// Place your answers for questions 1 and 2 here. Be sure to label each answer clearly.

**The answers for the Question 1 and Question 2 are as follows:**

**Question 1:** The answer to Question 1 are as follows:

1. **JavaScript Runtime and interaction with Chrome V8 engine:**

**JavaScript Runtime:** JavaScript runtime is an environment which does enable the V8 engine to run and enable a developer to access APIs to build programs. We can think of this environment as a huge container, containing V8 engine enabling it to run, and this environment is the one which helps the Chrome browser(generally any browser) to provide various modules which can be used together, like DOM, various methods, objects, properties, which are made available inside the JavaScript runtime environment by the browser.

Just like there are things which are provided by the Chrome browser inside the runtime environment, there are things which are provided by Node.js i.e. runtime environment on the server side. With Node.js for example, the reading and writing files is not something that V8 engine can do, and that is provided by the Node.js which is called binding, this whole structure is inside the JavaScript runtime.

**Interaction with Chrome’s V8 Engine:** Node.js uses the Google Chrome’s V8 JavaScript engine, coming to the JavaScript engine it is an assembler that is used to interpret the JavaScript code and provide the results. It is used by browsers (client side) and also on the server side. Node.js used Chrome’s V8 JavaScript engine to be used on server side.

The interaction of Node.js with V8 engine can be understood a bit with the last question’s last paragraph, an example with detail is provided here. Let’s take a scenario where the operation that needs to be performed is read and write on a local drive, in this case the V8 engine is not able to perform, in this scenario the binding rights are passed on to the Node.js and that performs the operation.

Other technologies that use Chrome’s V8 engine on the server side include – Couchbase and MongoDB.

1. **Event Driven / Non-blocking:**

**Event Driven:** The term Event Driven means that Node.js is always waiting for events to happen, events can be anything from client requesting some data or a website. The term can be understood with how Node.js works, it does have an event loop which is single threaded and constantly circling for an event. So the Event Loop for the node.js is a cyclic event single threaded which handles the request and processes them accordingly, the section “b” does discuss about the whole process with example.

**Non-blocking:** The term Non-blocking is a continuous explanation to the part a. The event loop is the one that takes on event and processes the request. Since the event loop is single threaded and can lead to blocking when a resource intensive request it made, to avoid the blocking the event loop pushes those requests to the API runtime which can let it be handled asynchronously. This whole process makes sure that there is no blockage and the event loop is always free to take on events.

Let’s take for example that a resource intensive request is made by the user to a server with Node.js on top of it, once the request is made, the event loop will get that the request is heavy and it will be transferred to the multi-threaded OS and this keeps the event loop free for other events.

Once the resource intensive event has been performed, event loop is notified by the runtime and the event loop sends it back to the source of the request.

1. **NPM:** The whole Node ecosystem is built on the idea of modules, now let’s understand what a module is, a module is a file with set of JavaScript codes (in this specific case). Which can be used over and over again for specific features, libraries to run the code and adding new dimension to the code. There are three categories in modules – Built in Core Modules, Local Modules, Third Party Modules. All these packages can be used for features as per required. Of these three categories, Third Party Modules can be installed using Node Package Manager, which helps in installing and using different versions of same module categories.

The steps are following to reproduce the NPM installation and package import:

1. To install a package globally and as an application dependency are as follows:

To install NPM package as global: npm install -g <package name>

To install NPM package as application dependency: To install the dependencies just use the command inside the application folder. npm install

1. Import a package into JS file to be used: To import package from other modules, we can use the following command.

Import {<value>} from ‘path of module’;

**Question 2:** The answer to Question 2 are as follows:

1. **MVC:** MVC or the Model View Controller is a Software Design Pattern
2. **Advantages:** The

Resource help taken from for these questions:

1. Zoom Recording (MMIS 6500) - September 3
2. Wikipedia (Node.js)
3. <https://nodejs.dev/learn/the-v8-javascript-engine>
4. Node JS coursework textbook
5. <https://blog.sessionstack.com/how-javascript-works-inside-the-v8-engine-5-tips-on-how-to-write-optimized-code-ac089e62b12e> (Amazing article)
6. <https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller>
7. <https://docs.npmjs.com/cli/install>