NODE.JS WEB SERVERS

DEVELOPING WEB APPLICATIONS
ACIT 2520

LEARNING OUTCOMES

- Learn how to use middleware
- Working with Git
- Deployment to cloud

LET'S START!

CONTINUE WITH THE PREVIOUS PROJECT

- Let's continue with our previous project
- First, let's add some middleware to our application
- Middleware is a function that has information about the *request*, *response* and the *next* function that will be executed after the called functions finishes execution

```
app.use((request, response, next) => {

});

app.get('/', (request, response) => {
    // response.send('<h1>Hello Express!</h1>');
    response.send({
        name: 'Your Name',
        school: [
            'BCIT',
            'SFU',
            'UBC'
        ]
    })
});

app.get('/info', (request, response) => {
    response.render('about.hbs', {
        title: 'About page',
        year: new Date().getFullYear(),
        welcome: 'Hello!'
    });
});

app.get('/404', (request, response) => {
    response.send({
        error: 'Page not found'
```

ADDING MIDDLEWARE

- Middleware works in synchronized way - which means no requests will be executed until we'll "unlock" the middleware
- Go ahead and run the code on your server and test if you can access it

```
app.use((request, response, next) => {

});

app.get('/', (request, response) => {
    // response.send('<h1>Hello Express!</h1>');
    response.send({
        name: 'Your Name',
        school: [
            'BCIT',
            'SFU',
            'UBC'
        ]
    })
});

app.get('/info', (request, response) => {
    response.render('about.hbs', {
        title: 'About page',
        year: new Date().getFullYear(),
        welcome: 'Hello!'
    });
});

app.get('/404', (request, response) => {
    response.send({
        error: 'Page not found'
```

