1 - HTTP API Endpoints

1. Learning Outcomes

On completion of this lab you will have:

- Installed the Node JS environment
- Installed the Postgres RDBMS
- Created a simple HTTP endpoint in Node
- Interfaced between Node and Postres using Massive JS
- Executed simple Postgres queries using SQL and exposed those using an HTTP API

2. Organisation

Please complete the exercises individually.

3. Grading

This worksheet is worth up to 7.5% of your overall module grade. You must attend and sign in at 10 labs in order to obtain full credit for your submitted worksheets. You may work on this worksheet during lab 1 and lab 2 with instructor assistance. You must also demonstrate your submission in order to receive credit - see below.

4. Submission

The deadline for submission is Wednesday Fri Feb 10, 2017 @23:59 through Webcourses.

5. Demonstration

You will give a brief demonstration of your submission to the lab instructor in lab 3.

6. Requirements

For this lab you will need to

- Use your own laptop with local tools or,
- Sign up for a free account with a cloud provider such as Nitrous.IO and Heroku

7. Resources

You are free to research whatever you need to solve the problems in this lab. Some recommended resources include:

- https://nodejs.org/en/
- https://www.postgresql.org/download/windows/
- https://www.postgresql.org/download/macosx/
- http://postgresguide.com/setup/example.html
- http://massive-js.readthedocs.io/en/latest/
- http://www.craigkerstiens.com/2015/11/30/massive-node-postgres-an-overview-and-intro/

8. Problem Sets

The following platform-independent tasks can be solved on Windows, Mac local Linux or Cloud Linux as you prefer

For your lab submission, take screenshots or cut-n-paste your solutions into a document which you will submit through webcourses

1	Install Node JS on your laptop or sign up for a free cloud-based Node provider. Verify that node and npm are installed and working correctly	10 Marks

2	Create a new project folder, change into it and run the following
	
	`npm init`
	`(entry point: index.js)`
	`npm install express --save`
	
	In your folder, create an index.js with the contents from the **Express** tutorial example at https://expressjs.com/en/starter/hello-world.html
	
	Run and verify that the your endpoint is responding at your designated port. For example on Mac or Linux
	
	`PORT=3000 npm start`
		15 Marks

3	Install Postgres (>= 9.5) on your laptop or sign up for a free cloud-based provider (*) (*) In some circumstances when using Postgres in the cloud, you may encounter college firewall problems communicating to your database instance. In this case, if available, access the Internet via a tethered phone or other 3G/4G access	10 Marks

<table>
<tr><td></td><td>point

Verify that your postgres server instance is working

<table><tr><td>pg_isready</td></tr></table></td><td></td></tr>
<tr><td>4</td><td>Load the sample database from the Postgres Guide

http://postgresguide.com/setup/example.html

Inspect the schema and table data using the psql client, e.g.

<table><tr><td>\d
\d products
table products;</td></tr></table></td><td>15 Marks</td></tr>
<tr><td>5</td><td>Install Massive JS from http://massive-js.readthedocs.io/en/latest/

Massive JS uses database reflection to create Javascript APIs to allow CRUD operations on a specified schema

Following the tutorial at http://massive-js.readthedocs.io/en/latest/quick_start/, rework the example models to access the sample database installed in step 4 (i.e. skip creating the example user model and focus on the user, products, purchaes and purchase_items tables from the step 4 instead.</td><td>25 Marks</td></tr>
<tr><td>6</td><td>Using Node, Express and Massive create the following HTTP API endpoints serving the following resources as JSON documents

<table><tr><td>GET /users</td><td>List all users</td></tr><tr><td>GET /users/:id</td><td>Show details of the specified user</td></tr><tr><td>GET /products</td><td>List all products</td></tr><tr><td>GET /products/:id</td><td>Show details of the specified products</td></tr><tr><td>GET /purchases</td><td>List all purchases</td></tr><tr><td>GET /purchases/:id</td><td>Show details of the specified purchases</td></tr></table></td><td>25 Marks</td></tr>
</table>

	Test each of these endpoints serves the expected data	