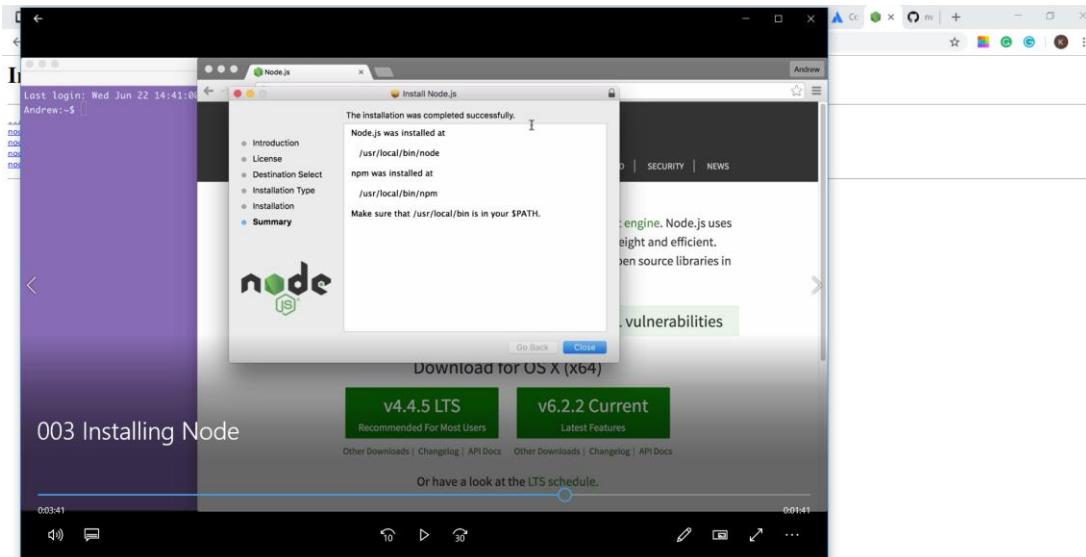


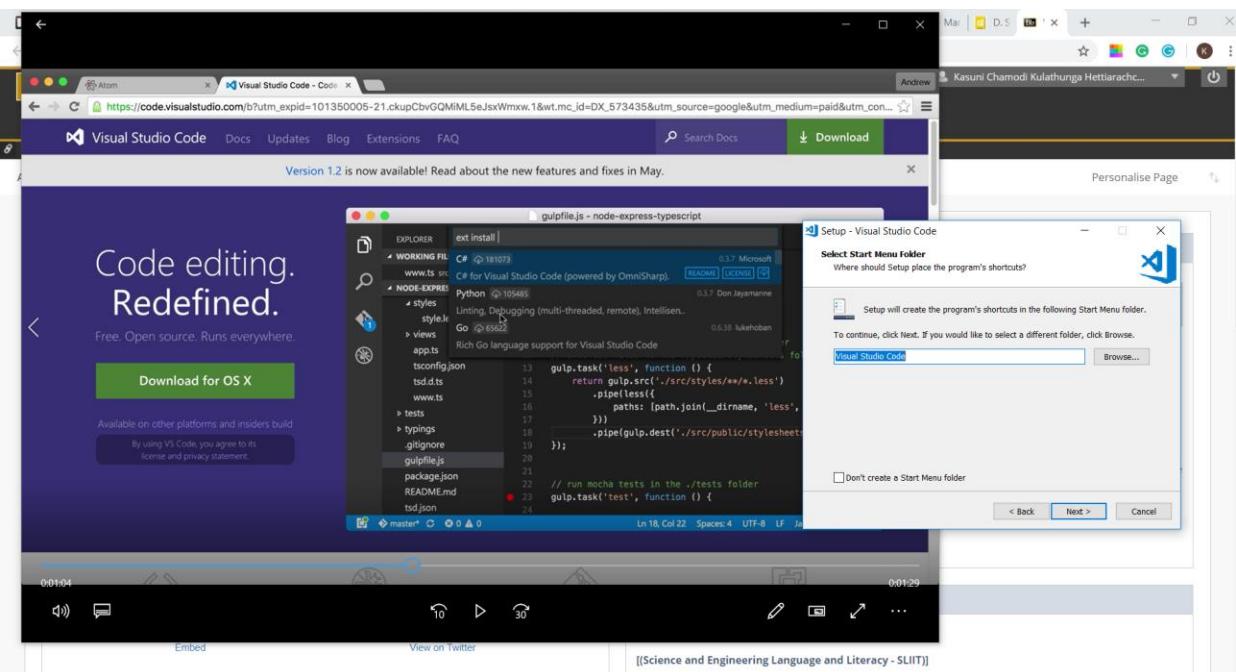
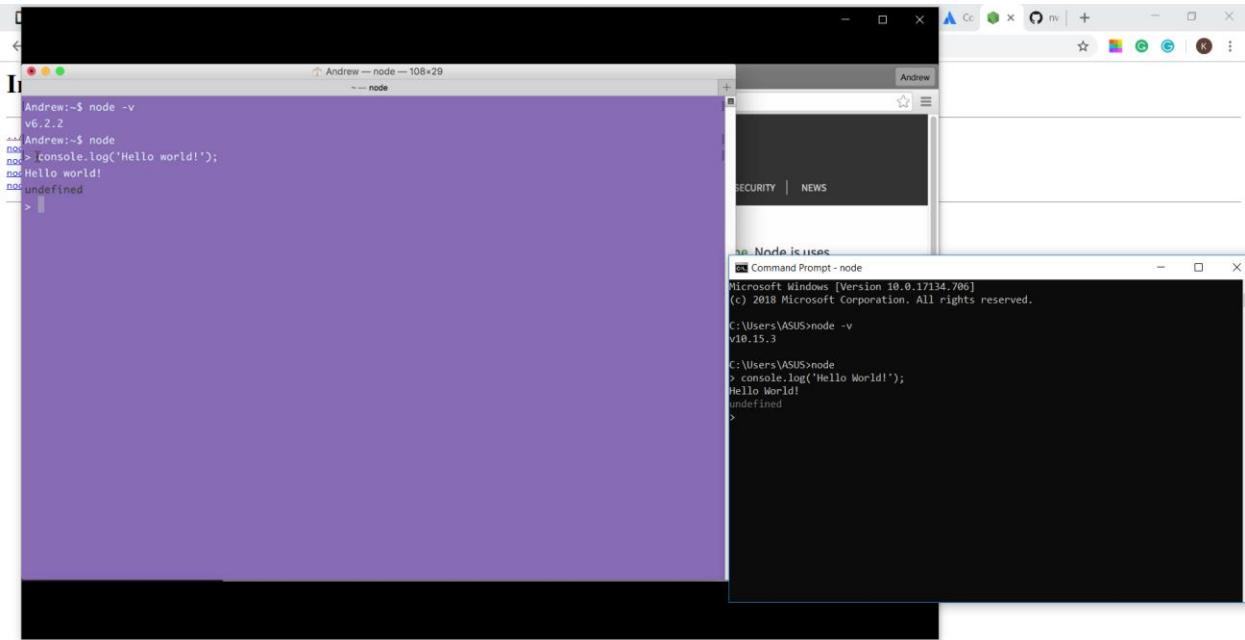
2) Installing Node

A screenshot of a Mac desktop showing three windows:

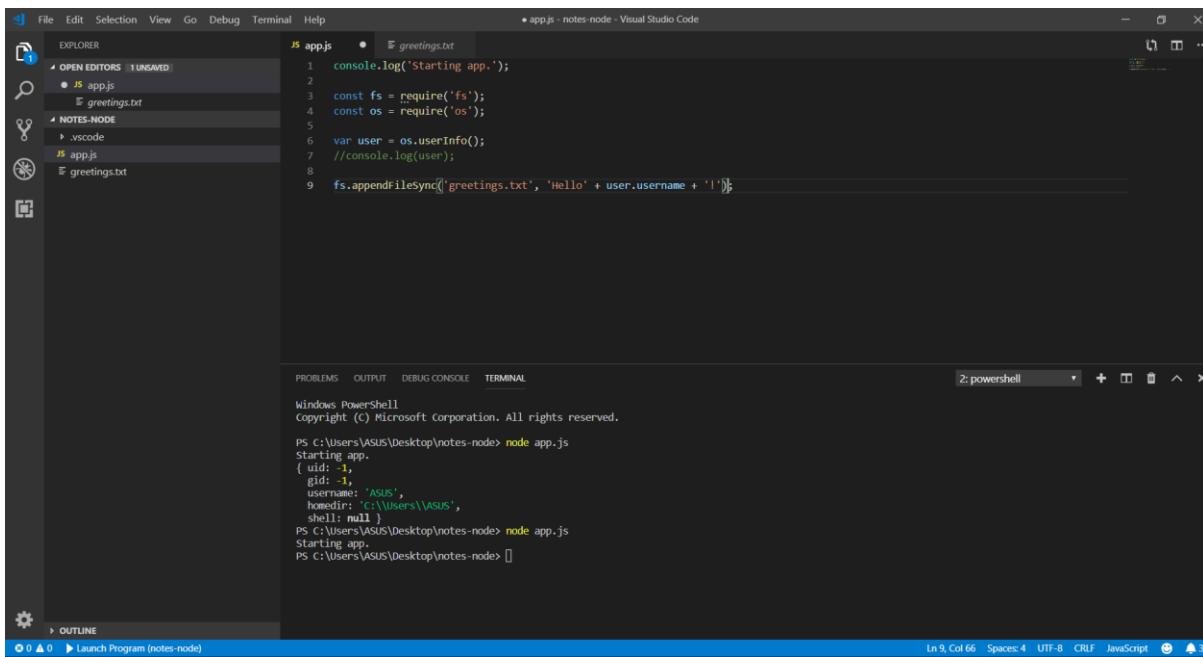
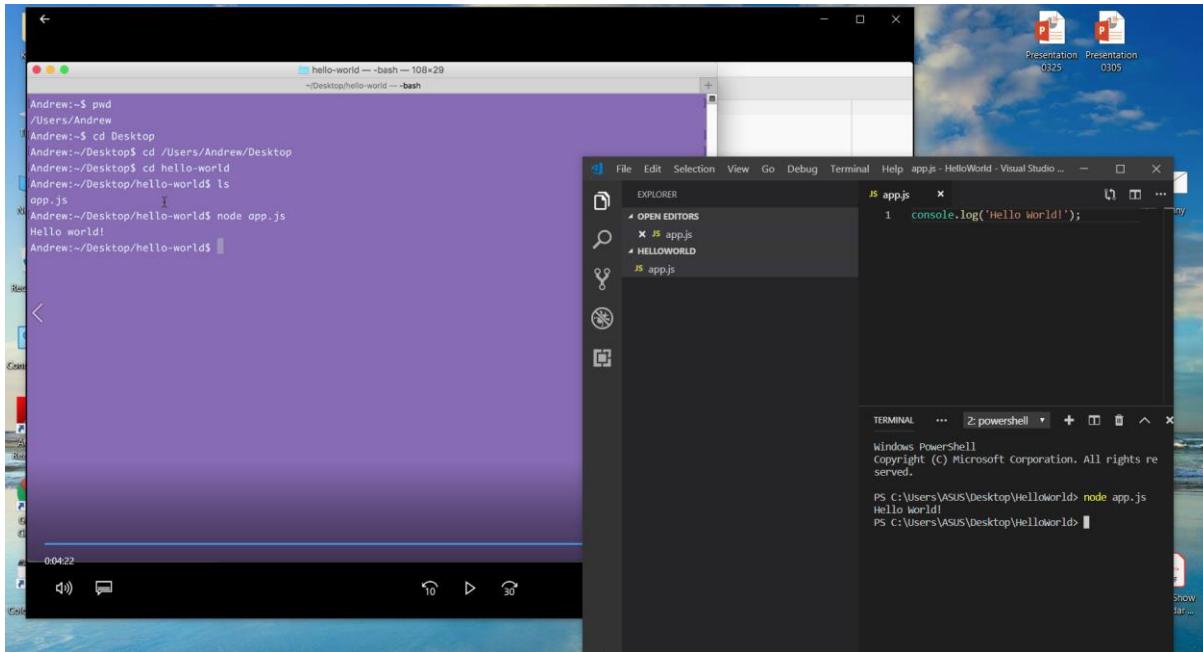
- Terminal:** Shows the command `node --version` being run, resulting in the output `v6.2.2`.
- Browser:** Shows a news article from BBC News about security.
- Command Prompt:** Shows the Node.js source code for the `process` module, specifically the `internal/promises` file.