BREAKPOINT

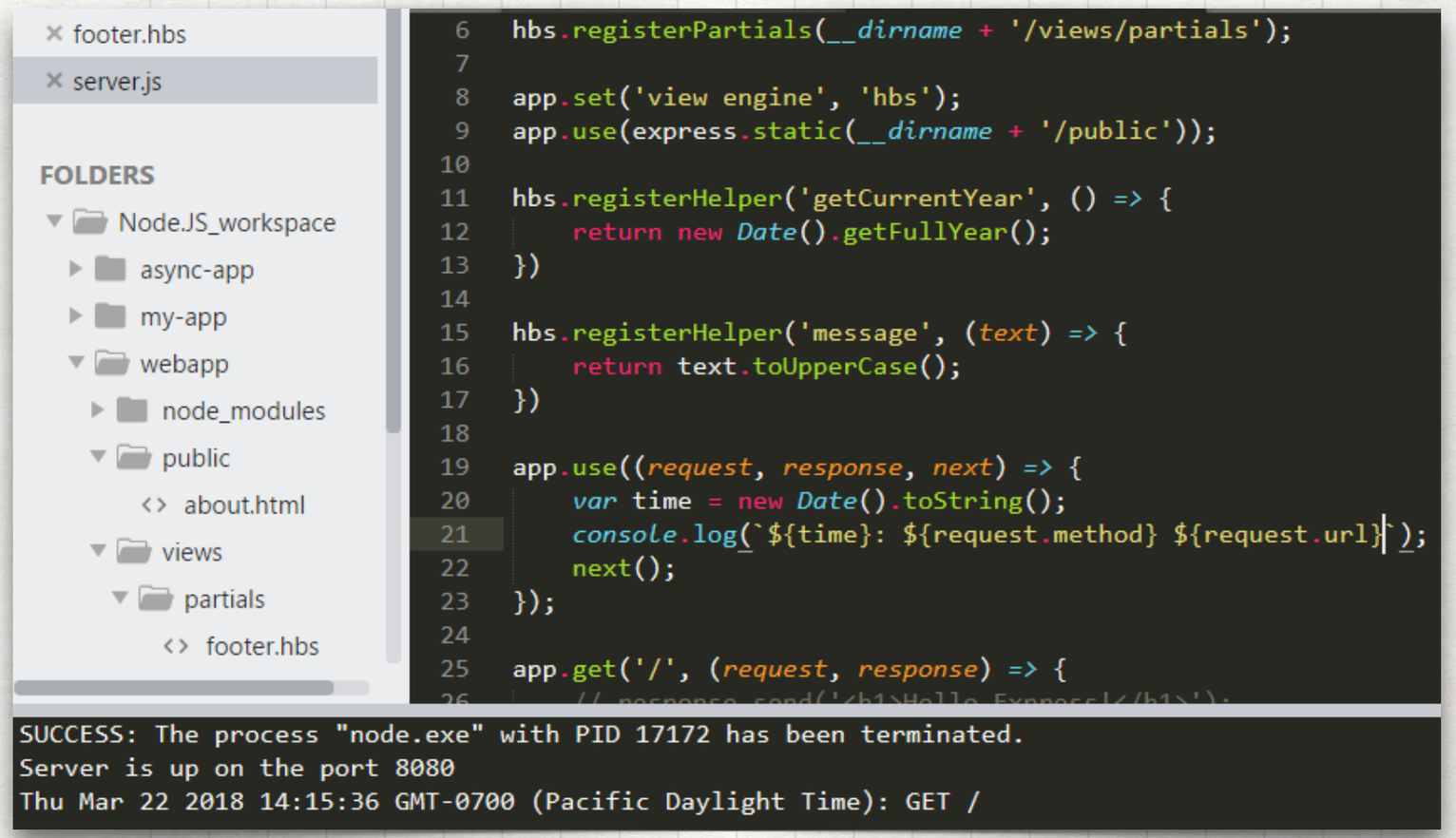
ADDING MIDDLEWARE

- The answer is no
- What we need to do, we need to use *next()* to allow further execution
- Let's see the example on the next slide

```
app.use((request, response, next) => {

});

app.get('/', (request, response) => {
    // response.send('<h1>Hello Express!</h1>');
    response.send({
        name: 'Your Name',
        school: [
            'BCIT',
            'SFU',
            'UBC'
        ]
    })
});

app.get('/info', (request, response) => {
    response.render('about.hbs', {
        title: 'About page',
        year: new Date().getFullYear(),
        welcome: 'Hello!'
    });
});

app.get('/404', (request, response) => {
    response.send({
        error: 'Page not found'
```

ADDING MIDDLEWARE

- Let's say we want to add logging to our application
- In our case, let's use simple *console.log()*



BREAKPOINT

ADDING MIDDLEWARE

- Now, instead of console logging, let's actually create a log file and append it, using *fs* library (please remember to require it!)

```
20  app.use((request, response, next) => {
21      var time = new Date().toString();
22      // console.log(`${time}: ${request.method} ${request.url}`);
23      var log = `${time}: ${request.method} ${request.url}`;
24      fs.appendFile('server.log', log + '\n', (error) => {
25          if (error) {
26              console.log('Unable to log message');
27          }
28      });
29      next();
30  });
```

CHALLENGE #10 - PART I

- Create a maintenance page that will be rendered with middleware

- Rules:
  - use *.hbs* template with simple HTML to display e.g. "The site is currently down for a maintenance"
  - create a new controller, using middleware
  - the function you'll create should be commented in order to get the site to work back normally

BREAKPOINT

ADDING GIT TO OUR PROJECT

- Next step will be to add git in our project
- To do this, please run *git init* in terminal, in your project's folder
- Next, you can run *git status* to check which files are not synchronized with the main repository

```
C:\Users\A01041335\Documents\Node.JS_workspace\webapp>git init
Initialized empty Git repository in C:/Users/A01041335/Documents/Node.JS_workspace/webapp/.git/

C:\Users\A01041335\Documents\Node.JS_workspace\webapp>git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        challenge.js
        challenge_blank.js
        node_modules/
        package-lock.json
        package.json
        public/
        server.js
        server.log
        views/
        weather_promises.js

nothing added to commit but untracked files present (use "git add" to track)

C:\Users\A01041335\Documents\Node.JS_workspace\webapp>
```

ADDING GIT TO OUR PROJECT

- Now, let's add first file to be tracked for changes:
  git add package.json
- And then let's run the *git status* again

```
C:\Users\A01041335\Documents\Node.JS_workspace\webapp>git add package.json
warning: LF will be replaced by CRLF in package.json.
The file will have its original line endings in your working directory.

C:\Users\A01041335\Documents\Node.JS_workspace\webapp>git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   package.json

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        challenge.js
        challenge_blank.js
        node_modules/
        package-lock.json
        public/
        server.js
        server.log
        views/
        weather_promises.js
```

ADDING GIT TO OUR PROJECT

- Finally, please add the following:
  public/
  views/
  package-lock.json
  server.js

