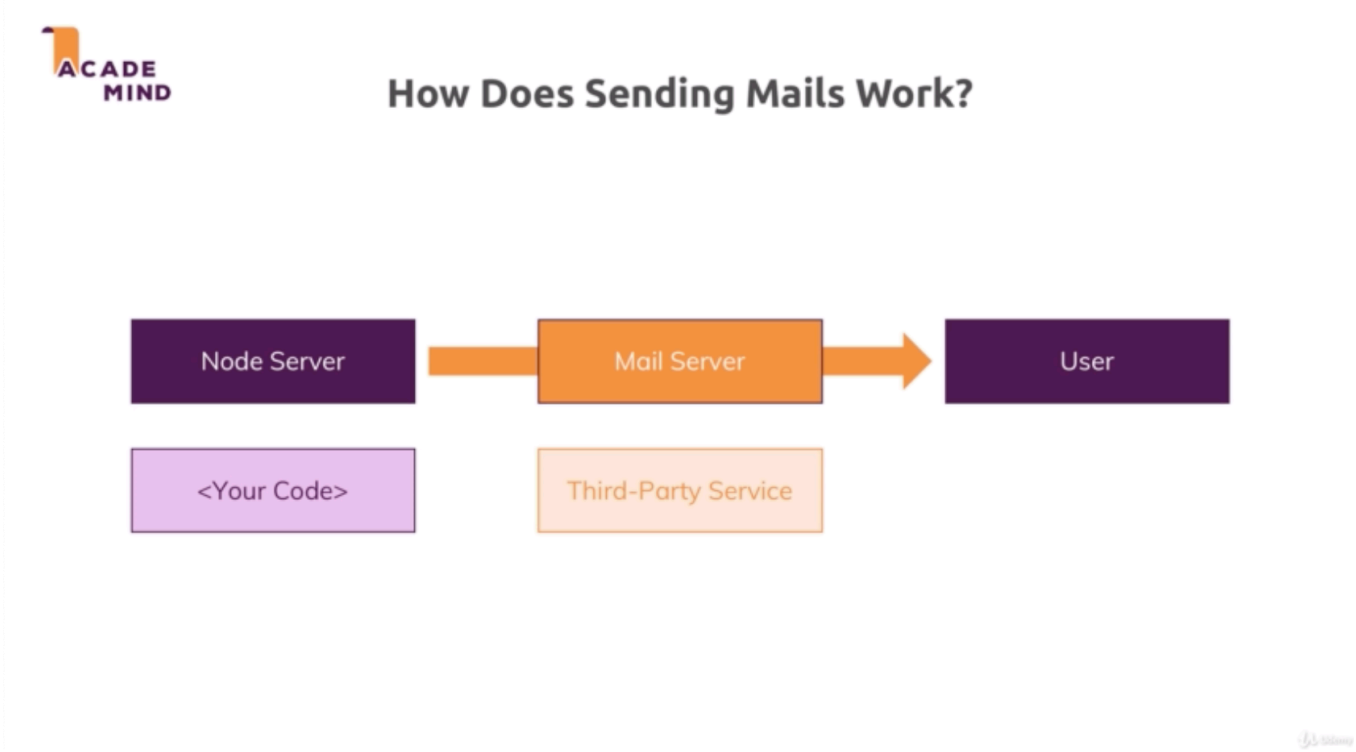