What is Node? & Installing VS Code



Hello World! – first program



3) Using Require

The screenshot shows a Mac desktop environment. On the left, a terminal window displays the following session:

```
Last login: Tue Jun 28 09:35:45 on ttys000
Andrew:~ Desktop$ cd Desktop
Andrew:~/Desktop$ mkdir notes-node
Andrew:~/Desktop$ cd notes-node
Andrew:~/Desktop/notes-node$ node app.js
Starting app.
Andrew:~/Desktop/notes-node$ node app.js
Starting app.
{
  uid: 501,
  gid: 20,
  username: 'Andrew',
  homedir: '/Users/Andrew',
  shell: '/bin/bash'
}
Andrew:~/Desktop/notes-node$ node app.js
Starting app.
Andrew:~/Desktop/notes-node$
```

On the right, a Visual Studio Code window is open. It contains two files: `app.js` and `greetings.txt`. The `app.js` file has the following content:

```
1 console.log('Starting app.');
2
3 const fs = require('fs');
4 const os = require('os');
5
6 var user = os.userInfo();
7
8 fs.appendFile('greetings.txt', `Hello ${user.username}!`);
```

The `greetings.txt` file contains the text "Hello ASUS!". The terminal at the bottom of the VS Code interface shows the output of running the script:

```
PS C:\Users\ASUS\Desktop\notes-node> node app.js
Starting app.
{
  uid: -1,
  gid: -1,
  username: 'ASUS',
  homedir: 'C:\\\\Users\\\\ASUS',
  shell: null
}
PS C:\Users\ASUS\Desktop\notes-node> node app.js
Starting app.
PS C:\Users\ASUS\Desktop\notes-node>
```

The screenshot shows a Windows desktop environment. On the left, a terminal window displays the following session:

```
Last login: Tue Jun 28 09:35:45 on ttys000
Andrew:~ Desktop$ cd Desktop
Andrew:~/Desktop$ mkdir notes-node
Andrew:~/Desktop$ cd notes-node
Andrew:~/Desktop/notes-node$ node app.js
Starting app.
Andrew:~/Desktop/notes-node$ node app.js
Starting app.
{
  uid: 501,
  gid: 20,
  username: 'Andrew',
  homedir: '/Users/Andrew',
  shell: '/bin/bash'
}
Andrew:~/Desktop/notes-node$ node app.js
Starting app.
Andrew:~/Desktop/notes-node$
```

On the right, a Visual Studio Code window is open. It contains two files: `app.js` and `greetings.txt`. The `app.js` file has the following content:

```
1 console.log('Starting app.');
2
3 const fs = require('fs');
4 const os = require('os');
5
6 var user = os.userInfo();
7
8 fs.appendFile('greetings.txt', `Hello ${user.username}!`);
```

The `greetings.txt` file contains the text "Hello ASUS!". The terminal at the bottom of the VS Code interface shows the output of running the script:

```
PS C:\Users\ASUS\Desktop\notes-node> node app.js
Starting app.
{
  uid: -1,
  gid: -1,
  username: 'ASUS',
  homedir: 'C:\\\\Users\\\\ASUS',
  shell: null
}
PS C:\Users\ASUS\Desktop\notes-node> node app.js
Starting app.
PS C:\Users\ASUS\Desktop\notes-node> node app.js
Starting app.
```

Requiring your own files

The screenshot shows a Mac OS X desktop environment. On the left, a terminal window titled "notes-node" displays the output of running "node app.js". It shows the Node.js module system and the contents of "greetings.txt". On the right, another terminal window titled "dude" also shows the same output. In the center, a code editor (VS Code) has three tabs open: "app.js", "notes.js", and "greetings.txt". The "app.js" file contains code to read "notes.js" and append its contents to "greetings.txt". The "notes.js" file defines a function "addNote" that adds a note to an array. The "greetings.txt" file contains the initial greeting "Hello ASUS! You are 25.". The status bar at the bottom indicates the file is 9:78.

```
Starting notes.js
Module {
  id: '/Users/Andrew/Desktop/notes-node/app.js',
  exports: {},
  parent: null,
  filename: '/Users/Andrew/Desktop/notes-node/app.js',
  loaded: false,
  children: [ [Circular] ],
  paths: [
    '/Users/Andrew/Desktop/notes-node/app.js',
    '/Users/Andrew/Desktop/notes-node/app.js',
    '/Users/Andrew/node_modules',
    '/Users/node_modules'
  ],
  filename: '/Users/Andrew/Desktop/notes-node/app.js',
  loaded: false,
  children: [],
  paths: [
    '/Users/Andrew/Desktop/notes-node/app.js',
    '/Users/Andrew/Desktop/notes-node/app.js',
    '/Users/Andrew/node_modules',
    '/Users/node_modules'
  ]
}
Andrew:~/Desktop/notes-node$ node app.js
Starting app.js
Starting notes.js
Andrew:~/Desktop/notes-node$ 00:01
```

```
notes-node — bash — 108x29
notes-node — bash
Module {
  id: '/',
  exports: {},
  parent: null,
  filename: '/Users/Andrew/Desktop/notes-node/app.js',
  loaded: false,
  children: [ [Circular] ],
  paths: [
    '/Users/Andrew/Desktop/notes-node/node_modules',
    '/Users/Andrew/Desktop/node_modules',
    '/Users/Andrew/node_modules',
    '/Users/node_modules'
  ],
  filename: '/Users/Andrew/Desktop/notes-node/notes.js',
  loaded: false,
  children: [],
  paths: [
    '/Users/Andrew/Desktop/notes-node/node_modules',
    '/Users/Andrew/Desktop/node_modules',
    '/Users/Andrew/node_modules',
    '/Users/node_modules'
  ]
}
Andrew:~/Desktop/notes-node$ node app.js
Starting app.js
Starting notes.js
Andrew:~/Desktop/notes-node$ 00:01
```

```
Starting notes.js
Module {
  id: '/Users/Andrew/Desktop/notes-node/app.js',
  exports: {},
  parent: null,
  filename: '/Users/Andrew/Desktop/notes-node/app.js',
  loaded: false,
  children: [ [Circular] ],
  paths: [
    '/Users/Andrew/Desktop/notes-node/app.js',
    '/Users/Andrew/Desktop/notes-node/app.js',
    '/Users/Andrew/node_modules',
    '/Users/node_modules'
  ],
  filename: '/Users/Andrew/Desktop/notes-node/app.js',
  loaded: false,
  children: [],
  paths: [
    '/Users/Andrew/Desktop/notes-node/app.js',
    '/Users/Andrew/Desktop/notes-node/app.js',
    '/Users/Andrew/node_modules',
    '/Users/node_modules'
  ]
}
Andrew:~/Desktop/notes-node$ node app.js
Starting app.js
Starting notes.js
Andrew:~/Desktop/notes-node$ 00:01
```

```
notes-node — bash — 108x29
notes-node — bash
Module {
  id: '/',
  exports: {},
  parent: null,
  filename: '/Users/Andrew/Desktop/notes-node/app.js',
  loaded: false,
  children: [ [Circular] ],
  paths: [
    '/Users/Andrew/Desktop/notes-node/node_modules',
    '/Users/Andrew/Desktop/node_modules',
    '/Users/Andrew/node_modules',
    '/Users/node_modules'
  ],
  filename: '/Users/Andrew/Desktop/notes-node/notes.js',
  loaded: false,
  children: [],
  paths: [
    '/Users/Andrew/Desktop/notes-node/node_modules',
    '/Users/Andrew/Desktop/node_modules',
    '/Users/Andrew/node_modules',
    '/Users/node_modules'
  ]
}
Andrew:~/Desktop/notes-node$ node app.js
Starting app.js
Starting notes.js
Andrew:~/Desktop/notes-node$ 00:01
```