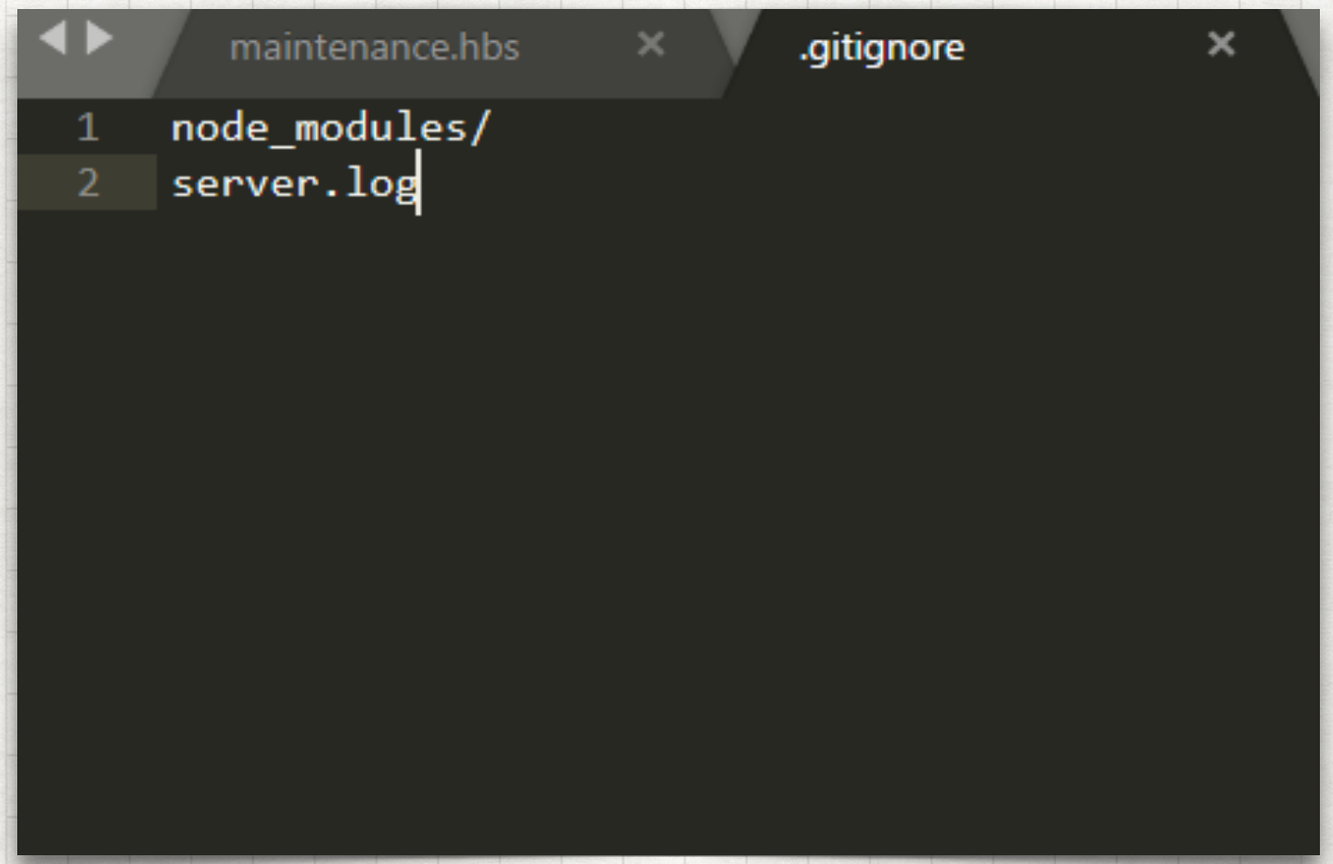
- The final result should be similar to the one on the right

```
hanges to be committed:
 (use "git rm --cached <file>..." to unstage)

       new file:   package-lock.json
       new file:   package.json
       new file:   public/about.html
       new file:   server.js
       new file:   views/about.hbs
       new file:   views/maintenance.hbs
       new file:   views/partials/footer.hbs

ntracked files:
 (use "git add <file>..." to include in what will be committed)

       challenge.js
       challenge_blank.js
       node_modules/
       server.log
       weather_promises.js
```

BREAKPOINT

GIT IGNORE

- Now, we don't want to share all files - for example node modules
- To prevent git to do this, we need to create a *.gitignore* file and put the ignored paths/ directories there
- To test if it works, please run the *git status* again
- We also want to add this file to repository, by using *git add .gitignore*