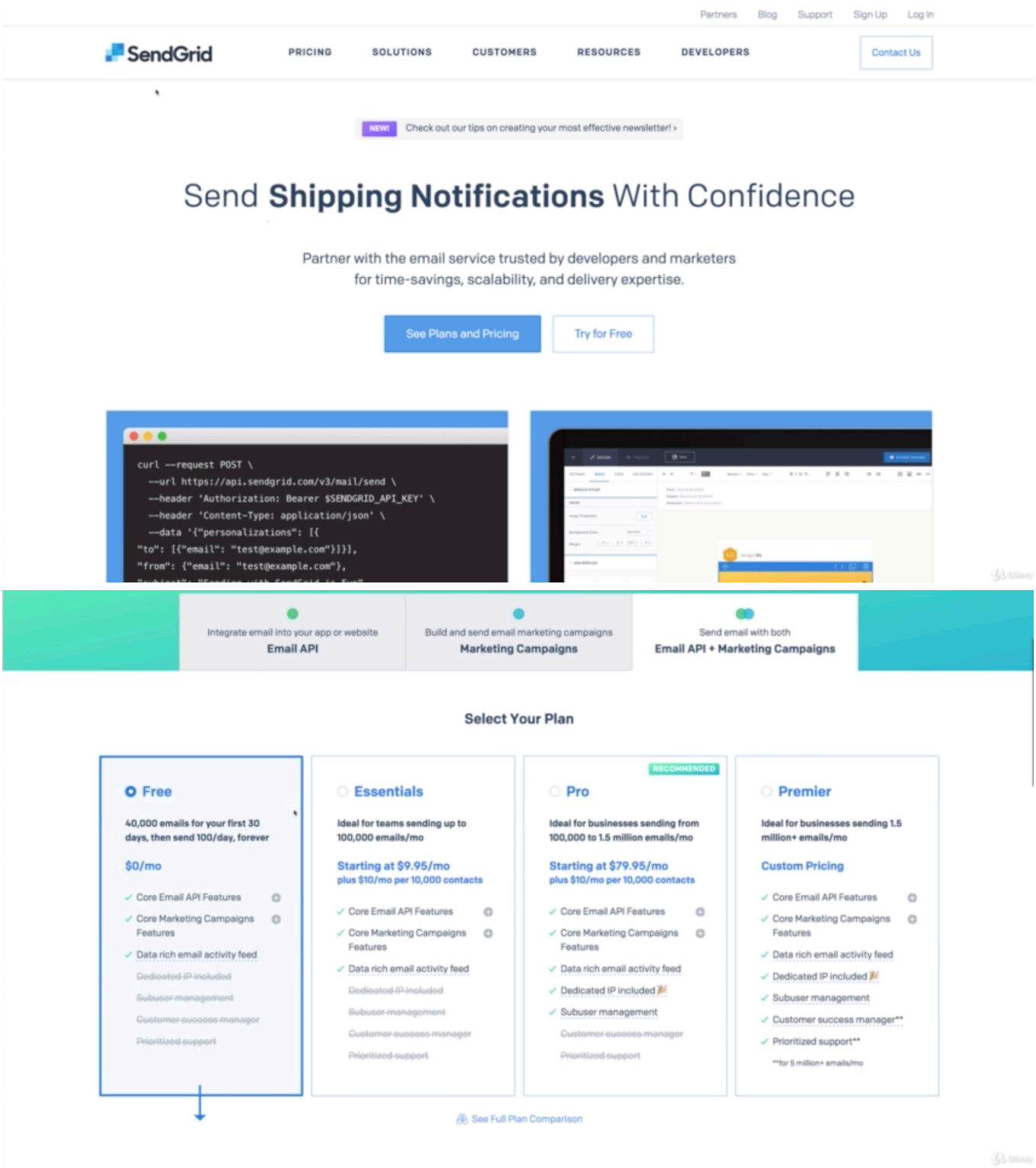
16. Sending Emails