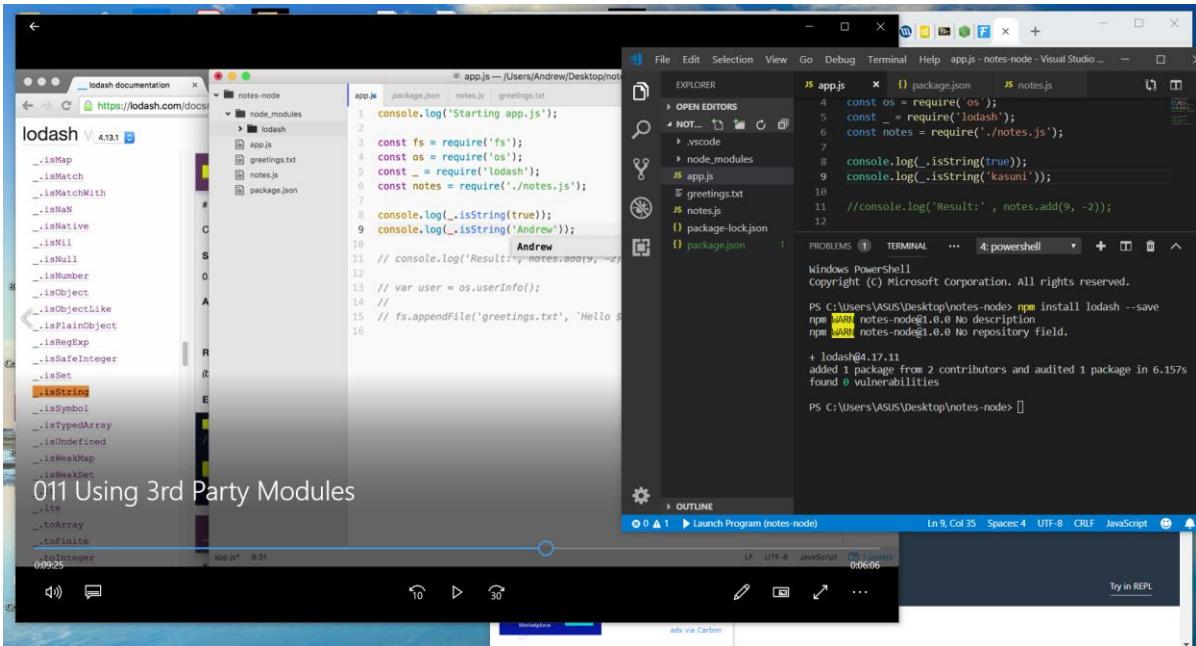
The screenshot shows a Windows 10 desktop environment. On the left, a terminal window titled "notes-node" displays the output of running "node app.js". It shows the Node.js module system and the contents of "greetings.txt". On the right, another terminal window titled "dude" also shows the same output. In the center, a code editor (VS Code) has three tabs open: "app.js", "notes.js", and "greetings.txt". The "app.js" file contains code to read "notes.js" and append its contents to "greetings.txt". The "notes.js" file defines a function "addNote" that adds a note to an array. The "greetings.txt" file contains the initial greeting "Hello ASUS! You are 25.". The status bar at the bottom indicates the file is 9:78.

```
Starting notes.js
Module {
  id: '/Users/Andrew/Desktop/notes-node/app.js',
  exports: {},
  parent: null,
  filename: '/Users/Andrew/Desktop/notes-node/app.js',
  loaded: false,
  children: [ [Circular] ],
  paths: [
    '/Users/Andrew/Desktop/notes-node/app.js',
    '/Users/Andrew/Desktop/notes-node/app.js',
    '/Users/Andrew/node_modules',
    '/Users/node_modules'
  ],
  filename: '/Users/Andrew/Desktop/notes-node/app.js',
  loaded: false,
  children: [],
  paths: [
    '/Users/Andrew/Desktop/notes-node/app.js',
    '/Users/Andrew/Desktop/notes-node/app.js',
    '/Users/Andrew/node_modules',
    '/Users/node_modules'
  ]
}
Andrew:~/Desktop/notes-node$ node app.js
Starting app.js
Starting notes.js
Andrew:~/Desktop/notes-node$ 00:01
```

```
notes-node — bash — 108x29
notes-node — bash
Module {
  id: '/',
  exports: {},
  parent: null,
  filename: '/Users/Andrew/Desktop/notes-node/app.js',
  loaded: false,
  children: [ [Circular] ],
  paths: [
    '/Users/Andrew/Desktop/notes-node/node_modules',
    '/Users/Andrew/Desktop/node_modules',
    '/Users/Andrew/node_modules',
    '/Users/node_modules'
  ],
  filename: '/Users/Andrew/Desktop/notes-node/notes.js',
  loaded: false,
  children: [],
  paths: [
    '/Users/Andrew/Desktop/notes-node/node_modules',
    '/Users/Andrew/Desktop/node_modules',
    '/Users/Andrew/node_modules',
    '/Users/node_modules'
  ]
}
Andrew:~/Desktop/notes-node$ node app.js
Starting app.js
Starting notes.js
Andrew:~/Desktop/notes-node$ 00:01
```

```
notes-node — bash — 108x29
notes-node — bash
Module {
  id: '/',
  exports: {},
  parent: null,
  filename: '/Users/Andrew/Desktop/notes-node/app.js',
  loaded: false,
  children: [ [Circular] ],
  paths: [
    '/Users/Andrew/Desktop/notes-node/node_modules',
    '/Users/Andrew/Desktop/node_modules',
    '/Users/Andrew/node_modules',
    '/Users/node_modules'
  ],
  filename: '/Users/Andrew/Desktop/notes-node/notes.js',
  loaded: false,
  children: [],
  paths: [
    '/Users/Andrew/Desktop/notes-node/node_modules',
    '/Users/Andrew/Desktop/node_modules',
    '/Users/Andrew/node_modules',
    '/Users/node_modules'
  ]
}
Andrew:~/Desktop/notes-node$ node app.js
Starting app.js
Starting notes.js
Andrew:~/Desktop/notes-node$ 00:01
```

Using 3rd party modules



This screenshot shows a dual-monitor setup. The left monitor displays a browser window for the lodash documentation (<https://lodash.com/docs>) with the search term 'notes'. The right monitor shows Visual Studio Code with an open project named 'notes-node'. The project structure includes files like package.json, app.js, notes.js, and greetings.txt. The code in app.js uses the lodash module to log strings and append them to a file. The terminal shows the command `npm install lodash` being run.

```
const os = require('os');
const _ = require('lodash');
const fs = require('fs');
const os = require('os');
const _ = require('lodash');
const notes = require('./notes.js');

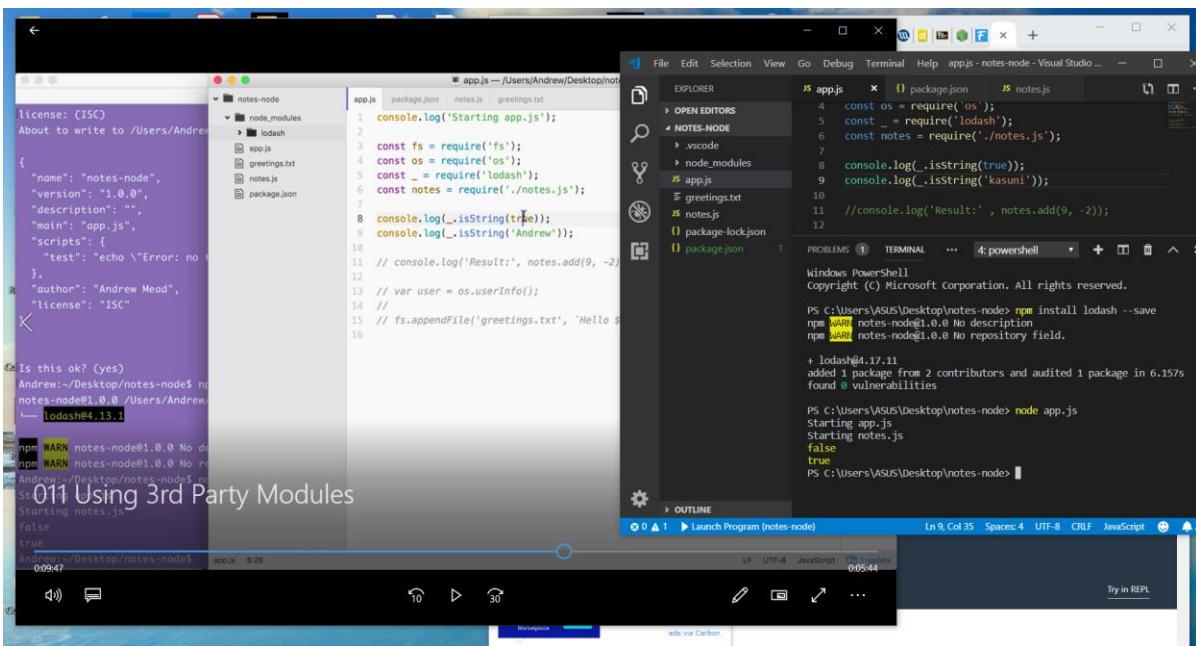
console.log(`Starting app.js`);

const user = os.userInfo();
const notes = [
    'Hello World',
    'Hello Kasumi'
].join('\n');

fs.appendFile('greetings.txt', notes);

console.log(`Result: ${notes}`);

```



This screenshot shows the continuation of the development process. The left monitor shows the terminal output of the application running, displaying the contents of the 'greetings.txt' file. The right monitor shows Visual Studio Code with the same project structure and code as the previous screenshot, but the terminal now shows the application's output: 'Hello World' and 'Hello Kasumi'.

```
Starting app.js
Starting notes.js
false
true
PS C:\Users\ASUS\Desktop\notes-node>
```

Restarting App with nodemon

The screenshot shows a Visual Studio Code interface with two tabs open: 'app.js' and 'notes.js'. The terminal window at the bottom displays the following log:

```
Andrew:~/Desktop/notes-node$ nodemon app.js
[nodemon] 1.9.2
[nodemon] to restart at any time, enter `rs`
[nodemon] watching: ***!
[nodemon] starting 'node app.js'
Starting app.js
Starting notes.js
[ 'Andrew', 1, 2, 3, 4 ]
[nodemon] clean exit - waiting
[nodemon] starting 'node app.js'
Starting app.js
Starting notes.js
[ 'kasumi', 1, 2, 3, 4 ]
[nodemon] clean exit - waiting for changes before restart
```

The screenshot shows a Visual Studio Code interface with two tabs open: 'app.js' and 'notes.js'. The terminal window at the bottom displays the following log:

```
Andrew:~/Desktop/notes-node$ nodemon app.js
[nodemon] 1.9.2
[nodemon] to restart at any time, enter `rs`
[nodemon] watching: ***!
[nodemon] starting 'node app.js'
Starting app.js
Starting notes.js
[ 'Andrew', 1, 2, 3, 4 ]
[nodemon] clean exit - waiting
[nodemon] restarting due to changes...
[nodemon] starting 'node app.js'
Starting app.js
Starting notes.js
[ 'Mike' ]
[nodemon] clean exit - waiting for changes before restart
```

Getting input from user

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows files: `app.js`, `notes.js`, `.vscode`, `node_modules`, `app.js`, `notes.js`, `package-lock.json`, and `package.json`.
- Code Editor:** Displays the `app.js` file content:1 console.log('Starting app.js');
2
3 const fs = require('fs');
4 const _ = require('lodash');
5
6 const notes = require('./notes.js');
7
8 var command = process.argv[2];
9 console.log(`Command: \${command}`);
10
11 if (command === 'add'){
12 console.log('Adding new note');
13 } else if (command === 'list'){
14 console.log('Listing all notes');
15 }
16 else {
17 console.log("error");
18 }
- Terminal:** Shows the execution of `node app.js list` and `node app.js add`.
- Status Bar:** Shows "Ln 18, Col 2" and "Spaces: 4" and "UTF-8".