GIT COMMIT

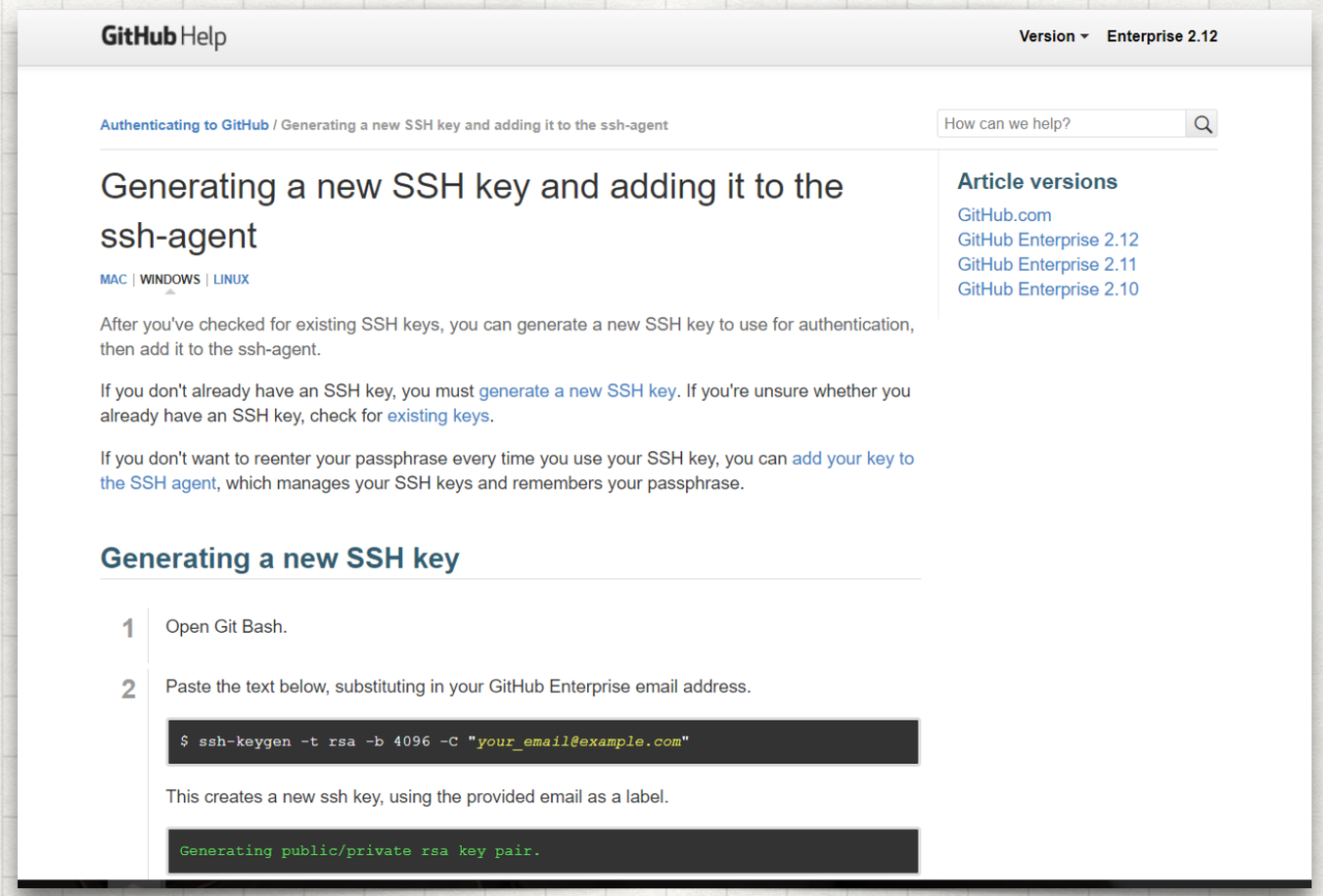
- A final step is to run: *git commit -m "Your message. Be very clear what you commit"*

```
C:\Users\A01041335\Documents\Node.JS_workspace\webapp>git commit -m "Initial commit"
[master (root-commit) 2a2ae6c] Initial commit
 Committer: Michal Aibin <A01041335@ad.bcit.ca>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 8 files changed, 1048 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 package-lock.json
 create mode 100644 package.json
 create mode 100644 public/about.html
 create mode 100644 server.js
 create mode 100644 views/about.hbs
 create mode 100644 views/maintenance.hbs
 create mode 100644 views/partials/footer.hbs

C:\Users\A01041335\Documents\Node.JS_workspace\webapp>
```

BREAKPOINT

CONNECTING TO GITHUB

- We first need to create a SSH keys to connect to GitHub and then add it to the *ssh-agent*
- All tutorials how to do it are here: https://help.github.com/enterprise/2.12/user/articles/connecting-to-github-with-ssh/
- We need to follow these steps: https://help.github.com/enterprise/2.12/user/articles/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent/