* Chapter 267: How Does Sending Emails Work?

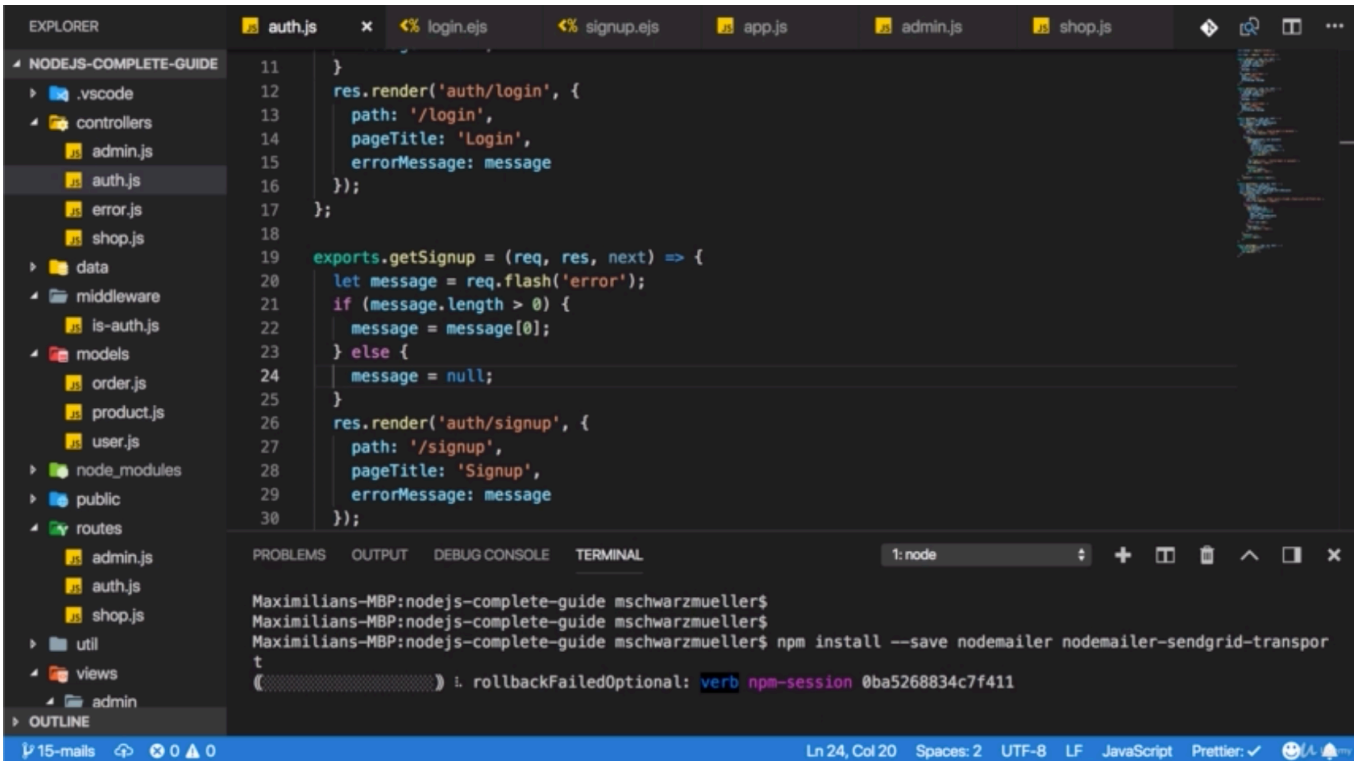


- node.js and express.js are language or frameworks runtimes that we use for writing our server side logic. but with node.js you can't trivially create a mailing server. Handling mails is different to handling incoming requests and response. because it have to handle millions of mails and security and so on.
- so you will never implement your own mail server because that is a very complex task. so in reality you typically use 3rd party mail serers.
- by the way, all major web applications you might be interacting with including Udemy don't have their own mail servers. they are using 3rd party providers like AWS or whatever it's for sending emails

