**Why would you consider a Scripting Language as JavaScript as your Backend Platform.**

It depends. If my backend would be big, complex and need a lot of structure
I would probably write the backend in Java instead, since that is what Java does.
It's structured, it delivers solid engineering and architecture.
Scripting languages, like JavaScript, are super fast, it doesn't have to be compiled and sometimes measured up to 20x faster than code written in Java or any other alternative. It's simple and flexible.

**Explain, using a relevant examples, your strategy for implementing a REST-API with Node/Express and show how you can "test" all the four CRUD operations programmatically using for example the Request package.**

See RestWithExpress.js

**Explain, using relevant examples, about testing JavaScript code, relevant packages (Mocha etc.) and how to test asynchronous code.**

To test JavaScript code, we can install and use the test librabry "Mocha". In Mocha we use a describe block, which corresponds to a test suite where we can place all our test. The describe block takes two parameters,the first one is the name of the block, and the second is an anonymous function, where we can place the so called "it blocks" which corresponds to our test methods. Like the describe block the it-blocks also takes two parameters; a name, and an anonymous function. Inside the anonymous function we can use the different assertion libraries like "chai", two assert different scenarios.

See MochaTest.js

**Explain, using relevant examples, concepts related to the testing a REST-API using Node/JavaScript + relevant packages**

See RestTest.js

**Explain, using relevant examples, different ways to mock out databases, HTTP-request etc.**

Mock objects are simulated objects that mimic the behavior of real objects in controlled ways. They have the same interface as the real objects they mimic, allowing a client object to remain unaware of whether it is using a real object or a mock object.
Nock is an HTTP mocking and expectations library for Node.js
Nock can be used to test modules that perform HTTP requests in isolation (that is without performing a real network operation).

Example: see Mocking.js