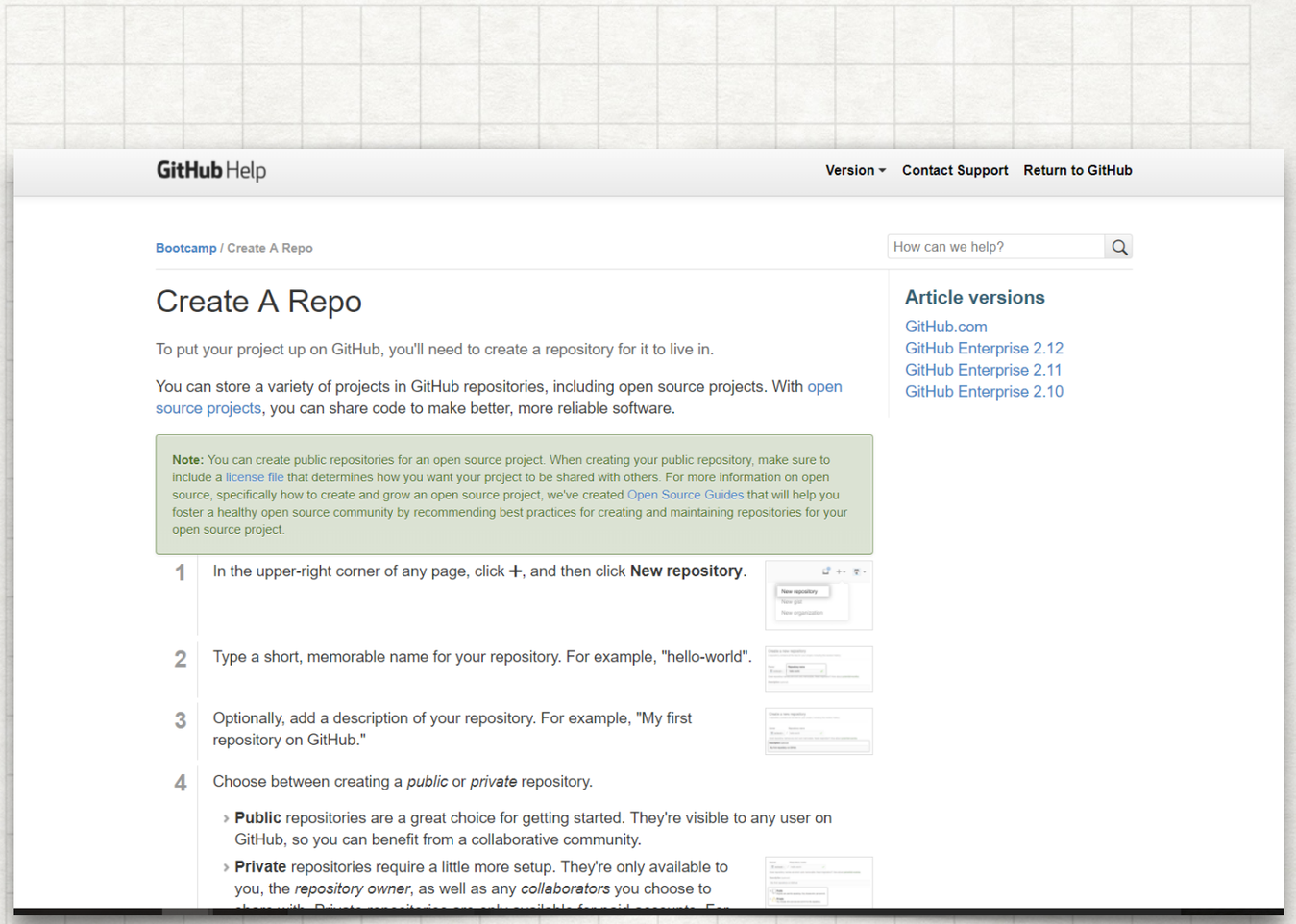
BREAKPOINT

CREATING A REPOSITORY

- After you finish, please check if everything went well:
  ssh -T git@github.com

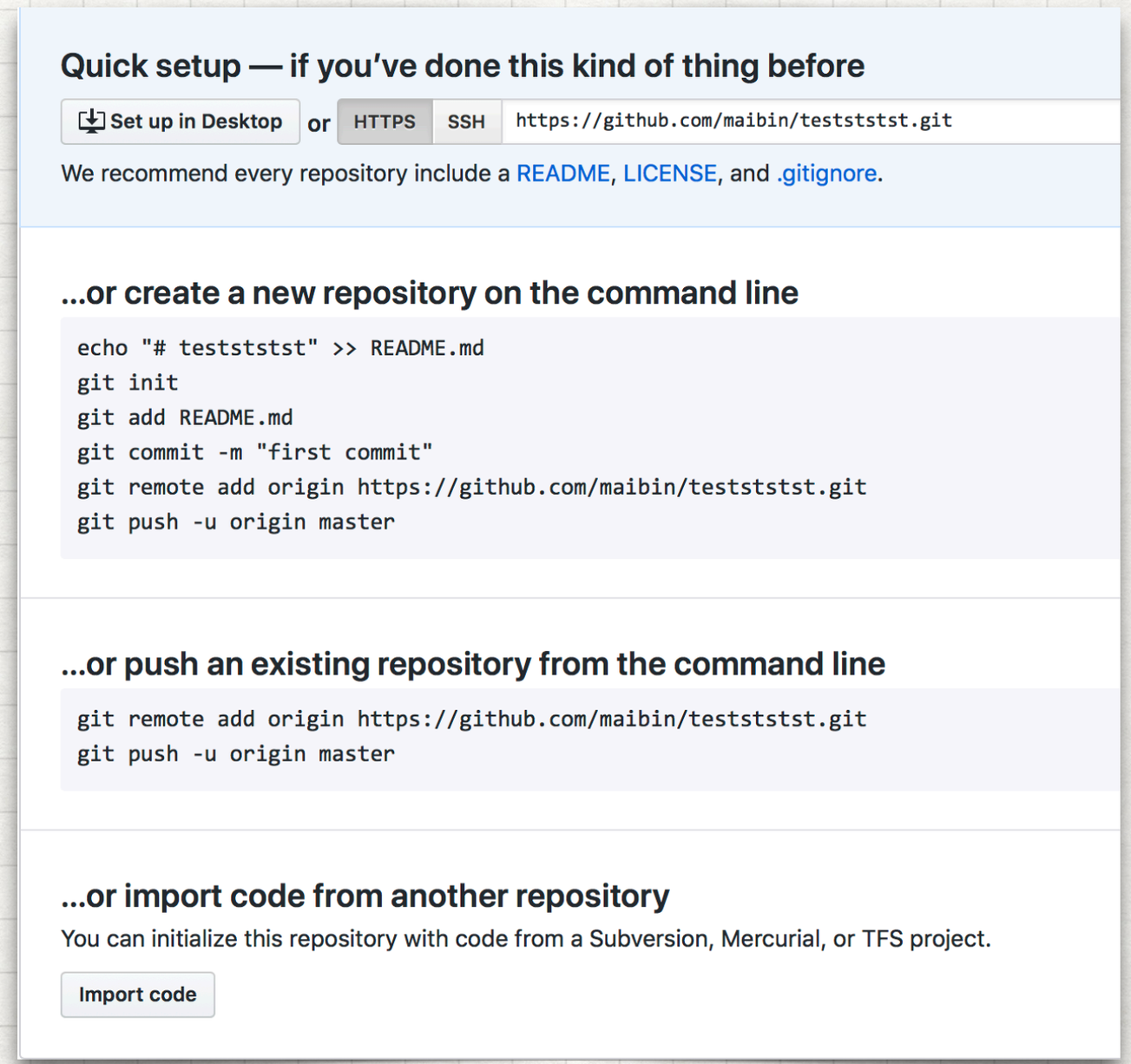
- Next step - create a repository:
  https://help.github.com/articles/create-a-repo/