* Chapter 268: Using SendGrid

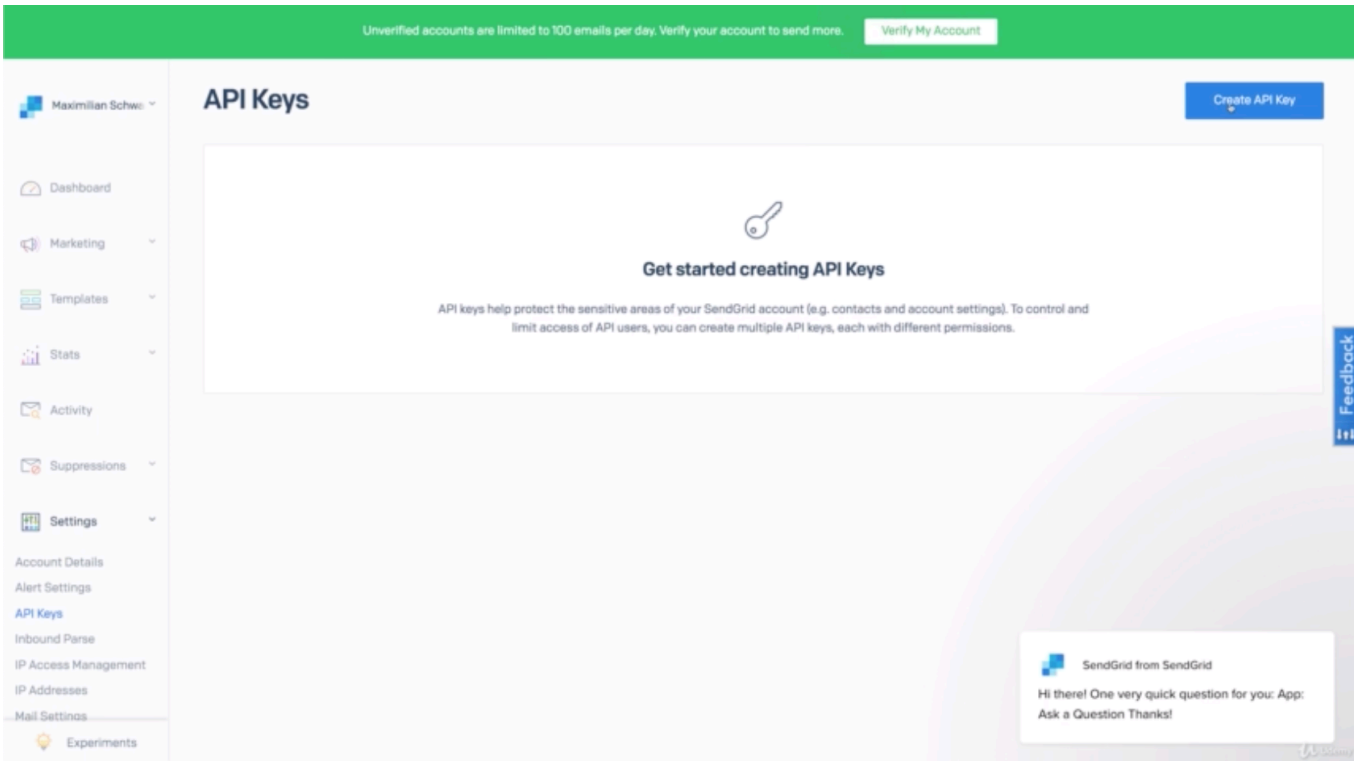
- we will use it because they have a free entry tier.
- there are many alternatives like Mailchimp, AWS, SCS etc.



