TELUS

React/Node.js 101

February 12th, 2021

Your workshop hosts

Sara

GTLP Technology Specialist

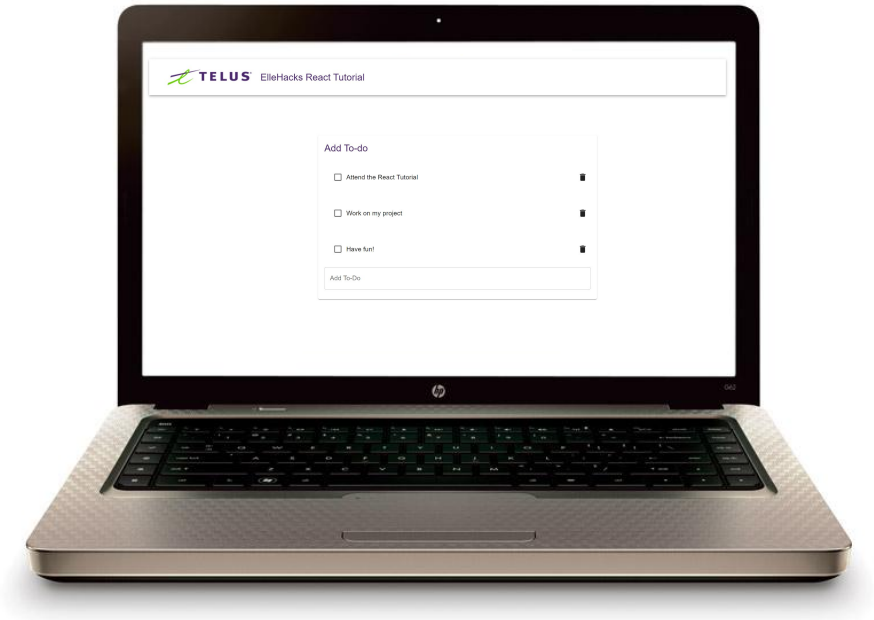
linkedin.com/in/sara-ghaemi/

Daniel

GTLP Technology Specialist

linkedin.com/in/daniel-levy789/

What you will get out of the workshop

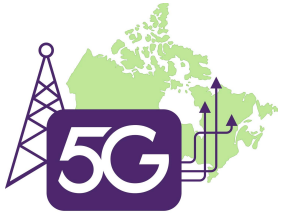


- Get an introduction to TELUS' Graduate Technology Leadership Program **(5 minutes)**
- Build a full stack web application using the MERN stack (Mongo, Express, React and Node.js) **(45 minutes)**
- Introducing the TELUS' Hackathon theme and Q&A **(10 minutes)**

TELUS' Graduate Technology Leadership Program

Our rotational program provides **targeted development** in the latest technologies

Next-generation wireless networks

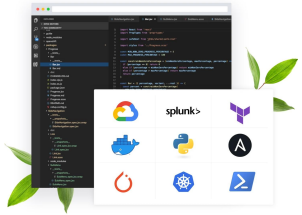


SDN / NFV

(Software-defined networking / Network function virtualization)

Cybersecurity and privacy solutions

Software Development



Future friendly technology services

(formerly Digital Home Solutions)

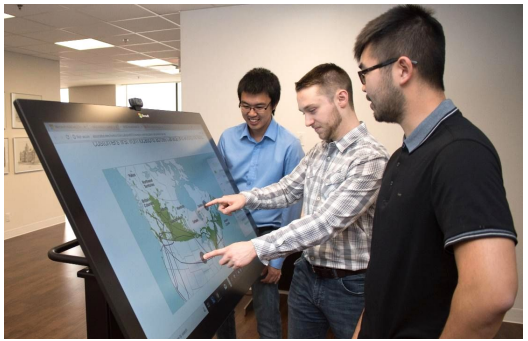
Learn more at **telus.com/gtlp**

The experience within GTLP

- Become a part of a great **workplace community**
- Accelerate your growth through **peer and leader mentorship**
- Work with **industry professionals and teams** from across the company
- Lead and partake in **sub-programs** within GTLP

GTLP and co-op programs

Applications are **OPEN** NOW: Apply by **Feb 26** for GTLP or **early March** for our co-op program at https://careers.telus.com/go/Students-and-New-Grads/2637017/



Connect with us

@telusgtlp

TELUS GTLP

gtlp@telus.com

telus.com/gtlp

Commitment: Empowering Women in STEM

Women in STEM Panels

Domestic Violence Awareness

Diversity Coffee Chats

University Mentorship for Women in STEM

International Women's Day

Leadership Opportunities

Learn more at **telus.com/about/diversity-and-inclusion**

What we are going to build today

FRONTEND
Present data

API
Share data

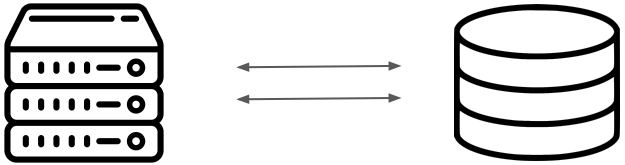
BACKEND
Manage data

DATABASE
Store data

CLIENT

GET /tasks

[{ id: 1, text: ‘Buy Milk’, isCompleted: False },
{ id: 2, text: ‘Visit the Vet’, isCompleted: False }]