- Very important note - please omit step #5, which means we don't want to create a README at this moment



CREATING A REPOSITORY

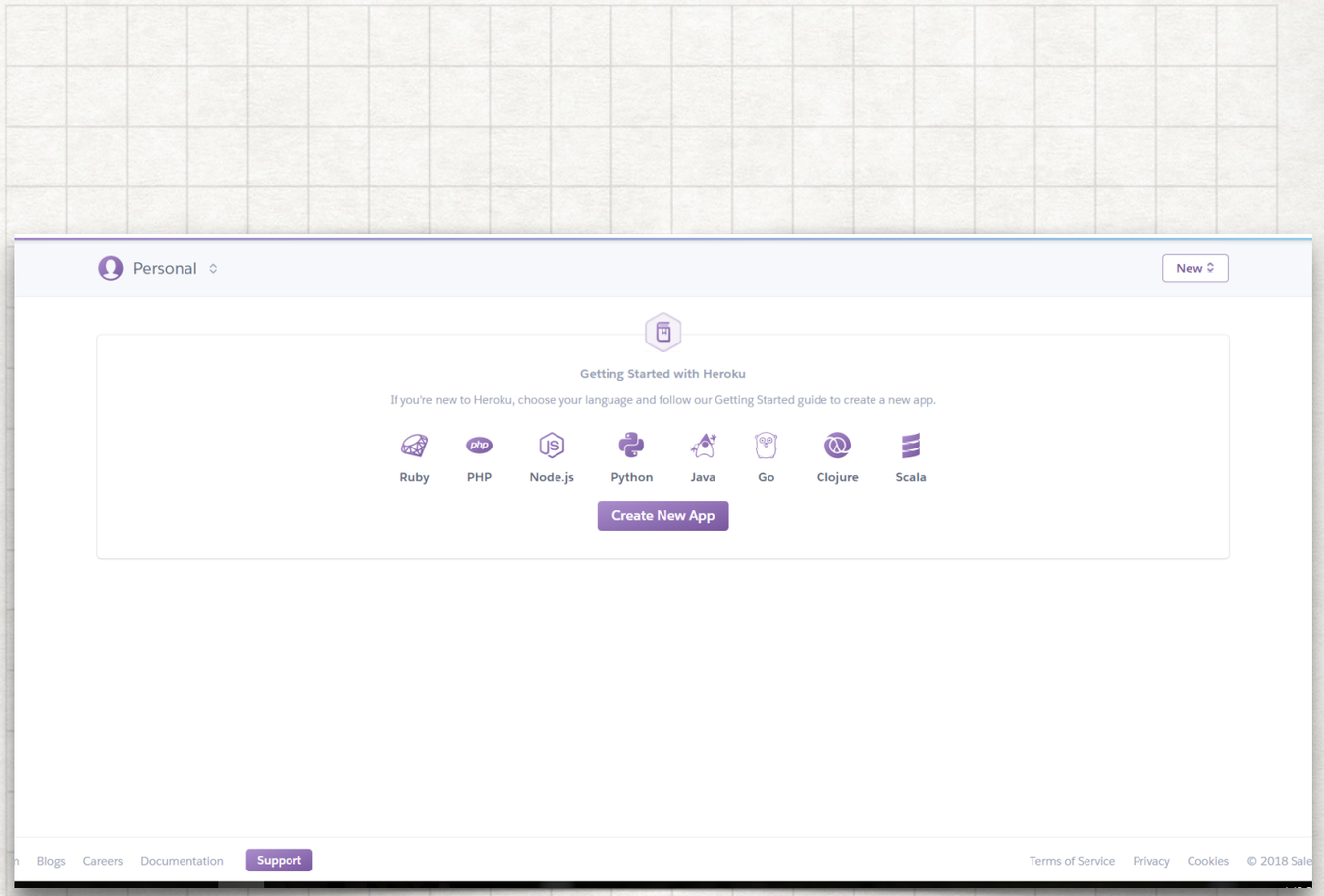
- If everything went well, you should see similar screen to the one presented on this slide
- Please keep save the presented information (e.g. notes) as you will need this in a second