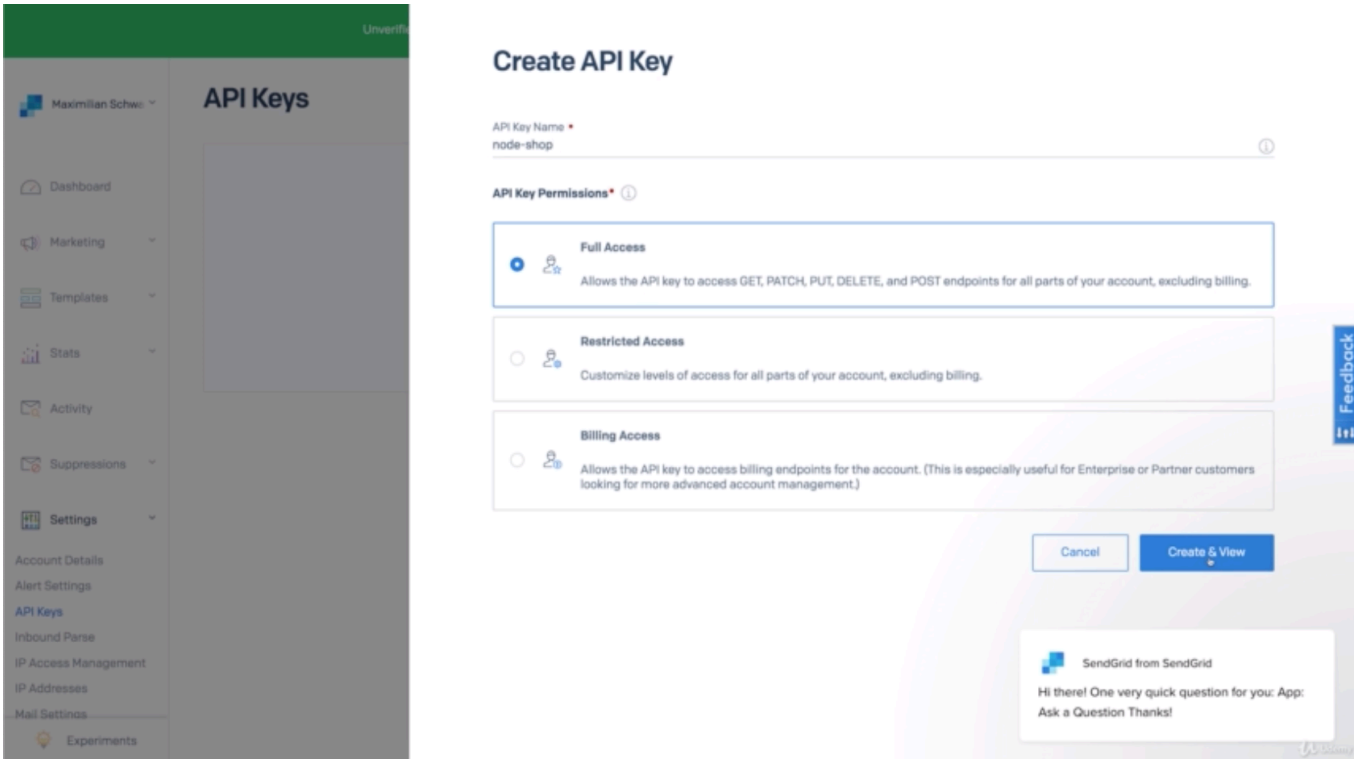
- 'npm install —save nodemailer nodemailer-sendgrid-transport' that is a package that will help us with intergrating sendgrid and conveniently use that together with nodemailer.
- 'nodemailer' is a package that makes sending emails from inside node.js.

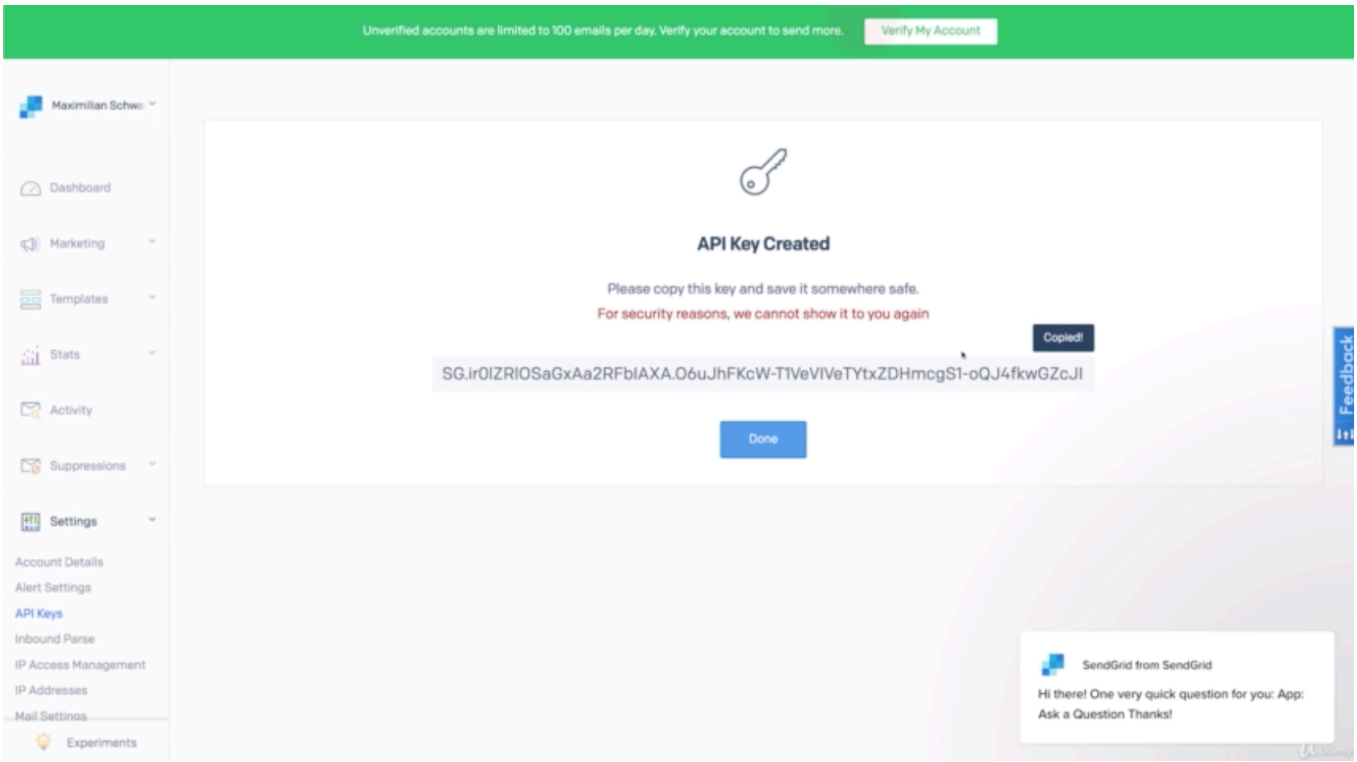
* Chapter 269: Using Nodemailer To Send An Email

