|  |  |
| --- | --- |
| **Job Description** | |
| **Position** | **Full Stack (Node.js) Developer** |
| **Reports to** | Chief Solutions Architect/ Product Owner |
| **Location** | Dubai (UAE) |
| **Timing** | 09:00 am – 06:00 pm GST |
| **About DIFX** | DIFX is the first comprehensive ecosystem to offer major trading verticals in a single unified and packaged offering. We provide a secure and hybrid working paradigm for a centralized and decentralized exchange with atomic swaps.  DIFX is the world’s first fully insured cross-asset trading platform. This platform was created as a Blockchain-based crypto currency exchange to bring significant and lasting change to financial markets. We envision using our immutable ledger technology for the unbanked, by creating a revolutionary exchange which is fast, secure, and efficient. Our number one priority is to bridge the gap between digital and traditional assets, thereby establishing a new financial order. |
| **Why work for DIFX?** | Each one of our people is part of one team, leveraging the wealth of our collective knowledge, passion and creativity. This is your opportunity to join a passionate team, working with people who are leading the industry through innovation, growth and value. Alongside amazing colleagues and engaging work, we want to help you get the best out of your career. We offer continuous learning and development to take your skills to the next level. It’s not just about being the best you can at work, but we also offer a variety of benefits to make your non-work life better; including paid holiday, flexible working, health and well-being initiatives and many more. If you’re looking to build your career, work with experts and most of all have fun, join us. |
| **Job Summary** | A passionate Full Stack Node.js Developer is one who believes in creating next-generation user interfaces that behave in a natural and humane way with their users and is considerate of their frailties.  If you are one such enthusiast who satisfies all the aforementioned conditions, we look forward to you joining our team.  We are looking for a Full Stack Node.js Developer with experience in building high-performing, scalable applications. |
| **Responsibilities** | * You will be responsible for Node.js based application development while providing expertise in the full software development lifecycle, from concept and design to testing * Develop application code for programs while following coding standards * Knowledge and understanding of Backend and Frontend technologies * Knowledge of SQL/NoSQL databases * Understanding and Knowledge of database, s/w development methodology and lifecycle. * Develop and execute unit tests. * Solve problems independently working with new technologies to create quick solutions. * You will work within a team, collaborate and add value through participation in peer code * Reviews, provide comments and suggestions, work with cross functional teams to achieve goals * Experience working with agile methodologies. |
| **Skills, Qualification & Experience** | * An engineering degree B.E/B.Tech/MCA/M.Tech/M.Sc * 4-6 years of total experience as a Full stack developer * At least 4 years of relevant experience working on Node.js * At least 3 years of experience working with node.js frameworks such as express.js and nest.js. * Hands-on experience with TypeScript at least one live project. * Hands-on at least one front end technology like Angular, ReactJs, VueJs. * Experience building secure RESTful APIs. * Hands-on with relational, non-relational databases (MySql, PostgreSQL, MongoDB etc). * Excellent in version control, especially in Git. * Excellent understanding of JavaScript fundamentals and ES6 standards. * Familiar with cloud web services like AWS, Azure or Google Cloud. * Proficiency in programming, data structures and algorithms. * Ability to architect, build and maintain complex Node.js applications with clean code. * Strong analytical and problem-solving skills and an entrepreneurial mind-set. |
| **Notice** | Immediate or up to 15 days |
| **Employment Type** | Full-time |

DIFX is an equal opportunity employer and makes employment decisions without regard to race, color, religion, sex, sexual orientation, gender identity, national origin, protected veteran status, disability status, age, or any other status protected by law.

Offer of employment with DIFX is conditioned upon the successful completion of a background verification check, subject to applicable laws and regulations.

Share your CV on [career@difx.io](mailto:career@difx.io)

Screening Questions:

How many years of work experience do you have using Node.js?

How many years of work experience do you have working as a Full Stack Developer?

What is your current monthly salary?

What is your expected monthly salary?

If you are serving notice then how many days are remaining? (Mention your exact notice days if you are not serving notice)

We must fill this position urgently. Can you start immediately?

Do you have any offer in hand?

Interview Questions

1. What is the use of NODE\_ENV? Does it affect the performance of application?

If the project is in the production stage, Node.js promotes the convention of making use of NODE\_ENV variable to flag it. This helps in taking better judgment during the development of the projects. Also, when you set your NODE\_ENV to production, your application tends to perform 3 times faster.

1. Differentiate between readFile vs createReadStream in Node.js?

Node.js provides two ways to read and execute files which are using readFile and CreateStream. readFile() is a fully buffered process which returns the response only when the complete file is pushed into the buffer and is read. It is a memory intensive process and in case of large files, the processing can be very slow. Whereas createReadStream is a partially buffered which treats the entire process as an event series. The entire file is split into chunks which are then processed and sent back as a response one by one. Once done, they are finally removed from the buffer. Unlike readFile, createReadStream is really effective for the processing of the large files.

1. Explain the concept of Punycode in Node.js?

In Node.js, Punycode is an encoding syntax that is used for converting Unicode (UTF-8) string of characters into a basic ASCII string of characters. It is important as the hostnames can only understand the ASCII characters. Thus, Node.js version 0.6.2 onwards, it was bundled up with the default Node package. If you want to use it with any previous versions, you can easily do that by using the following code:

Syntax:

punycode = require('punycode');

1. Does Node.js provide any Debugger?

Node.js do provide a simple TCP based protocol and debugging client that comes built-in. In order to debug your JavaScript file, you can use the below debug argument followed by js file name that you want to debug.

Syntax:

node debug [script.js | -e "script" | &lt;host&gt; : &lt;port&gt; ]

1. Is cryptography supported in Node.js?

Yes, Node.js does support cryptography through a module called Crypto. This module provides various cryptographic functionalities like cipher, decipher, sign and verify functions, a set of wrappers for open SSL’s hash HMAC etc. For example:

Syntax:

1 const crypto = require'crypto');

2 const secret = 'akerude';

3 const hash = crypto.createHmac('swaEdu', secret).update('Welcome to Edureka').digest('hex');

4 console.log(hash);

1. Explain the reason as to why Express ‘app’ and ‘server’ must be kept separate.

Express ‘app’ and ‘server’ must be kept separate as by doing this, you will be separating the API declaration from the network related configuration which benefits in the below listed ways:

* It allows testing the API in-process without having to perform the network calls
* Faster testing execution
* Getting wider coverage metrics of the code
* Allows deploying the same API under flexible and different network conditions
* Better separation of concerns and cleaner code

API declaration should reside in app.js:

1 var app = express();

2 app.use(bodyParser.json());

3 app.use("/api/events", events.API);

4 app.use("/api/forms", forms);

Server network declaration should reside in /bin/www:

1 var app = require('../app');

2 var http = require('http');

3 //Get port from environment and store in Express

4 var port = normalizePort(process.env.PORT || '8000');

5 app.set('port', port);

6 //Create HTTP server.

7 var server = http.createServer(app);