BREAKPOINT

HEROKU

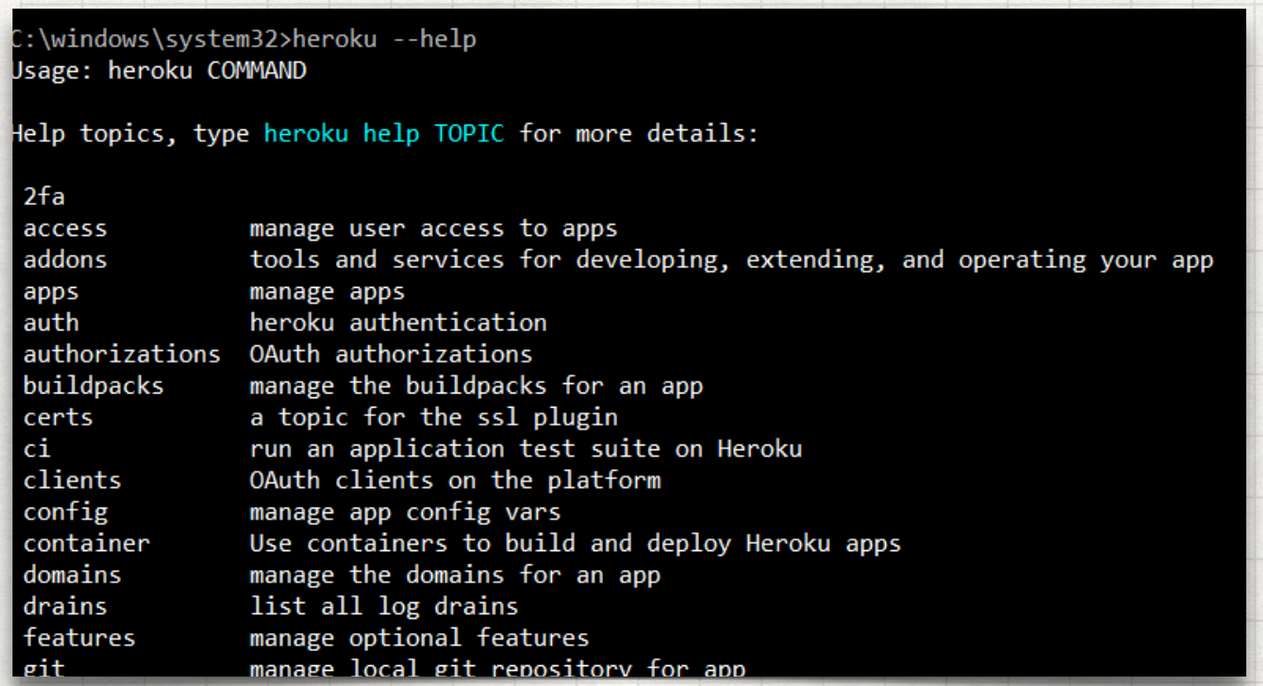
- Heroku is a platform that allows us to deploy web applications in the cloud.
- Go ahead and create a free account in here: https://www.heroku.com/
- Once finished, you should see something similar to the image on the right



BREAKPOINT

HEROKU

- Next, let's check if everything went well and display the manual of Heroku in the terminal:
  heroku --help
  Please note that initial execution of heroku might take some time to load all the information



HEROKU

- In the next step, we need to login, using the *heroku login* command

```
C:\windows\system32>heroku login
Enter your Heroku credentials:
Email: maibin@bcit.ca
Password: ********
Logged in as maibin@bcit.ca

C:\windows\system32>
```

HEROKU

- Next step, we need to add our SSH keys to heroku, as shown on the right

```
C:\windows\system32>heroku keys:add
Found an SSH public key at C:\Users\A01041335\.ssh\id_rsa.pub
? Would you like to upload it to Heroku? Yes
Uploading C:\Users\A01041335\.ssh\id_rsa.pub SSH key... done

C:\windows\system32>
```