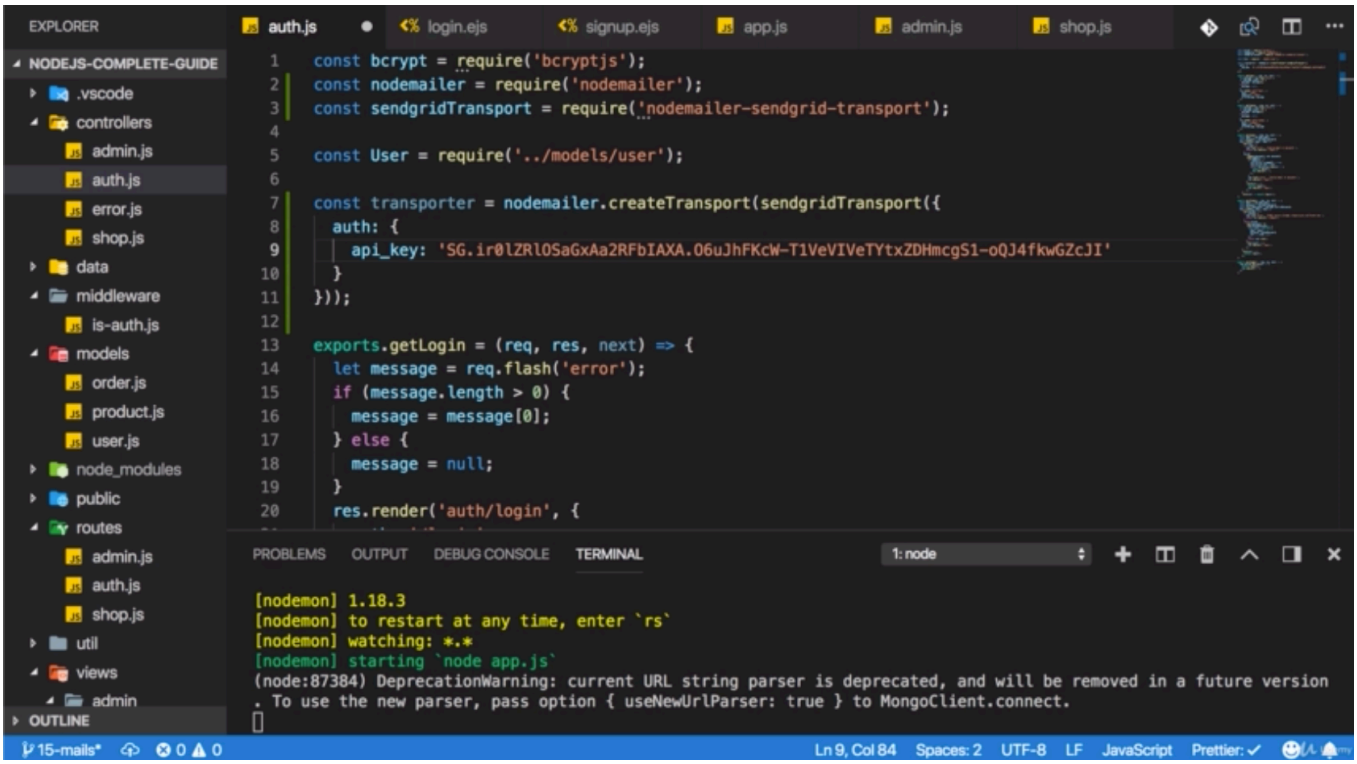
1. update
- ./controllers/auth.js



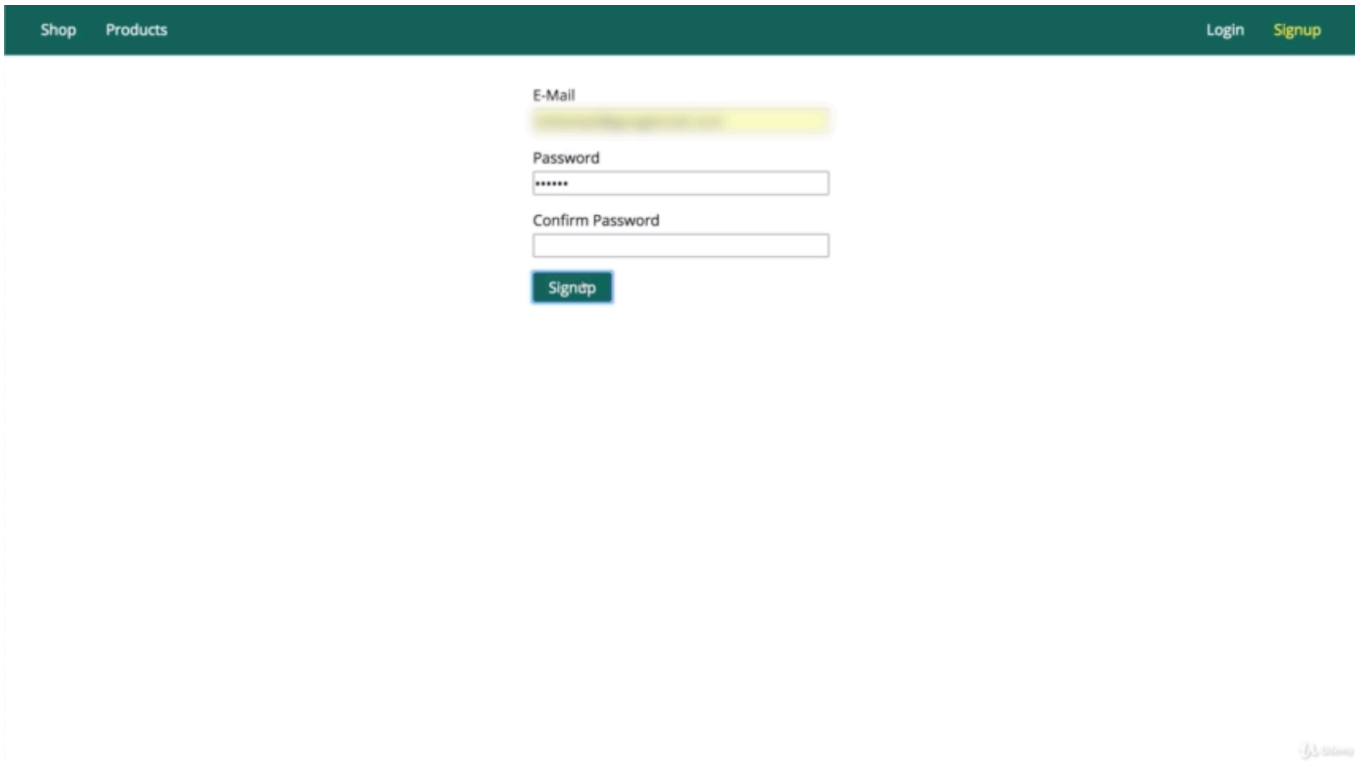
- go to settings and go to API Keys and click ‘Create API Key’.