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows files: `notes-node`, `app.js`, `notes.js`, `.vscode`, `node_modules`, `app.js`, `notes.js`, `package-lock.json`, and `package.json`.
- Code Editor:** Displays the `app.js` file content:1 console.log('Starting app.js');
2
3 const fs = require('fs');
4 const _ = require('lodash');
5
6 const notes = require('./notes.js');
7
8 var command = process.argv[2];
9 console.log(`Command: \${command}`);
10
11 if (command === 'add'){
12 console.log('Adding new note');
13 } else if (command === 'list'){
14 console.log('Listing all notes');
15 }
16 else if (command === 'read'){
17 console.log('Reading note');
18 } else if (command === 'remove'){
19 console.log('Removing note');
20 } else {
21 console.log('Command not recognized');
22 }
- Terminal:** Shows the execution of various commands: `node app.js list`, `node app.js add`, `node app.js read`, `node app.js remove`, and `node app.js rmd`.
- Status Bar:** Shows "Ln 24, Col 1" and "Spaces: 4" and "UTF-8".

Simplified input with yargs

The screenshot shows two terminal windows in Visual Studio Code. The top terminal window displays the output of running node app.js add multiple times with different arguments. The bottom terminal window shows the output of running node app.js read --title accounts, which triggers a TypeScript error because notes.getNote is not a function.

```

PS C:\Users\ASUS\Desktop\new-notes-node> node app.js add
Starting app.js
Starting notes.js
Command: add
process [ 'C:\Program Files\nodejs\node.exe',
  'C:\Users\ASUS\Desktop\new-notes-node\app.js',
  'add' ]
yargs { _: [ 'add' ], '$0': 'app.js' }
Adding new note
PS C:\Users\ASUS\Desktop\new-notes-node> node app.js add encrypted
Starting app.js
Starting notes.js
Command: add
process [ 'C:\Program Files\nodejs\node.exe',
  'C:\Users\ASUS\Desktop\new-notes-node\app.js',
  'add',
  'encrypted' ]
yargs { _: [ 'add', 'encrypted' ], '$0': 'app.js' }
Adding new note
PS C:\Users\ASUS\Desktop\new-notes-node> node app.js add --title=secrets
Starting app.js
Starting notes.js
Command: add
process [ 'C:\Program Files\nodejs\node.exe',
  'C:\Users\ASUS\Desktop\new-notes-node\app.js',
  'add',
  '--title=secrets' ]
yargs { _: [ 'add' ], title: 'secrets', '$0': 'app.js' }
Adding new note
PS C:\Users\ASUS\Desktop\new-notes-node>

PS C:\Users\ASUS\Desktop\new-notes-node>

```



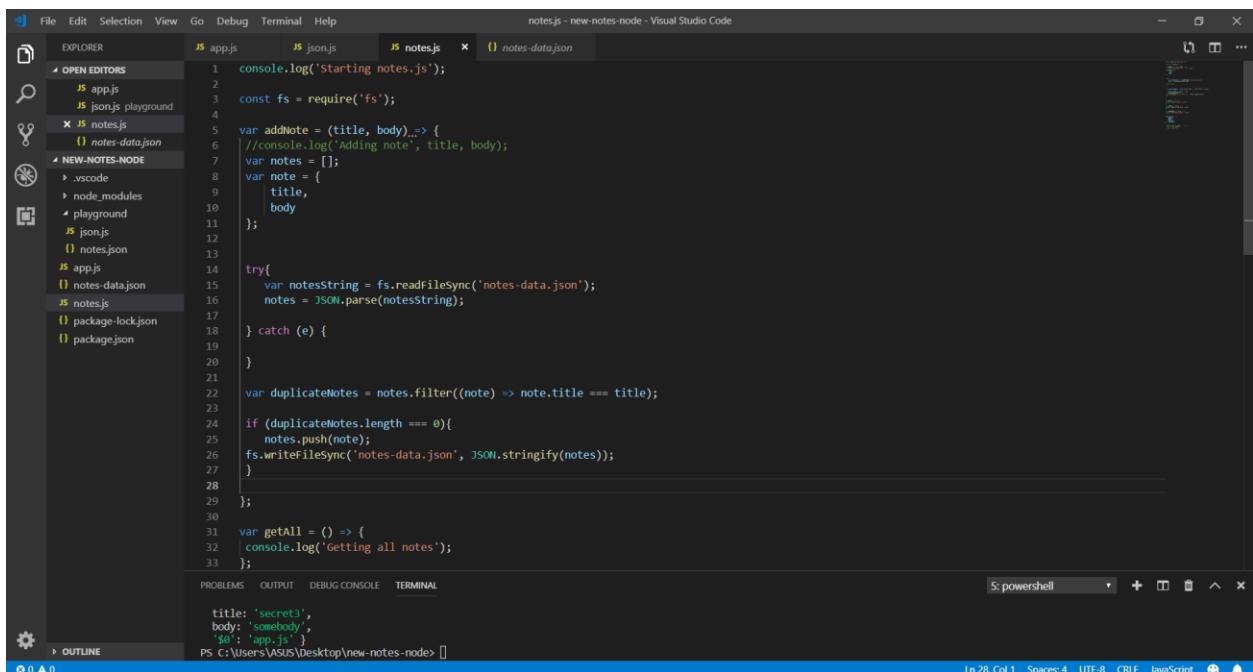
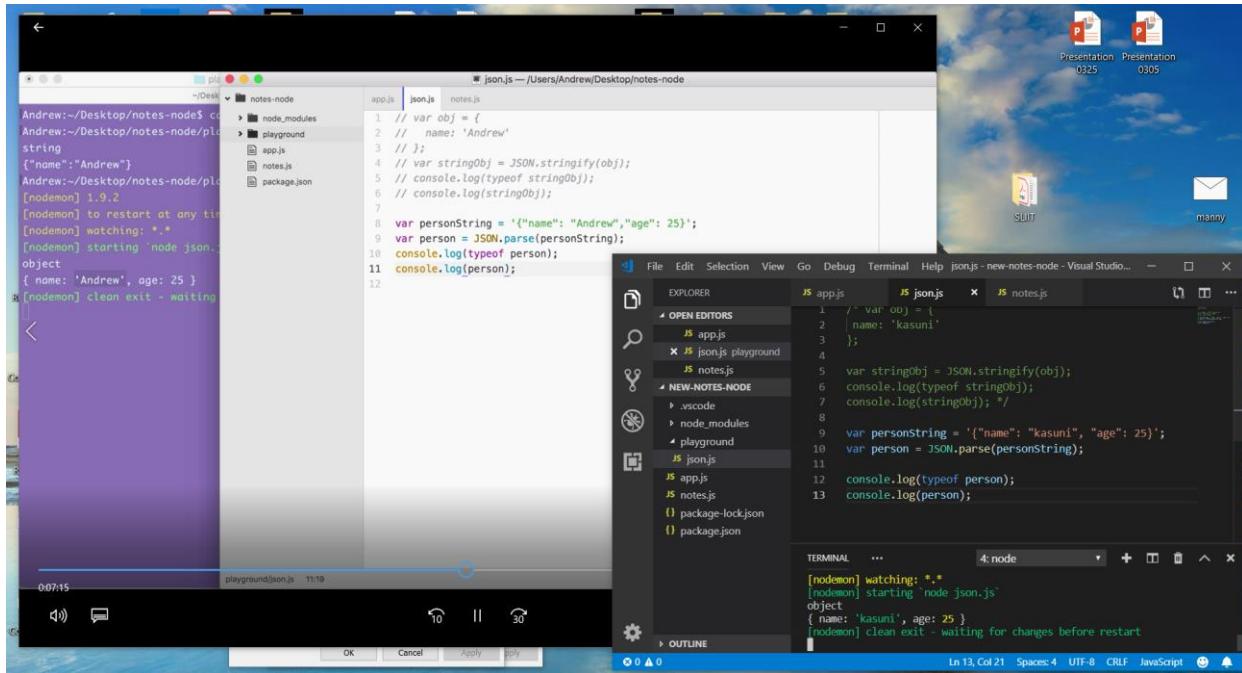
```

notes-node -- bash - 108x29
~/Desktop/notes-node -- bash

Yargs { _: [ 'list' ], '$0': 'app.js' }
Getting all notes
Andrew:~/Desktop/notes-node$ node app.js read --title accounts
Starting app.js
Starting notes.js
Command: read
Yargs { _: [ 'read' ], title: 'accounts', '$0': 'app.js' }
/Users/Andrew/Desktop/notes-node/app.js:19
  notes.getNote(argv.title);
  ^
TypeError: notes.getNote is not a function
  at Object.<anonymous> (/Users/Andrew/Desktop/notes-node/app.js:19:32)
  at Module._compile (module.js:541:32)
  at Object.Module._extensions..js (module.js:550:10)
  at Module.load (module.js:458:32)
  at tryModuleLoad (module.js:417:12)
  at Function.Module._load (module.js:409:3)
  at Module.runMain (module.js:575:10)
  at run (node.js:348:7)
  at startup (node.js:140:9)
  at node.js:63:3
Andrew:~/Desktop/notes-node$ node app.js read --title accounts
Starting app.js
Starting notes.js
Command: read
Yargs { _: [ 'read' ], title: 'accounts', '$0': 'app.js' }
Getting note accounts
Andrew:~/Desktop/notes-node$ 018 Removing a Note-subtitle-en.vtt

```

Working with JSON



Refactoring for reusability

```
File Edit Selection View Go Debug Terminal Help app.js - new-notes-node - Visual Studio Code

OPEN EDITORS
  app.js
  jsons
  notes.js
  notes-data.json
  NEW-NOTES-NODE
    vscode
    node_modules
    playground
    jsons
    notes.js
    package-lock.json
    package.json

  PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
  Starting app.js
  Starting notes.js
  Command: add
  args: [ 'add' ]
  title: 'to buy'
  body: 'somebody'
  $0: 'app.js'
  PS C:\Users\VGUS\Desktop\new-notes-node> node app.js add --title="to buy" --body="f
  ood"
  Starting app.js
  Starting notes.js
  Command: add
  args: [ 'add' ]
  title: 'to buy'
  body: 'food'
  $0: 'app.js'
  Note created
  Title: to buy
  Body: food
  PS C:\Users\VGUS\Desktop\new-notes-node>
```