HEROKU + GITHUB

- Now it's time to connect Heroku with GitHub

```
A01041335@CAS-Y25603 MINGW64 ~
$ ssh -v git@heroku.com
OpenSSH_7.5p1, OpenSSL 1.0.2k  26 Jan 2017
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: Connecting to heroku.com [50.19.85.154] port 22.
debug1: Connection established.
debug1: identity file /c/Users/A01041335/.ssh/id_rsa type 1
debug1: key_load_public: No such file or directory
debug1: identity file /c/Users/A01041335/.ssh/id_rsa-cert type -1
debug1: key_load_public: No such file or directory
debug1: identity file /c/Users/A01041335/.ssh/id_dsa type -1
debug1: key_load_public: No such file or directory
debug1: identity file /c/Users/A01041335/.ssh/id_dsa-cert type -1
debug1: key_load_public: No such file or directory
debug1: identity file /c/Users/A01041335/.ssh/id_ecdsa type -1
debug1: key_load_public: No such file or directory
debug1: identity file /c/Users/A01041335/.ssh/id_ecdsa-cert type -1
debug1: key_load_public: No such file or directory
debug1: identity file /c/Users/A01041335/.ssh/id_ed25519 type -1
debug1: key_load_public: No such file or directory
debug1: identity file /c/Users/A01041335/.ssh/id_ed25519-cert type -1
debug1: Enabling compatibility mode for protocol 2.0
debug1: Local version string SSH-2.0-OpenSSH_7.5
```

BREAKPOINT

MODIFY THE CODE

- To deploy our apps to the cloud, we need to do one change - we need to enable Heroku to decide on which port our app will be running

```
1   const express = require('express');
2   const hbs = require('hbs');
3   const fs = require('fs');
4
5   const port = process.env.PORT || 8080;
6
7   ///
8
9   app.listen(port, () => {
10      console.log(`Server is up on the port ${port}`);
11  });
```

MODIFY THE CODE

- Next, we need to add a startup script to our *package.json* file

```
{
  "name": "webapp",
  "version": "1.0.0",
  "description": "My First Webapp",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "start": "node server.js"
  },
  "author": "Michal Aibin",
  "license": "ISC",
  "dependencies": {
    "express": "^4.16.2",
    "hbs": "^4.0.1",
    "request": "^2.83.0"
  }
}

```

MODIFY THE CODE

- Now, let's check what will happen when we'll run the *git status* command

```
A01041335@CAS-Y25603 MINGW64 ~/Documents/Node.JS_workspace/webapp (master)
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   package.json
        modified:   server.js

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        challenge.js
        challenge_blank.js
        weather_promises.js

no changes added to commit (use "git add" and/or "git commit -a")
```

MODIFY THE CODE

- We need to add the modified files using *git add* command
- Then, you need to commit and push to remote repository:

git commit -m "Your msg"

git push origin master

```
A01041335@CAS-Y25603 MINGW64 ~/Documents/Node.JS_workspace/webapp (master)
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   package.json
        modified:   server.js

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        challenge.js
        challenge_blank.js
        weather_promises.js

no changes added to commit (use "git add" and/or "git commit -a")
```

CREATE A HEROKU APP

- Finally, we can create a Heroku web application

```
C:\Users\A01041335\Documents\Node.JS_workspace\webapp>heroku create
Creating app... !
 !    Invalid credentials provided.
Enter your Heroku credentials:
Email: maibin@bcit.ca
Password: ********
Creating app... done, stormy-refuge-41606
https://stormy-refuge-41606.herokuapp.com/ | https://git.heroku.com/stormy-refuge-41606.git

C:\Users\A01041335\Documents\Node.JS_workspace\webapp>
```

CREATE A HEROKU APP

- And push the the code from GitHub to Heroku

```
 !    Login is currently incompatible with git bash/Cygwin/MinGW

A01041335@CAS-Y25603 MINGW64 ~/Documents/Node.JS_workspace/webapp (master)
$ git push heroku
Counting objects: 17, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (14/14), done.
Writing objects: 100% (17/17), 9.93 KiB | 3.31 MiB/s, done.
Total 17 (delta 4), reused 0 (delta 0)
remote: Compressing source files... done.
remote: Building source:
remote:
remote: -----> Node.js app detected
remote:
remote: -----> Creating runtime environment
remote:
remote:        NPM_CONFIG_LOGLEVEL=error
remote:        NODE_VERBOSE=false
remote:        NODE_ENV=production
remote:        NODE_MODULES_CACHE=true
```

CREATE A HEROKU APP

- Now we can open our app in two ways:
  - go to dashboard, press on the webapp name and then choose *open app* (top right corner)
  - run it directly from the command line:

```
C:\Users\A01041335\Documents\Node.JS_workspace\webapp>heroku open

C:\Users\A01041335\Documents\Node.JS_workspace\webapp>
```

CHALLENGE #10 - PART II

- Modify your code (even simple changes are fine)
- git status, add, commit, push origin master, push heroku
- Open GitHub and create README (through GUI)
- Commit README from browser (using GUI)
- Then fetch and pull to your local computer
- Show me your final log message, e.g.:

```
A01041335@CAS-Y25603 MINGW64 ~/Documents/Node.JS_workspace/webapp (master)
$ git pull
Updating 4c069f6..cb322fb
Fast-forward
 README.md | 2 ++
 1 file changed, 2 insertions(+)
 create mode 100644 README.md
```