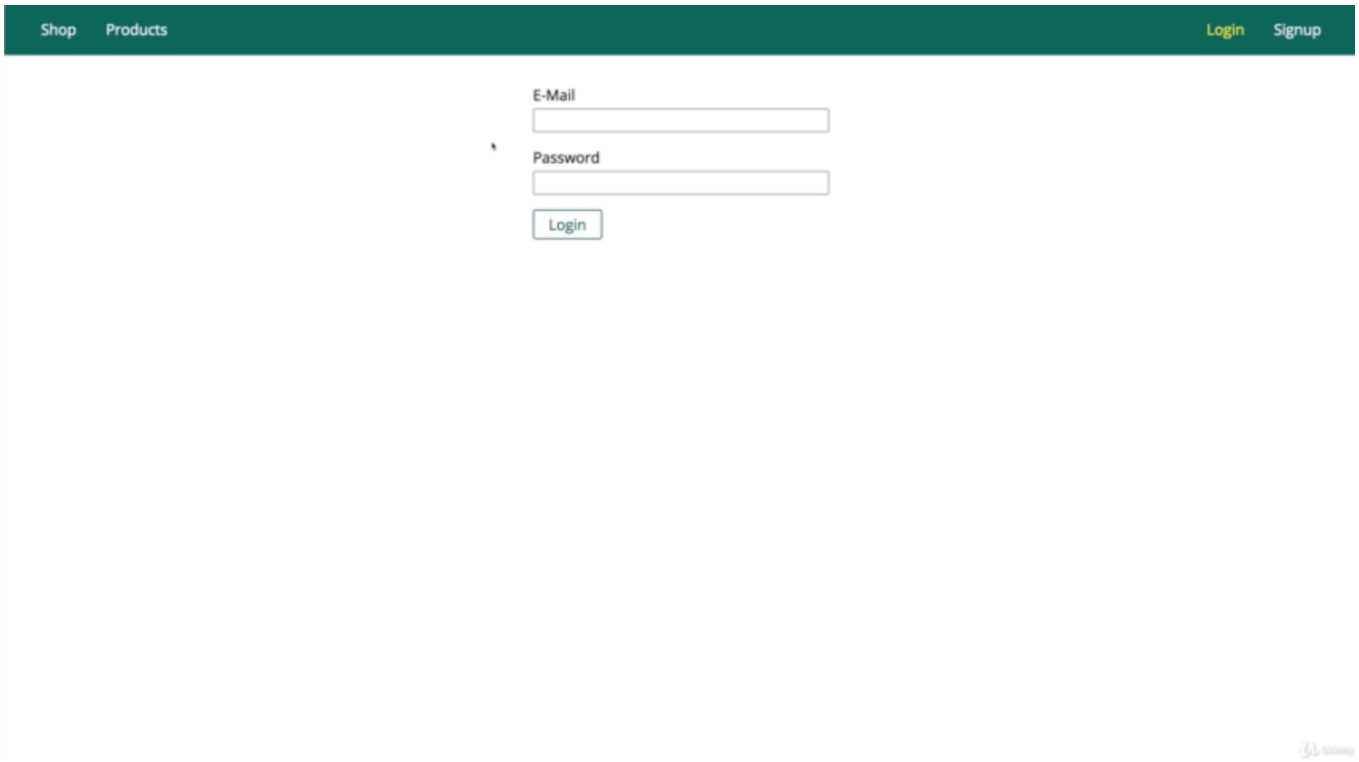



Unverified accounts are limited to 100 emails per day. Verify your account to send more.
Verify My Account
Dashboard
Marketing
Templates
Stats
Activity
Suppressions
Settings
Account Details
Alert Settings
API Keys
Inbound Parse
IP Access Management
IP Addresses
Experiments
API Key Created
Please copy this key and save it somewhere safe.
For security reasons, we cannot show it to you again
Copied!
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
Done
Feedback
SendGrid from SendGrid
Hi there! One very quick question for you: App:
Ask a Question Thanks!


EXPLORER
auth.js
login.ejs
signup.ejs
app.js
admin.js
shop.js
NODEJS-COMPLETE-GUIDE
.vscode
controllers
admin.js
auth.js
error.js
shop.js
data
middleware
is-auth.js
models
order.js
product.js
user.js
node_modules
public
routes
admin.js
auth.js
shop.js
util
views
admin
OUTLINE
const bcrypt = require('bcryptjs');
const nodemailer = require('nodemailer');
const sendgridTransport = require('nodemailer-sendgrid-transport');
const User = require('../models/user');
const transporter = nodemailer.createTransport(sendgridTransport({
auth: {
api_key: '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'
}
}));
exports.getLogin = (req, res, next) => {
let message = req.flash('error');
if (message.length > 0) {
message = message[0];
} else {
message = null;
}
res.render('auth/login', {
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
1: node
[nodemon] 1.18.3
[nodemon] to restart at any time, enter `rs`
[nodemon] watching: *.*
[nodemon] starting `node app.js`
(node:87384) DeprecationWarning: current URL string parser is deprecated, and will be removed in a future version
. To use the new parser, pass option { useNewUrlParser: true } to MongoClient.connect.
15-mails*
0 0
Ln 9, Col 84
Spaces: 2
UTF-8
LF
JavaScript
Prettier:

——————————





- now you could use real email address. and i'm redirected.
- and now check your email account. and in that email account, you should have an email from shop@nodecomplete.com with that message.
- if you don't get it, verify you entered a correct e-mail.

```
//./controllers/auth.js

const bcrypt = require('bcryptjs');
const nodemailer = require('nodemailer')
const sendgridTransport = require('nodemailer-sendgrid-transport')

const User = require('../models/user');

```

```
/**initialize a couple of things
 * for nodemailer, we need to initialize a so-called transporter.
 * which is some setup telling nodemailer how your e-mails will be delivered
 *
 * execute 'sendgridTransport' as a function
 * because this function will then return configuration that nodemailer can use to use sendgrid.
 *
 * with the transporter configured,
 * you can now use it to send an email
 * and i wanna send a e-mail after signing up.
 * so go to 'postSignup'
*/
const transporter = nodemailer.createTransport(sendgridTransport({
  auth: {
    /**both are values you get from inside your sendgrid account */
    api_key: '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'
  }
}))

exports.getLogin = (req, res, next) => {
  let message = req.flash('error')
  if(message.length > 0){
    message = message[0]
  } else {
    message = null
  }
  res.render('auth/login', {
    path: '/login',
    pageTitle: 'Login',
    errorMessage: message
  });
};

exports.getSignup = (req, res, next) => {
  let message = req.flash('error')
  if(message.length > 0){
    message = message[0]
  } else {
    message = null
  }
  res.render('auth/signup', {
    path: '/signup',
    pageTitle: 'Signup',
    errorMessage: message
  });
};

exports.postLogin = (req, res, next) => {
  const email = req.body.email;
  const password = req.body.password;
  User.findOne({ email: email })
    .then(user => {
      if (!user) {
        req.flash('error', 'Invalid email or password.')
        return res.redirect('/login');
```