Removing a note

```
File Edit Selection View Go Debug Terminal Help notes.js - new-notes-node - Visual Studio Code

OPEN EDITORS
  app.js
  jsons
  notes.js
  notes-data.json
  NEW-NOTES-NODE
    vscode
    node_modules
    playground
    jsons
    notes.js
    package-lock.json
    package.json

  PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
  Starting app.js
  Starting notes.js
  Command: remove
  args: [ 'remove' ]
  title: 'secret2'
  $0: 'app.js'
  Note removed
  PS C:\Users\VGUS\Desktop\new-notes-node> node app.js remove --title=secret2
  Starting app.js
  Starting notes.js
  Command: remove
  args: [ 'remove' ]
  title: 'secret2'
  $0: 'app.js'
  Note removed
  PS C:\Users\VGUS\Desktop\new-notes-node>
```

Reading notes and reusability

```
File Edit Selection View Go Debug Terminal Help app.js - new-notes-node - Visual Studio Code

OPEN EDITORS
  app.js
  jsons
  notes.js
  notes-data.json
  NEW-NOTES-NODE
    vscode
    node_modules
    playground
    jsons
    notes.js
    package-lock.json
    package.json

  PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
  Starting app.js
  Starting notes.js
  Command: add
  args: [ 'add' ]
  title: 'things to do'
  body: 'go to movies'
  $0: 'app.js'
  Note created
  Title: things to do
  Body: go to movies
  PS C:\Users\VGUS\Desktop\new-notes-node> node app.js add --title="things to do" --body="go to movies"
  Starting app.js
  Starting notes.js
  Command: add
  args: [ 'add' ]
  title: 'things to do'
  body: 'go to movies'
  $0: 'app.js'
  Note created
  Title: things to do
  Body: go to movies
  PS C:\Users\VGUS\Desktop\new-notes-node> node app.js add --title="things to do" --body="go to movies"
  Starting app.js
  Starting notes.js
  Command: add
  args: [ 'add' ]
  title: 'things to do'
  body: 'go to movies'
  $0: 'app.js'
  Note created
  Title: things to do
  Body: go to movies
  PS C:\Users\VGUS\Desktop\new-notes-node> node app.js add --title="things to do" --body="go to movies"
  PS C:\Users\VGUS\Desktop\new-notes-node>
```

Listing notes

The screenshot shows the Visual Studio Code interface with the terminal tab active. The terminal output shows the execution of `node app.js list`. The application logs the creation of a note titled 'Note created' and then lists all notes, which include 'secret', 'somebody', 'secret3', 'somebody', 'to buy', 'food', 'things to do', and 'things to do2'. The terminal also shows the command-line arguments passed to the script.

```
20 console.log('Note created');
21 notes.logNote(note);
22 /* console.log( .. );
23 console.log( title: ${note.title} );
24 console.log( body: ${note.body} ); */
25 } else {
26   console.log('Note title taken');
27 }
28 } else if (command === 'list'){
29   //console.log('listing all notes');
30   var allNotes = notes.getAll();
31   console.log(`Printing ${allNotes.length} Note(s).`);
32   allNotes.forEach((note) => notes.logNote(note));
33 }

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Note created
--
Title: things to do2
Body: go to movies
PS C:\Users\ASUS\Desktop\new-notes-node>
PS C:\Users\ASUS\Desktop\new-notes-node> node app.js list
Starting app.js
Starting notes.js
Command: list
args [ '_' [ 'list' ], '$0': 'app.js' ]
Printing 5 Note(s).
--
Title: secret
Body: somebody
--
Title: secret3
Body: somebody
--
Title: to buy
Body: food
--
Title: things to do
Body: undefined
--
Title: things to do2
Body: go to movies
PS C:\Users\ASUS\Desktop\new-notes-node>
```

Requiring arguments and advanced yargs

The screenshot shows the Visual Studio Code interface with the terminal tab active. The terminal output shows the execution of `node app.js remove`. The application prompts for a note title and then removes it. The command-line arguments are shown at the bottom of the terminal.

```
10 describe: 'title or note',
11 demand: true,
12 alias: 't'
13 };
14
15 const bodyOptions = {
16   describe: 'Body of note',
17   demand: true,
18   alias: 'b'
19 };
20
21 //const argv = yargs.argv;
22 const argv = yargs
23 .command('add', 'Add a new note', {
24   title: titleOptions,
25   body: bodyOptions
26 })
27 .command('list', 'List all notes')
28 .command('read', 'Read a note', {
29   title: titleOptions,
30 })
31 .command('remove', 'Remove a note', {
32   title: titleOptions
33 })
34 .help()
35 .argv;

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Starting app.js
Starting notes.js
app.js remove

Remove a note
Options:
  --version Show version number          [boolean]  --help     Show help
                                                [boolean]  --title   Title of note
Missing required argument: title
PS C:\Users\ASUS\Desktop\new-notes-node>
```

Arrow functions

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows files like `app.js`, `arrow-function.js`, and `notes.js`.
- Editor:** Displays the code for `arrow-function.js`:


```
4  }; */
5
6 var square = x => x * x;
7 console.log(square(9));
8
9 var user = {
10   name: 'kasuni',
11   sayHi: () => {
12     console.log(arguments);
13     console.log(`Hi. I'm ${this.name}`);
14   },
15   sayHiAlt () {
16     console.log(arguments);
17     console.log(`Hi. I'm ${this.name}`);
18   }
19 };
20 user.sayHi(1,2,3);
```
- Terminal:** Shows the output of running the script:


```
'c:\Users\ASUS\Desktop\new-notes-node\playground\arrow-function.js',
loaded: false,
children: [],
paths:
[ 'C:\Users\ASUS\Desktop\new-notes-node\playground\node_modules',
  'C:\Users\ASUS\Desktop\new-notes-node\node_modules',
  'C:\Users\ASUS\node_modules',
  'C:\Users\node_modules',
  'C:\node_modules' ],
{
  '3': 'C:\Users\ASUS\Desktop\new-notes-node\playground\arrow-function.js',
  '4': 'C:\Users\ASUS\Desktop\new-notes-node\playground'
}
Hi. I'm undefined
[nodemon] clean exit - waiting for changes before restart
```
- Status Bar:** Shows "In 14, Col 7" and "JavaScript".

4) Async Basics, Call stack & event loop, Callback functions & APIs and Pretty printing objects

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows files like `async-basics.js`, `callbacks.js`, and `app.js`.
- Editor:** Displays the code for `app.js`:


```
3 request({
4   url: 'https://maps.googleapis.com/maps/api/geocode/json?address=1%20lombard%20streets%20san%20francisco%20ca',
5   json: true
6 }, (error, response, body) => {
7   console.log(JSON.stringify(body, undefined, 2));
8 });
```
- Terminal:** Shows the output of running the script:


```
run 'npm audit fix' to fix them, or 'npm audit' for details
PS C:\Users\ASUS\Desktop\weather-app\playground> node app.js
internal/modules/cjs/loader.js:582:15
throw err;
^

Error: Cannot find module 'C:\Users\ASUS\Desktop\weather-app\playground\app.js'
at Function.Module._resolveFilename (internal/modules/cjs/loader.js:582:15)
at Function.Module._load (internal/modules/cjs/loader.js:580:25)
at Function.Module.runMain (internal/modules/cjs/loader.js:754:12)
at startup (internal/bootstrap/node.js:283:19)
at bootstrapNodeJSCore (internal/bootstrap/node.js:622:3)
PS C:\Users\ASUS\Desktop\weather-app\playground>
```
- Status Bar:** Shows "In 8, Col 4" and "JavaScript".

What's makes up an HTTP request & Encoding user input

The screenshot shows the Visual Studio Code interface with two tabs open: 'app.js' and 'callbacks.js'. The 'app.js' tab contains the following code:

```
const request = require('request');
const yargs = require('yargs');

const argv = yargs
    .request({
        url: 'https://maps.googleapis.com/maps/api/geocode/json?address='
    })
    .json(true)
    .exec((error, response, body) => {
        console.log(`Address: ${body.results[0].formatted_address}`);
        console.log(`Latitude: ${body.results[0].geometry.location.lat}`);
        console.log(`Longitude: ${body.results[0].geometry.location.lng}`);
    });

module.exports = argv;
```

The 'callbacks.js' tab is currently unsaved. The Explorer sidebar shows the project structure under 'weather-app' with files like 'node_modules', 'playground', 'app.js', and 'package.json'. The terminal shows npm warnings about missing dependencies.

The screenshot shows the Visual Studio Code interface with the same file structure as the previous screenshot. The 'app.js' tab now includes command-line argument handling and address encoding:

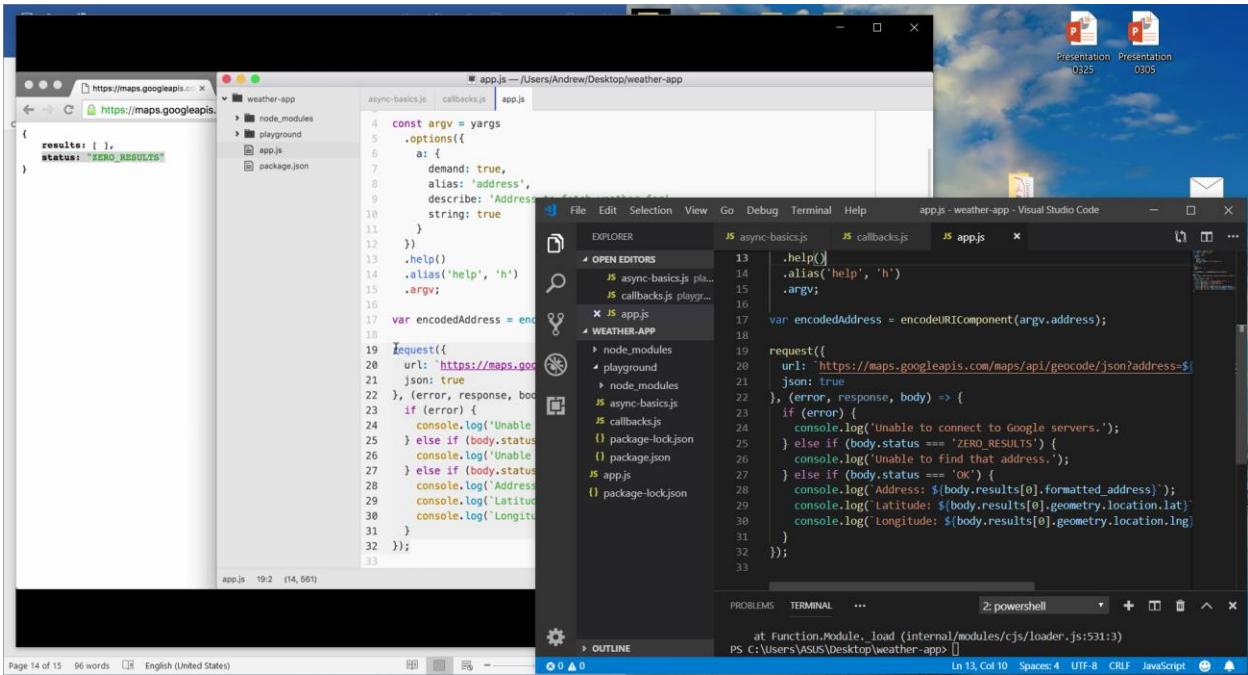
```
const argv = yargs
    .options({
        a: {
            demand: true,
            alias: 'address',
            describe: 'Address',
            string: true
        }
    })
    .help()
    .alias('help', 'h')
    .argv;

var encodedAddress = encodeURI(argv.address);

request({
    url: 'https://maps.googleapis.com/maps/api/geocode/json?address=' + encodedAddress
})
.json(true)
.exec((error, response, body) => {
    if (error) {
        console.log('unable to connect to Google servers.');
    } else if (body.status === 'ZERO_RESULTS') {
        console.log('unable to find that address.');
    } else if (body.status === 'OK') {
        console.log(`Address: ${body.results[0].formatted_address}`);
        console.log(`Latitude: ${body.results[0].geometry.location.lat}`);
        console.log(`Longitude: ${body.results[0].geometry.location.lng}`);
    }
});
```

