL is any database that doesn’t follow this relational structure Popular databases in the category are MongoDB and Redis

[11](https://images.slideplayer.com/16/4895831/slides/slide_11.jpg) MongoDB Document-based NoSQL database The storage style is similar to JSON objects meaning it is possible to have stored data in these document objects (like API responses) This allows for flexible, fast data retrieval

[12](https://images.slideplayer.com/16/4895831/slides/slide_12.jpg) MongoDB example

[13](https://images.slideplayer.com/16/4895831/slides/slide_13.jpg) Other concerns for a backend developer Caching Store recently-accessed data on the server to minimize load on server People use Redis for this (key-value pair database) Rest APIs Standardized way of interacting with the server’s functionalities GET, POST, PUT, DELETE Sessions Authentication (OAuth)

Web development languages: A backend developer needs to know at least one server-side programming language. This includes Java, Python, Ruby, . Net, PHP, C++, and so on. Let us take a quick look at these.

### a. **Python**

Easily the most popular programming language that is used by leading companies and startups across the world. This multi-purpose language is used in Data Science, Artificial Intelligence, Machine Learning, Web development, Internet of Things (IoT) and more.

### b. **Java**

Also hugely popular today given that around 90 percent of Fortune 500 companies are said to use it. It continues to be used to make Android mobile apps, large backend environments and game engines.

### c. **Ruby**

With its large standard library, enough compatibility to allow implementation on all major platforms and native plug-in API in C, Ruby is quite popular today. This interpreted, high-level, general-purpose programming language is dynamically typed and it also uses garbage collection.

### d. **C++**

Often referred to as the foundation which several other programming languages have been built on, C++ is used to create applications that run locally on machines such as your computer.

### e. **PHP**

A general-purpose scripting language particularly ideal for web development, PHP or PHP: Hypertext Preprocessor is a leading server-side programming language. It is widely used for making dynamic and interactive web pages.

Databases and caches: Given their goals, back-end developers need to have thorough knowledge about various Database Management System (DBMS) technologies; the leading names being MySQL, MongoDB, Oracle, SQLServer, Redis. Knowledge about caching mechanisms such as varnish, Memcached and Redis is also very useful.

Servers: Experience of working with servers such as Apache, Nginx, Microsoft Internet Information Services (IIS) is certainly quite valuable for a back-end developer. A strong grasp over Linux goes a long way too, especially when it comes to administering servers.

API (REST & SOAP): Back-end and full stack developers stand to benefit considerably from the knowledge of web services or APIs. Another key factor is experience in building and utilising REST (Representational State Transfer, a software architectural style defining a set of constraints to be used for creating web services) and SOAP (Simple Object Access Protocol, a messaging protocol for interchanging data in a decentralized and distributed environment) services.

## **Roles and responsibilities of back-end developers**

Now let us look at all that a back-end developer does using the tools mentioned above. For starters, he/ she needs to:

• Understand the key objectives of the website and come up with effective solutions

• Store data well and ensure that it is smoothly delivered to the user who is supposed to have access to it

• Develop a payment processing system where data is accepted, securely stored and correct charges are made

• Manage API resources that work across devices

• Help build and manage frameworks or the architecture to make it easier to program against.

• Organize the logic of the system that runs across various devices

• Help with data science analyses

• Help implement algorithms

• Help solve system-related problems

## **Skills needed to become a back-end developer**

A back-end developer should have the following:

• Preferably a degree in Computer Science or similar field

• Proficiency in coding languages such as Java, Python, Ruby, .Net is a must

• Proficient knowledge of the framework that is used by the company

• Understanding of front-end web technologies like HTML, CSS, and JavaScript (to communicate with team members on the front-end)

• Ability to manage a hosting environment, including databases and scaling applications to handle load changes

• Knowledge of accessibility and security compliance

• Experience with version control, such as Git

• Excellent problem-solving skills

• Good communication skills

MAANG

### Expertise in Solving Data Structures and Algorithms

### 2. Contribution to Open-source Projects

### 3. Coding Competition Expertise

### 4. Thinking Out of the Box