SERVER

The tools you will need

Required

Node.js Runtime v12+

Node Package Manager V6+
(Included in Node.js install)

Local Mongodb Database
Installation guide

Code Editor
e.g. VS Code, Sublime, VIM

Internet Browser
e.g. Chrome 49+, Firefox 50+, Safari 10+, IE 12+

Recommended

VS Code

Google Chrome

Optional

git
Local version control

GitHub
Code collaboration

Getting started

Step 1. Clone OR download the files

```
$ git clone https://github.com/daniel-levy/ellehacks-react-workshop.git
```

Step 2. Verify Node installation

```
$ node -v
```

Step 3. Follow the Mongo installation guide and start the local server if not running already in a new terminal

```
$ cd <MONGO LOCATION>
$ mongod
```

Step 4. Open the app folders in a separate terminal

```
$ cd app
$ npm install
$ npm start
```

Step 5. Go to localhost:3000 in your browser

Application Demo

Tools we are using to create a server

A javascript runtime based on chrome's javascript engine.

A back end web application framework for Node.js

Setting up your server

```
    // www
…   ...
    // Get port from environment and store in Express
13  var port = normalizePort('8080');
14  app.set('port', port);
15
16  // Create HTTP server.
17  var server = http.createServer(app);
18
19  // Listen on provided port, on all network interfaces.
20  server.listen(port);
21  server.on('error', onError);
22  server.on('listening', onListening);
…   ...
```

Refer to **/backend/bin/www** in repo

/server/ *(noun)*:
A machine (or CPU process) that **provides** data and/or a service to clients

/client/ *(noun)*:
A machine (or CPU process) that **requests** data and/or a service from the server

Note: A single machine (e.g. your laptop) can run as both a client and a server at the same time

Setting up your server

```
    // app.js
…   ...
3   const express = require('express'); // import the library
4   const todoRouter = require('./routes/todoRoutes');
5   const cors = require('cors');
6   const app = express(); // create the express object
…   ...
15  app.use(cors({origin: '*'}));
16  app.use('/todos', todoRouter); // load the router module
…   ...
```

Refer to **/backend/src/app.js** in repo

Note: Express documentation can be found here: https://expressjs.com/en/guide/routing.html

Communicating between client and server

/API/ *(noun)*:
Application programming interface. It enables two entities to communicate with each other.

Four common examples of actions in REST APIs:

- **GET:** request data from a server
- **POST:** send new data to server
- **PUT:** update existing data on a server
- **DELETE:** remove existing data on a server

Creating your endpoints

```
    // todoRoutes.js
…   ...
5   router.get('/', async (req, res) => {
6     const todos = await getTodos()
7     res.send(todos);
8   });
9
10  router.post('/', async (req, res) => {
11    const text = req.body.text;
12    const result = await addTodo(text)
13    res.send(result);
14  });
…   ...
```

Refer to **/backend/src/routes/todoRoutes.js** in repo

Creating your endpoints

```
15  router.put('/:id', async (req, res) => {
16    const id = req.params.id;
17    const todoText = req.body.text;
18    const isCompleted = req.body.isCompleted;
19    const result = await updateTodo(id, todoText, isCompleted)
20    res.send(result);
21  });
22
23  router.delete('/:id', async (req, res) => {
24    const id = req.params.id;
25    const result = await deleteTodo(id)
26    res.send(result);
27  });
```

Refer to **/backend/src/routes/todoRoutes.js** in repo

Creating your controller

```
const { addValueToDb, getAllValuesFromDb,
updateValueInDb, deleteValueFromDb } = require("../services/dbService")

const getTodos = async () => {
  const values = await getAllValuesFromDb()
  return values
}
const addTodo = async (text) => {
  return await addValueToDb(text)
}
const updateTodo = async (id, todoText, isCompleted) => {
  return await updateValueInDb(id, todoText, isCompleted)
}
const deleteTodo = async (id) => {
  return await deleteValueFromDb(id)
}
```

Refer to **/backend/src/controllers/todoController.js** in repo

TELUS

React Basics

What is React?