The terminal shows the application running and printing address details to the console. A search bar at the top right is set to 'Search n-4-6-http'.

Callback errors



A screenshot of Visual Studio Code showing an error in the file `app.js`. The error message is "at Function.Module._load (internal/modules/cjs/loader.js:531:3)". The code in `app.js` is as follows:

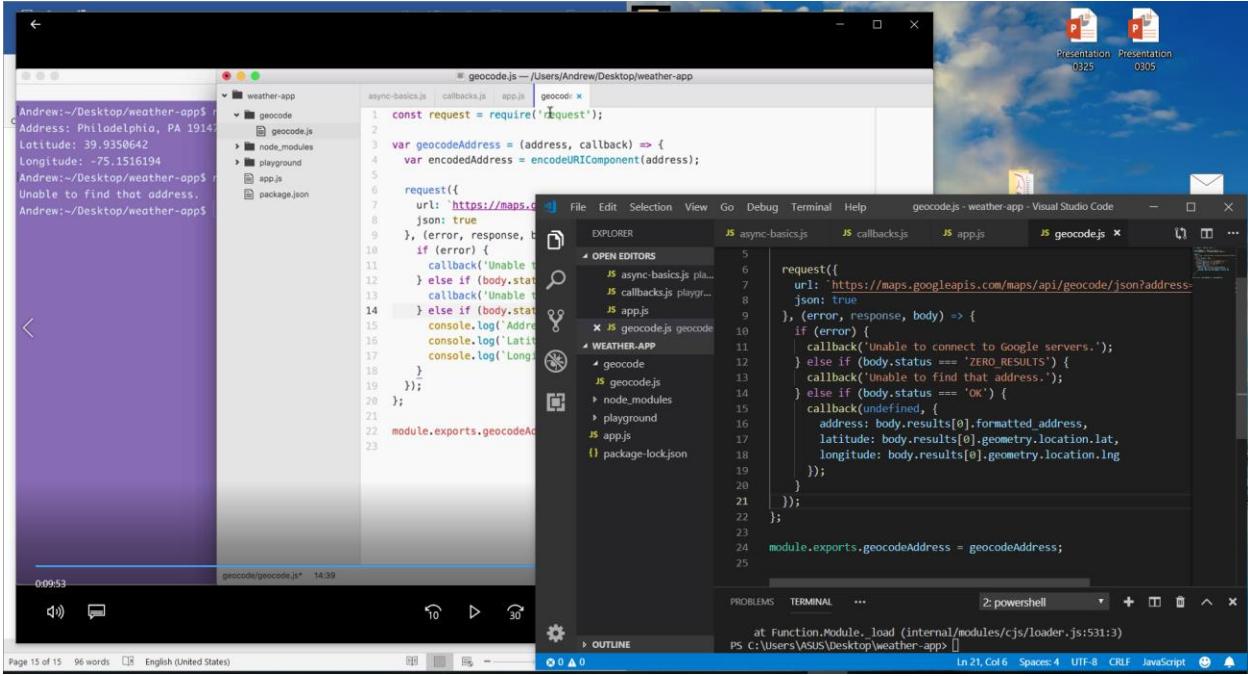
```
const argv = yargs
    .options({
        a: {
            demand: true,
            alias: 'address',
            describe: 'Address',
            string: true
        }
    })
    .help()
    .alias('help', 'h')
    .argv;

var encodedAddress = encodeURI(argv.address);

request({
    url: 'https://maps.googleapis.com/maps/api/geocode/json?address=' + encodedAddress,
    json: true
}, (error, response, body) => {
    if (error) {
        console.log('Unable to connect to Google servers.');
    } else if (body.status === 'ZERO_RESULTS') {
        console.log('Unable to find that address.');
    } else if (body.status === 'OK') {
        console.log(`Address: ${body.results[0].formatted_address}`);
        console.log(`Latitude: ${body.results[0].geometry.location.lat}`);
        console.log(`Longitude: ${body.results[0].geometry.location.lng}`);
    }
});
```

The cursor is at the end of the line `});`. The code editor has tabs for `async-basics.js`, `callbacks.js`, and `app.js`.

Abstracting callbacks



A screenshot of Visual Studio Code showing a refactored file named `geocode.js`. The code is as follows:

```
const request = require('request');

var geocodeAddress = (address, callback) => {
    var encodedAddress = encodeURIComponent(address);

    request({
        url: 'https://maps.googleapis.com/maps/api/geocode/json?address=' + encodedAddress,
        json: true
    }, (error, response, body) => {
        if (error) {
            callback('Unable to connect to Google servers.');
        } else if (body.status === 'ZERO_RESULTS') {
            callback('Unable to find that address.');
        } else if (body.status === 'OK') {
            console.log(`Address: ${body.results[0].formatted_address}`);
            console.log(`Latitude: ${body.results[0].geometry.location.lat}`);
            console.log(`Longitude: ${body.results[0].geometry.location.lng}`);
        }
    });
};

module.exports.geocodeAddress = geocodeAddress;
```

The code editor has tabs for `async-basics.js`, `callbacks.js`, `app.js`, and `geocode.js`. The terminal shows the output of running the application, which includes the address and coordinates of Philadelphia, PA.

Wiring up weather search

The screenshot shows a Windows desktop environment with two terminal windows and two code editors in Visual Studio Code.

Terminal 1 (Left):

```
Andrew:~/Desktop/weather-app$ node app.js Philadelphia, PA
{
  "address": "Philadelphia, PA",
  "latitude": 39.9396284,
  "longitude": -75.18663959999999
}
Andrew:~/Desktop/weather-app$ 
85.27
Andrew:~/Desktop/weather-app$ 
85.28
Andrew:~/Desktop/weather-app$ 
Unable to fetch weather.
Andrew:~/Desktop/weather-app$ 
Unable to connect to Forecast.io server.
Andrew:~/Desktop/weather-app$ 
```

Terminal 2 (Right):

```
File Edit Selection View Go Debug Terminal Help app.js - weather-app - Visual Studio Code
EXPLORER JS app.js JS callbacks.js JS geocode.js JS geocoding.js
OPEN EDITORS JS app.js JS callbacks.js JS geocode.js JS geocoding.js
WEATHER-APP JS app.js JS geocode.js JS node_modules JS playground JS package-lock.json
OUTLINE PROBLEMS TERMINAL ... 2:powershell
at Function.Module._load (internal/modules/cjs/loader.js:531:3)
PS C:\Users\ASUS\Desktop\weather-app> 
```

Code Editors:

- Left Editor (app.js):**

```
const request = require('request');
request({
  url: 'https://api.forecast.io/forecast/4a04d1c42fd9d32c97a2c291a3',
  json: true
}, (error, response, body) => {
  if (error) {
    console.log('Unable to connect to Forecast.io server.');
  } else if (response.statusCode === 400) {
    console.log('Unable to fetch weather.');
  } else if (response.statusCode === 200) {
    console.log(body.currently.temperature);
  }
});
```
- Right Editor (geocode.js):**

```
// ...
// .help()
// .alias('help', 'h')
// .argv;
// ...
// geocode.geocodeAddress(argv.address, (errorMessage, results) => {
//   if (errorMessage) {
//     console.log(errorMessage);
//   } else {
//     console.log(JSON.stringify(results, undefined, 2));
//   }
// });
const request = require('request');

request({
  url: 'https://api.forecast.io/forecast/4a04d1c42fd9d32c97a2c291a3',
  json: true
}, (error, response, body) => {
  if (error) {
    console.log('Unable to connect to Forecast.io server.');
  } else if (response.statusCode === 400) {
    console.log('Unable to fetch weather.');
  } else if (response.statusCode === 200) {
    console.log(body.currently.temperature);
  }
});
```

Chaining callbacks together

The screenshot shows a Windows desktop environment with two terminal windows and two code editors in Visual Studio Code.

