Vérifier l’existence des informations du client au moment de la validation

it("should return -1 if customer informations are not set", () => {

//Arrange

ctrl.customerForm = {};

//Act

var result = ctrl.onValidate();

//Assert

expect(result).toEqual(-1);

});

Implémenter la vérification (montrer que ça ne fonctionne pas)

onValidate(){

if(this.customerForm == {}){

return -1;

}

}

Utiliser la fonction isEmpty de la bibliothéque pour vérifier si l’objet est vide

if(\_.isEmpty(this.customerForm)){

return -1;

}

Vérifier l’existence de toutes les informations du client au moment de la validation (nom, email, numéro de compte)

it("should return -2 if one or more customer informations are not defined", () => {

//Arrange

ctrl.customerForm = {

name: "john",

email:"",

accountNumber: "3456"

};

//Act

var result = ctrl.onValidate();

//Assert

expect(result).toEqual(-2);

});

if(this.customerForm.name != "" || this.customerForm.email != "" || this.customerForm.accountNumber != ""){

return -2;

}

Refacto : utiliser la fonction \_.isEmpty de lodash

if(\_.isEmpty(this.customerForm.name) || \_.isEmpty(this.customerForm.email) || \_.isEmpty(this.customerForm.accountNumber)){

return -2;

}

Vérifier l’existence des informations de l’adresse

it("should return -3 if address informations are not set", () => {

//Arrange

ctrl.addressForm = {};

//Act

var result = ctrl.onValidate();

//Assert

expect(result).toEqual(-3);

});

ça fonctionne pas puisqu’il nous manque les informations du client

it("should return -3 if address informations are not set", () => {

//Arrange

ctrl.addressForm = {};

//Act

var result = ctrl.onValidate();

//Assert

expect(result).toEqual(-3);

});

Fonctionne quand on ajoute les informations du client

it("should return -3 if address informations are not set", () => {

//Arrange

ctrl.customerForm = {

name: "john",

email:"john.eliott@gmail.com",

accountNumber: "3456"

};

ctrl.addressForm = {};

//Act

var result = ctrl.onValidate();

//Assert

expect(result).toEqual(-3);

});

Vérifier l’existence de toutes les informations sur l’adresse

it("should return -4 if one or more address informations are not defined", () => {

//Arrange

//Arrange

ctrl.customerForm = {

name: "john",

email:"john.eliott@gmail.com",

accountNumber: "3456"

};

ctrl.addressForm = {

street: "2 rue de Paris",

postalCode:"",

country: "FR"

};

//Act

var result = ctrl.onValidate();

//Assert

expect(result).toEqual(-4);

});

La partie du code permettant de vérifier les informations de l’adresse

if(\_.isEmpty(this.addressForm.street) || \_.isEmpty(this.addressForm.postalCode) || \_.isEmpty(this.addressForm.country)){

return -4;

}

Vérifier si la rue est valide

it("should return -5 if street is not valid", () => {

//Arrange

//Arrange

ctrl.customerForm = {

name: "john",

email:"john.eliott@gmail.com",

accountNumber: "3456"

};

ctrl.addressForm = {

street: "rue du commerce",

postalCode:"36543",

country: "FR"

};

//Act

var result = ctrl.onValidate();

//Assert

expect(result).toEqual(-5);

});

if(this.addressForm.street != '^\d+\s[A-z]+\s[A-z]+'){

return -5;

}

Extraire la vérification dans une fonction (refacto)

it("should return false if street is not valid", () => {

//Arrange

var street ="rue du commerce";

//Act

var result = ctrl.isValidStreet(street);

//Assert

expect(result).toEqual(false);

});

Créer une fonction

isValidStreet(street){

return street.match(/^\d+\s[A-z]+\s[A-z]+/g);

}

Appeler la function

if(!this.isValidStreet(this.addressForm.street)){

return -5;

}

Ecrire le test pour vérifier la validité du code postal

it("should return -6 if postal code is not valid", () => {

//Arrange

ctrl.customerForm = {

name: "john",

email:"john.eliott@gmail.com",

accountNumber: "3456"

};

ctrl.addressForm = {

street: "2 rue de Paris",

postalCode:"123456",

country: "FR"

};

//Act

var result = ctrl.onValidate();

//Assert

expect(result).toEqual(-6);

});

Faire fonctionner le test

if(!this.addressForm.postalCode.match(/\d{5}/g)){

return -6;

}

Le test ne fonctionne pas

if(!this.addressForm.postalCode.match(/\d{5}/g) || this.addressForm.postalCode.length != 5){

return -6;

}

Refacto

Extraire la vérification du code postal dans une fonction

it("should return null if postal code is not valid", () => {

//Arrange

var postalCode ="123456";

//Act

var result = ctrl.isValidPostalCode(postalCode);

//Assert

expect(result).toEqual(false);

});

isValidPostalCode(postalCode){

return postalCode.match(/\d{5}/g) && postalCode.length == 5;

}

Modifier la fonction de validation

if(!this.isValidPostalCode(this.addressForm.postalCode)){

return -6;

}

Vérifier la méthode de payement