React is a **JavaScript library**

Currently one of the most popular JavaScript libraries

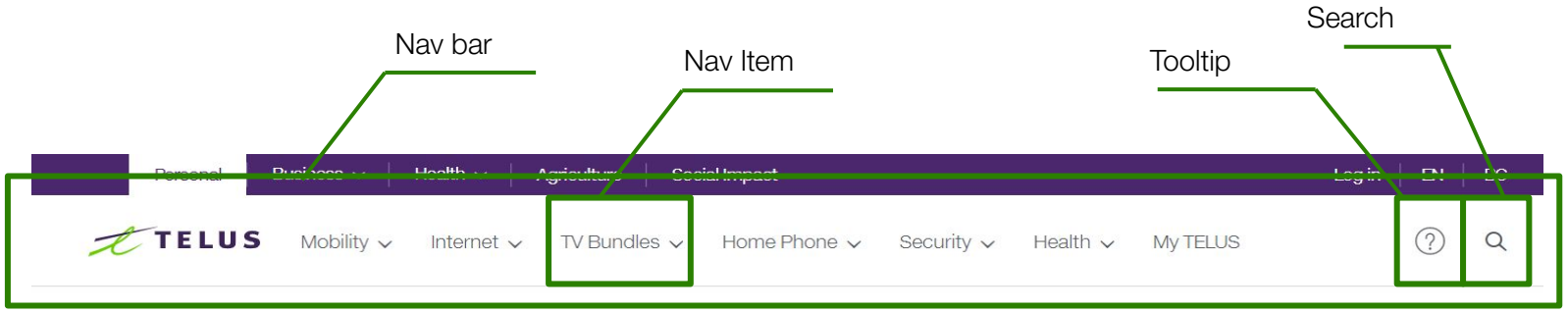
Used to build user interfaces

Uses JSX syntax (a mix of HTML and JS)

Made up of multiple **reusable code blocks** called components

```
const name = 'Josh';
const element = <h1>Hello, {name}</h1>;
```

Components



Reusable UI blocks

UI is made up of multiple components put together

Allows you to think about each piece of the UI in isolation

Essentially a **function that returns HTML elements**

Components

```
9   const Todo = ({ todo, toggleTodo, removeTodo }) => {
10    return (
11      <Box sx={{ display: 'flex', p: 1, borderRadius: 1, alignItems: 'center'}}>
12        <FormControl component="fieldset" variant="standard" sx={{ flexGrow: 1, p: 2 }}>
13          <FormGroup>
14            <FormControlLabel
15              key={`${todo._id}-${todo.text}`}
16              control={
17                <Checkbox
18                  checked={todo.isCompleted}
19                  onChange={() => {toggleTodo(todo._id, todo.text, todo.isCompleted)}}
20                  name={`${todo._id}-${todo.text}`}
21                />
22              }
23              label={todo.text}
24            />
25          </FormGroup>
26        </FormControl>
27        <DeleteIcon onClick={() => {removeTodo(todo._id)}}/>
28      </Box>
29    )
30  }
```

Refer to **/frontend/src/components/Todo.js** in repo

Props

Similar concept to **arguments/parameters** in plain JS

Passing data between components is done through props

Props allow you to create **generic components** that behave differently based on the given props


```
<Button variant="standard">Standard Common</Button>
```


```
<Button variant="secondary">Find out more</Button>
```

Props

```
45  <div>
46    <Header />
47    <PaddedContent>
48      <Card sx={{ width: 700 }}>
49        <CardContent>
50          <Typography variant="h5" component="div" sx={{ color: "#4B286D", paddingBottom: '12px' }}>
51            Add To-do
52          </Typography>
53          <>
54            {todos.map((todo) => {
55              return(<Todo key={todo._id} todo={todo} toggleTodo={toggleTodo} removeTodo={removeTodo}></Todo>)
56            })}
57          </>
58          <TodoForm addTodo={addTodo}/>
59        </CardContent>
60      </Card>
61    </PaddedContent>
62  </div>
```

Refer to **/frontend/src/App.js** in repo

States

Similar concept to **props** but is managed within the component

Similar to **variables** declared within a function

When states are changed, the component re-renders

Hooks

Hooks are special functions that allow you to use React features

UseState: declare a state variable and initializes it

UseEffect: performs a function after rendering

```
1   import React, { useState } from 'react';
2   function App() {
3     const [count, setCount] = useState(0);
4
5     function handleClick() {
6       setCount(count + 1);
7     }
8
9     return (
10      <div>
11        <button onClick={handleClick}>Click me!</button>
12      </div>
13    )
14  }
```

How to ***kickstart*** your own web application

1
Start with our base

Don't reinvent the wheel! Clone the repo OR download the zip

2
Mockups

Before coding, try designing an interactive prototype using Figma

3
Improve the BE

Try adding more endpoints. Consult the Express documentation.

4
Improve the UI

Add new components. Try 'Material UI' component library.

Challenge Statement

- **Leverage technology to support social sustainability and community building**
- Some example topics include technological efforts towards:
  - Improving accessibility to basic human needs
  - Support resources for individuals and community groups facing social issues
  - Social engagement and connection

Questions?

let’s make the future friendly™