Terminal 1 (Left):

```
Andrew:~/Desktop/weather-app$ node app.js
80.52
Andrew:~/Desktop/weather-app$ 
{
  "temperature": 81.61,
  "apparentTemperature": 87.02
}
Andrew:~/Desktop/weather-app$ 
Flemington, NJ 08822, USA
It's currently 81.56. It feels like 87.02.
Andrew:~/Desktop/weather-app$ 
Key West, FL 33040, USA
It's currently 84.17. It feels like 84.17.
Andrew:~/Desktop/weather-app$ 
```

Terminal 2 (Right):

```
File Edit Selection View Go Debug Terminal Help geocode.js - weather-app - Visual Studio Code
EXPLORER JS app.js JS weather.js JS geocode.js JS geocoding.js
OPEN EDITORS JS app.js JS weather.js JS geocode.js JS geocoding.js
WEATHER-APP JS app.js JS weather.js JS node_modules JS playground JS package-lock.json
OUTLINE PROBLEMS TERMINAL ... 2:powershell
at Function.Module._load (internal/modules/cjs/loader.js:531:3)
PS C:\Users\ASUS\Desktop\weather-app> 
```

Code Editors:

- Left Editor (app.js):**

```
const weather = require('./weather/weather');
const argv = yargs
  .options({
    a: {
      demand: true,
      alias: 'address',
      describe: 'Address to geocode',
      string: true
    }
  })
  .help()
  .alias('help', 'h')
  .argv;

geocode.geocodeAddress(argv.address, (errorMessage, results) => {
  if (errorMessage) {
    console.log(errorMessage);
  } else {
    console.log(results);
    weather.getWeatherForAddress(argv.address, (error, weather) => {
      if (error) {
        console.log(error);
      } else {
        console.log(`It's currently ${weather.currently.temperature}. It feels like ${weather.currently.apparentTemperature}.`);
      }
    });
  }
});
```
- Right Editor (geocode.js):**

```
const request = require('request');
request({
  url: 'https://maps.googleapis.com/maps/api/geocode/json?address=' + encodeURIComponent(argv.address),
  json: true
}, (error, response, body) => {
  if (error) {
    callback('Unable to connect to Google servers.');
  } else if (body.status === 'ZERO_RESULTS') {
    callback('Unable to find that address.');
  } else if (body.status === 'OK') {
    callback(undefined, {
      address: body.results[0].formatted_address,
      latitude: body.results[0].geometry.location.lat,
      longitude: body.results[0].geometry.location.lng
    });
  }
});

module.exports.geocodeAddress = geocodeAddress;
```

Intro to ES6 promises

A screenshot of a Windows desktop showing two instances of Visual Studio Code. The left instance is running a Node.js application named 'weather-app' with a terminal window displaying log messages from nodemon. The right instance shows code for a geocoding application. The terminal window in the right instance shows a promise being resolved with the message 'Hey, It worked!'.

```
[nodemon] to restart at any time
[nodemon] watching: ***!
[nodemon] starting 'node playground'
Success: Hey, It worked!
[nodemon] clean exit - waiting
[nodemon] restarting due to change
[nodemon] starting 'node playground'
Success: Hey, It worked!
[nodemon] clean exit - waiting
[nodemon] restarting due to change
[nodemon] starting 'node playground'
Success: Hey, It worked!
[nodemon] clean exit - waiting
[nodemon] restarting due to change
[nodemon] starting 'node playground'
Success: Hey, It worked!
[nodemon] clean exit - waiting
[nodemon] restarting due to change
[nodemon] starting 'node playground'
Success: Hey, It worked!
[nodemon] clean exit - waiting
[nodemon] restarting due to change
[nodemon] starting 'node playground'
Success: Hey, It worked!
[nodemon] clean exit - waiting
[nodemon] restarting due to change
[nodemon] starting 'node playground'
Success: Hey, It worked!
[nodemon] clean exit - waiting
[nodemon] restarting due to change
[nodemon] starting 'node playground'
Success: Hey, It worked!
```

```
var somePromise = new Promise((resolve, reject) => {
  setTimeout(() => {
    resolve('Hey, It worked!');
    // reject('Unable to fulfill promise');
  }, 2500);
});

somePromise.then((message) => {
  console.log('Success: ' + message);
}, (errorMessage) => {
  console.log('Error: ' + errorMessage);
});
```

5) Web Servers and application Deployment

A screenshot of a Windows desktop showing a terminal window and a code editor in Visual Studio Code. The terminal window shows the user performing a git push operation to a remote repository. The code editor shows a file named 'server.js' containing Node.js server code. A tooltip is visible over the code, highlighting a 'callback' function.

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

views/projects.hbs

no changes added to commit (use "git add" and/or "git commit -a")
Andrew:~/Desktop/node-web-server (master)$ git add .
Andrew:~/Desktop/node-web-server (master)$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)
    modified:   server.js
    modified:   views/partials/header.hbs
    new file:   views/projects.hbs

Andrew:~/Desktop/node-web-server (master)$ git commit -m 'Add project page'
[master e4ae6f2] Add project page
 3 files changed, 21 insertions(+)
 create mode 100644 views/projects.hbs
Andrew:~/Desktop/node-web-server (master)$ git push
Counting objects: 7, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (7/7), done.
Writing objects: 100% (7/7), 791 bytes | 0 bytes/s, done.
Total 7 (delta 3), reused 0 (delta 0)
```

```
File Edit Selection View Go Debug Terminal Help sever.js - Untitled (Workspace) - Visual Studio Code
OPEN EDITORS
  JS server.js
UNTITLED (WORKSPACE)
  node-web-server
    package-lock.json
    package.json
    server.js
  public
    help.html
  views
    partials
      about.hbs
      home.hbs
      maintenance.hbs
      projects.hbs
  PROBLEMS OUTPUT TERMINAL ... 1: powershell
PS C:\Users\ASUS\Desktop\node-web-server>
PS C:\Users\ASUS\Desktop\node-web-server>
```

6) Testing your application

The screenshot shows a Windows desktop environment with Visual Studio Code running in the foreground. The terminal window on the left displays the command `npm test` and its output, which includes tests for a `Server` and `App` module, both of which pass. The main editor window on the right contains the file `app.test.js` with code using `expect` and `rewire` modules to test an application's database interaction. The code includes assertions for saving and calling a spy function.

```
const expect = require('expect');
const rewire = require('rewire');

var app = rewire('../app');

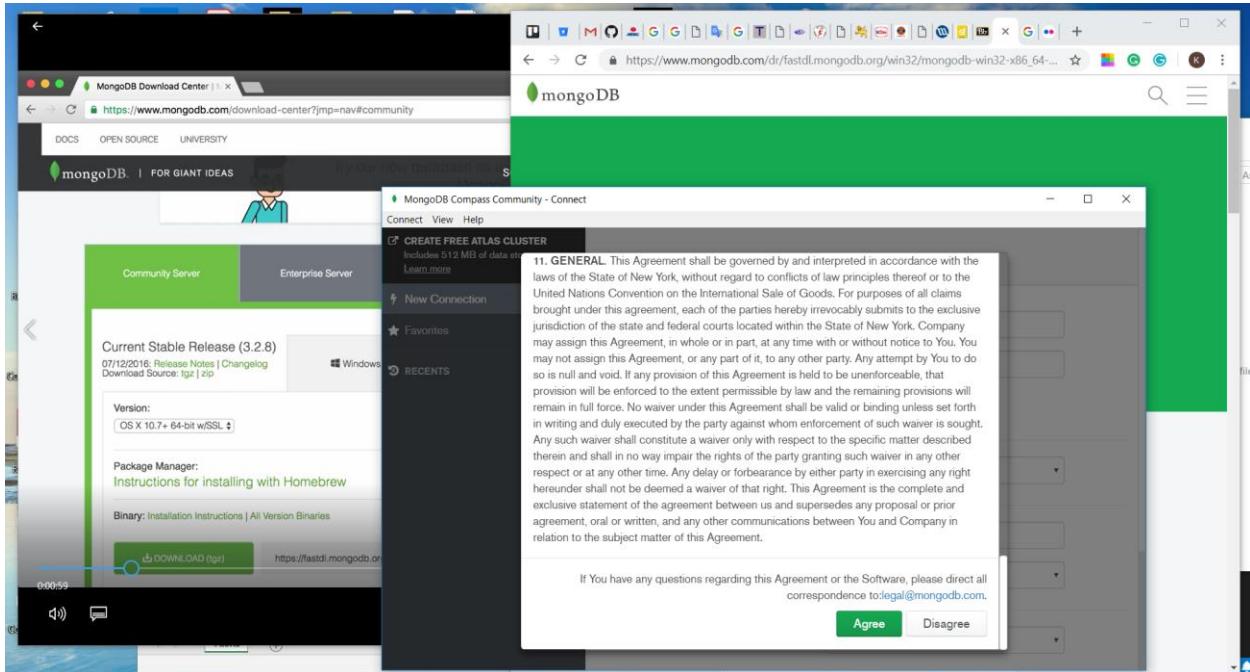
describe('App', () => {
  var db = {
    saveUser: expect.createSpy()
  };
  app.__set__('db', db);

  it('should call the spy correctly', () => {
    var spy = expect.createSpy();
    spy('Andrew', 25);
    expect(spy).toHaveBeenCalled();
  });

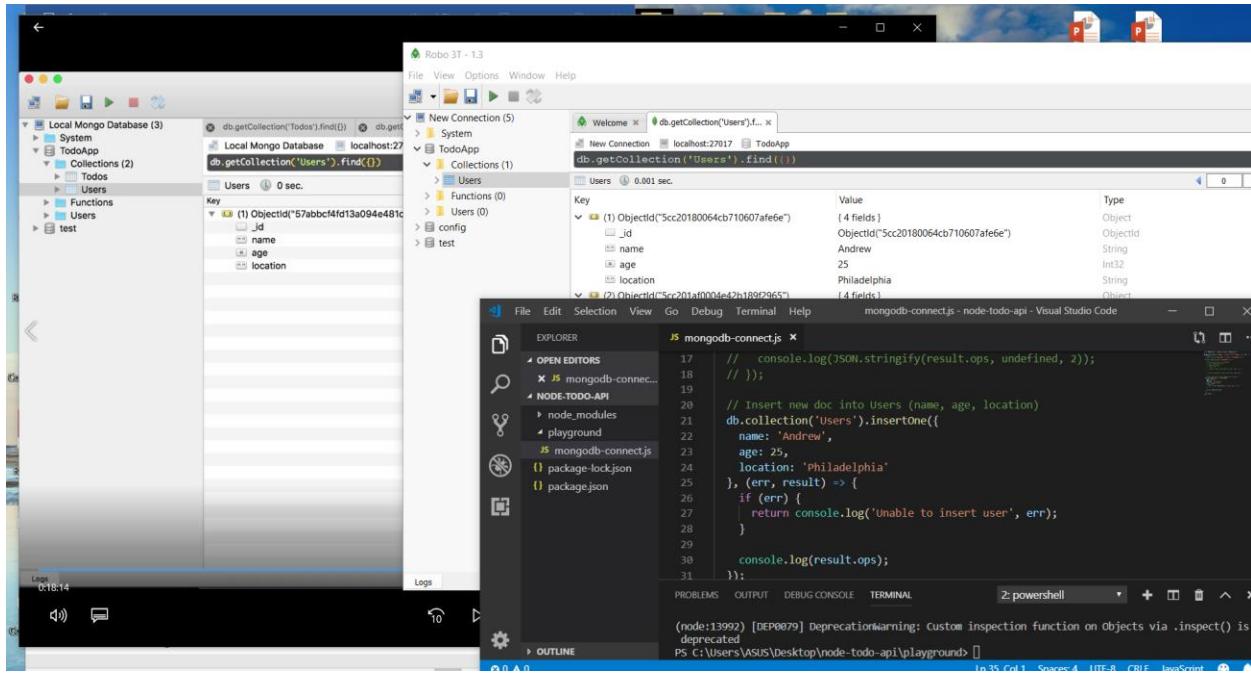
  it('should call saveUser', () => {
    var email = 'andrew@example.com';
    var password = '123abc';
    app.handleSignup(email, password);
    expect(db.saveUser).toHaveBeenCalledWith(email, password);
  });
});

