CODING GARDEN FULL STACK PROJECT

1. 26:00 – how to deploy a vue app using ‘now’.
2. Look into creating a postgres docker container.
3. Read more about “promises” in javascript.
4. Difference between using promises and async \_ await.
5. Knex seed: the idea of a seed is that you’re resetting the database to how it’s supposed to be at the very beginning.
6. **Heroku deployment**: When you add the Heroku postgres addon command, it creates an environment variable which is the production connection. Hence, the need to have a production connection inside your database.config.
7. Look into how to build an image url API. Hint: there’s an image uri validator: which is like:

Image\_url: Joi.string().uri({

Scheme: [

‘git’,

/git\+https?/

]

});

1. Further look into callback functions and Promises.
2. Conduct extensive research into destructuring.
3. Read on HTTP status codes.

passport.use(new GoogleStrategy({

clientID: process.env.GOOGLE\_CLIENT\_ID,

clientSecret: process.env.GOOGLE\_CLIENT\_SECRET,

callbackURL: '/auth/google/callback'

}, async (accessToken, refreshToken, profile, cb) => {

const email = profile.emails[0].value;

const googleUser = {

display\_name: profile.displayName,

email,

google\_id: profile.id,

image\_url: profile.photos[0].value,

role\_id: 1,

};

try {

let user = await users.findByEmail(email);

if (user) {

googleUser.role\_id = user.role\_id;

user = await users.update(user.id, googleUser);

} else {

},

},

},

))

user = await users.insert(googleUser);

try {

let user = await users.findByEmail(email);

if (user) {

googleUser.role\_id = user.role\_id;

user = await users.update(user.id, googleUser);

} else {

user = await users.insert(googleUser);

}

console.log('got user from db',);

return cb(null, user);

} catch (error) {

return cb(error);

}

http://localhost:3000/auth/google

**WHAT IS DESTRUCTURING?**

Destructuring simply implies breaking down a complex structure into simpler parts. This syntax can be used for variable declaration or variable assignment.

**WHAT IS A PROMISE?**

A Promise object represents the eventual completion (or failure) of an asynchronous operation and its resulting value. A promise allows you to associate handlers with an asynchronous action’s eventual success or failure.

**What is try…catch?**

The try…catch statement marks a block of statements to try and specify a response should an exception be thrown.