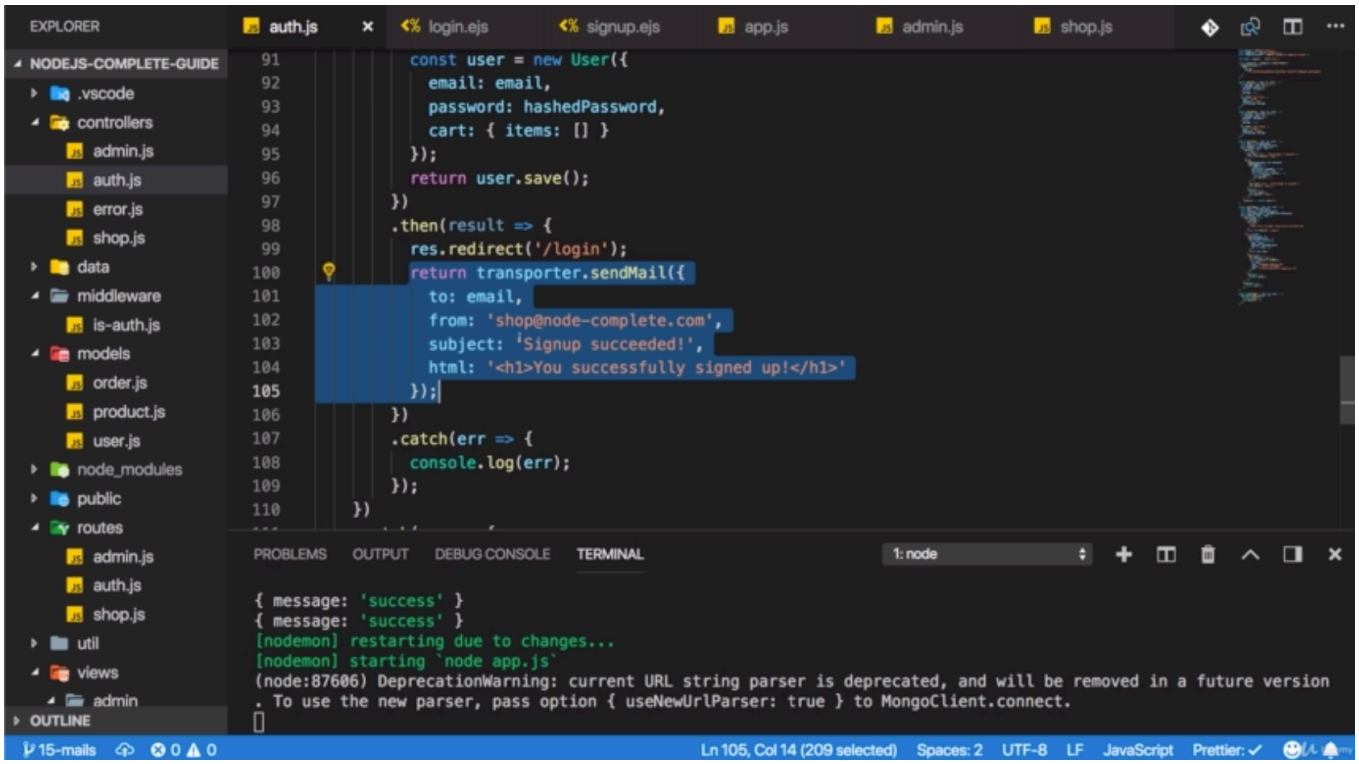
```
      }
      bcrypt
        .compare(password, user.password)
        .then(doMatch => {
          if (doMatch) {
            req.session.isLoggedIn = true;
            req.session.user = user;
            return req.session.save(err => {
              console.log(err);
              res.redirect('/');
            });
          }
          req.flash('error', 'Invalid email or password.')
          res.redirect('/login');
        })
        .catch(err => {
          console.log(err);
          res.redirect('/login');
        });
    })
    .catch(err => console.log(err));
};

exports.postSignup = (req, res, next) => {
  const email = req.body.email;
  const password = req.body.password;
  const confirmPassword = req.body.confirmPassword;
  User.findOne({ email: email })
    .then(userDoc => {
      if (userDoc) {
        req.flash('error', 'E-Mail exists already, please pick a different one. ')
        return res.redirect('/signup');
      }
      return bcrypt
        .hash(password, 12)
        .then(hashedPassword => {
          const user = new User({
            email: email,
            password: hashedPassword,
            cart: { items: [] }
          });
          return user.save();
        })
        .then(result => {
          res.redirect('/login');
          return transporter.sendMail({
            /**you pass a javascript object
             * where you configure the email you wanna send
             */
            to: email,
            from: 'shop@nodecomplete.com',
            subject: 'Signup Succeeded!',
            html: '<h1>You Successfully Signed Up!</h1>'
          })
        })
        .catch(err => {
```

```
        console.log(err)
      })
    })
    .catch(err => {
      console.log(err);
    });
};

exports.postLogout = (req, res, next) => {
  req.session.destroy(err => {
    console.log(err);
    res.redirect('/');
  });
};
```

* Chapter 270: Potential Limitation For Large Scale Apps



- it's good that we don't block the redirect but that we direct and send the mail at the same time because if you have an application with a lot of requests and you would wait for the email to be sent before you redirect. you might slow down your application because you are sending a lot of emails.
- depending on the size of your app we are talking about really huge apps, you could look into totally different approaches where you have some server side scripts running every x hours or every x minutes that send e-mails to newly signed users.
- so you should consider not using this in a blocking way because if you wait for this to be finished, this can be slow and you have to evalutate if it's worth waiting this or if your user can continue even without that mail being delivered.