it('should call saveUser with user object', () => {
  var email = 'kasuni@example.com';
  var password = '123abc';
  app.handleSignup(email, password);
  expect(db.saveUser).toHaveBeenCalledWith({email, password});
});
```

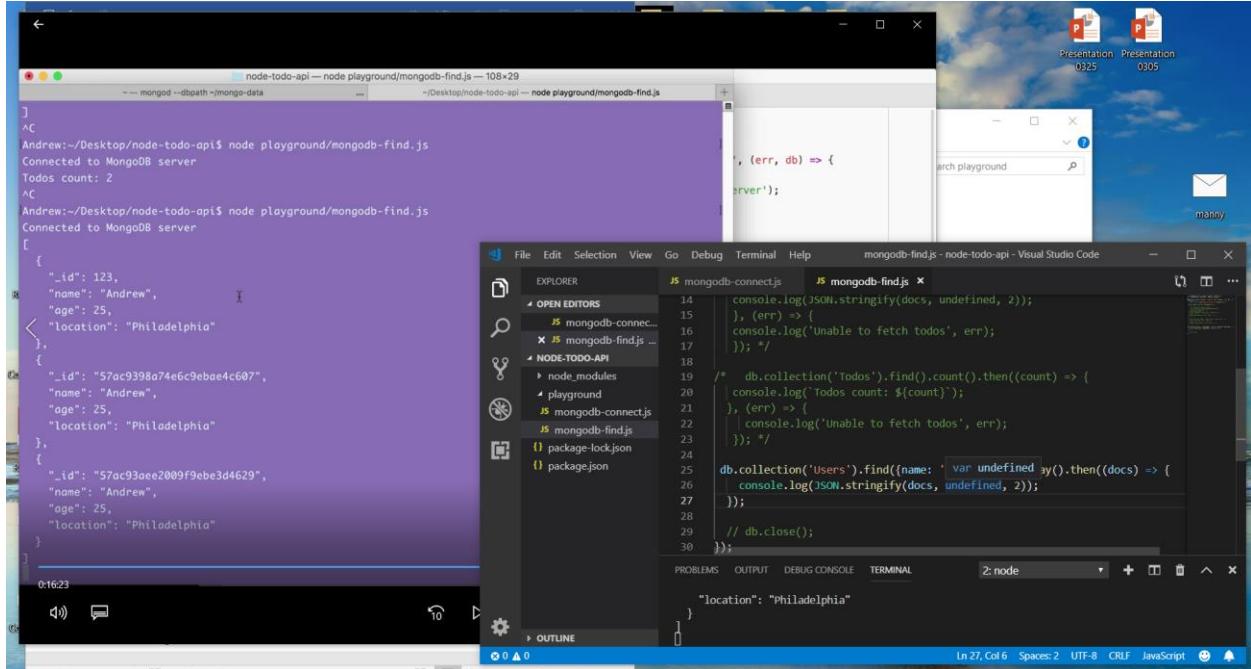
7) MongoDB and REST APIs



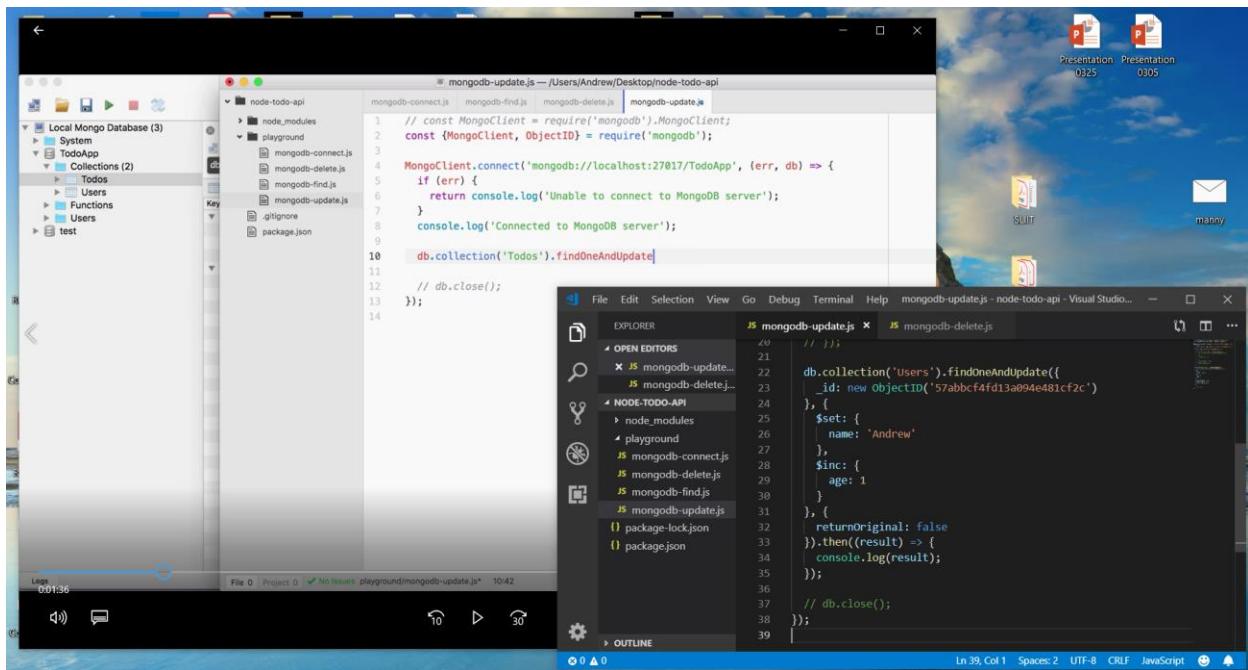
Connecting to Mongo and Writing Data



The ObjectId & Fetching data



Deleting Document & Updating Document



A screenshot of Visual Studio Code showing two files: `mongodb-update.js` and `mongodb-delete.js`. The `mongodb-update.js` file contains code to connect to a MongoDB database and update a document in the `Todos` collection. The `mongodb-delete.js` file contains code to delete a document from the `Users` collection.

```
// const MongoClient = require('mongodb').MongoClient;
const {MongoClient, ObjectId} = require('mongodb');

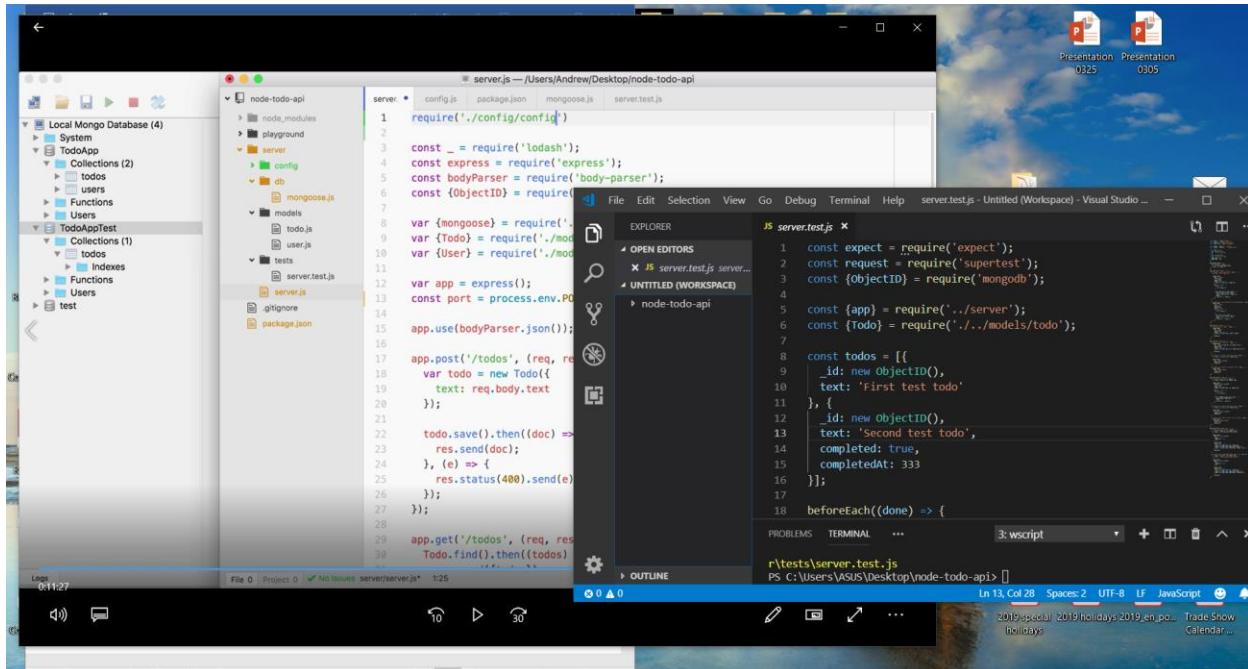
MongoClient.connect('mongodb://localhost:27017/TodoApp', (err, db) => {
  if (err) {
    return console.log('Unable to connect to MongoDB server');
  }
  console.log('Connected to MongoDB server');

  db.collection('Todos').findOneAndUpdate(
    { _id: '57abbc4fd13a094e481cf2c' },
    {
      $set: {
        name: 'Andrew'
      },
      $inc: {
        age: 1
      }
    },
    {
      returnOriginal: false
    }).then(result => {
    console.log(result);
  });
  // db.close();
});

// db.close();
```

```
db.collection('Users').findOneAndDelete({
  _id: new ObjectId('57abbc4fd13a094e481cf2c')
}, {
  $set: {
    name: 'Andrew'
  },
  $inc: {
    age: 1
  }
});
```

Creating a Test Database



A screenshot of Visual Studio Code showing a test file `server.test.js` for a Node.js application. The file uses the `supertest` library to test the `Todo` endpoint of the application. It sends a POST request to create a new todo item and a GET request to retrieve all todos.

```
const expect = require('expect');
const request = require('supertest');
const {ObjectId} = require('mongodb');

const app = require('../server');
const port = process.env.PORT || 3001;

app.use(bodyParser.json());
app.use('/todos', todosRouter);

app.post('/todos', (req, res) => {
  var todo = new Todo({
    text: req.body.text
  });

  todo.save().then((doc) => {
    res.send(doc);
  }, (e) => {
    res.status(400).send(e);
  });
});

app.get('/todos', (req, res) => {
  Todo.find().then((todos) => {
    res.send(todos);
  });
});

beforeEach((done) => {
```