**Тестування функцій**

|  |  |  |  |
| --- | --- | --- | --- |
| get\_menu | 1 |  | мокап форма |

app.get('/theme:id', (req, res) => {

themes.find({}, (err, allTheme) => { // Запит до БД

if (err) {

res.render('theme', { data: null, message: req.flash('err') })

} else {

res.render('theme', { data: allTheme }) // Відправлення результатів запиту для відображення

}

})

})

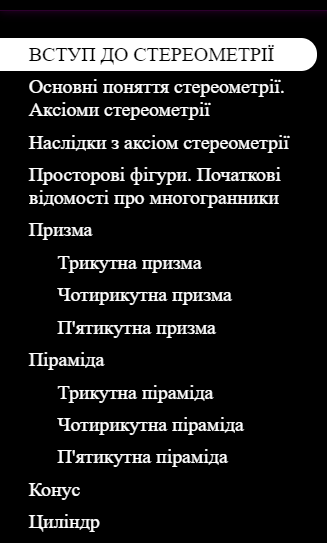


Рис. 1.1 Результат тестування функції з відображення меню

|  |  |  |  |
| --- | --- | --- | --- |
| get\_material | 2 |  | мокап форма |

app.get('/theme:id', (req, res) => {

material.find({theme.id}, (err, allMaterial) => { // Запит до БД

if (err) {

res.render('theme', { data: null, message: req.flash('err') })

} else {

res.render('theme', { data: allMaterial }) // Відправлення результатів запиту для відображення

}

})

})



Рис. 1.2 Результат тестування функції з відображення матеріалів

|  |  |  |  |
| --- | --- | --- | --- |
| get\_task | 3 |  | мокап форма |

app.get('/theme:id', (req, res) => {

task.find({theme.id}, (err, allTask) => { // Запит до БД

if (err) {

res.render('theme', { data: null, message: req.flash('err') })

} else {

res.render('theme', { data: allTask }) // Відправлення результатів запиту для відображення

}})

})



Рис. 1.3 Результат тестування функції з відображення завдання

|  |  |  |  |
| --- | --- | --- | --- |
| get\_solution | 4 |  | мокап форма |

app.get('/theme:id', (req, res) => {

solution.find({task.id}, (err, allSolution) => { // Запит до БД

if (err) {

res.render('theme', { data: null, message: req.flash('err') })

} else {

res.render('theme', { data: allSolution }) // Відправлення результатів запиту для відображення

}

})

})

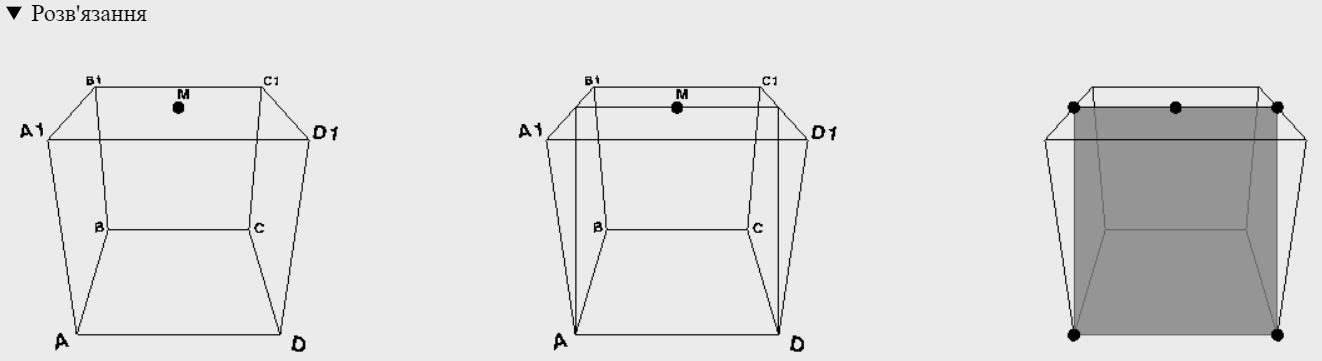


Рис. 1.4 Результат тестування функції з відображення розв’язання

|  |  |  |  |
| --- | --- | --- | --- |
| get\_object | 5 |  | мокап форма |

app.get('/theme:id', (req, res) => {

object.find({theme.id}, (err, allObject) => { // Запит до БД

if (err) {

res.render('theme', { data: null, message: req.flash('err') })

} else {

res.render('theme', { data: allObject }) / Відправлення результатів запиту для відображення

}

})

})

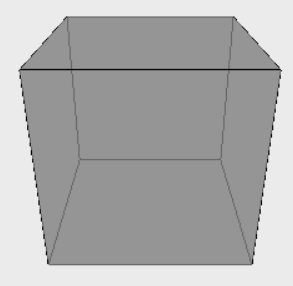


Рис. 1.5 Результат тестування функції з відображення об’єкту

|  |  |  |  |
| --- | --- | --- | --- |
| use\_object | 6 |  | мокап форма |

Зміна стану об’єкта відбувається на стороні клієнта. Приклад обертання.

{

document.querySelector('#cube').onclick = function(e){

if(rotateCube==false){rotateCube = true} else {rotateCube=false}

}}

if (rotateCube==true)

{

var time = 0.001

matrix.makeRotationY(time \* 2 \* Math.PI);

camera.position.applyMatrix4(matrix);

camera.lookAt(0,0,0);

quater = camera.quaternion;

}else if (rotateCube==false)

{

camera.quaternion = quater;

}

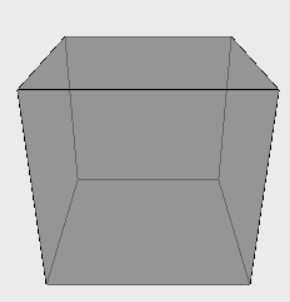
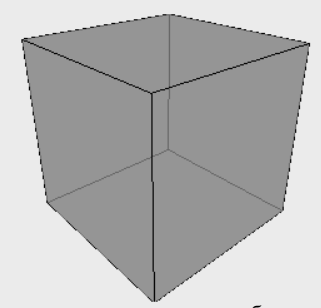
 → 

Рис. 1.6 Результат тестування функції з обертання об’єкту

|  |  |  |  |
| --- | --- | --- | --- |
| use\_object | 7 |  | мокап форма |

Приклад побудови перерізу об’єкта

{

document.querySelector('#cube').onclick = function(e){

CubeClickCounter++;

if(rotateCube==false){rotateCube = true} else {rotateCube=false}

}}

If(CubeClickCounter==2)

{

scene.remove(Cube,EdgeCube,mesh);

scene.add(CubeHalfCube, CubeHalfEdgeCube,CubeHalfCube2, CubeHalfEdgeCube2);

CubeHalfEdgeCube.position.set(-0.03, 0, -0.03);

CubeHalfCube.position.set(-0.03, 0, -0.03);

CubeHalfEdgeCube2.position.set(0.03, 0, 0.03);

CubeHalfCube2.position.set(0.03, 0, 0.03);

} else if(CubeClickCounter==4)

{

scene.remove(CubeHalfCube, CubeHalfEdgeCube, CubeHalfCube2, CubeHalfEdgeCube2);

scene.add(EdgeCube, mesh, Cube);

CubeClickCounter=0;

}

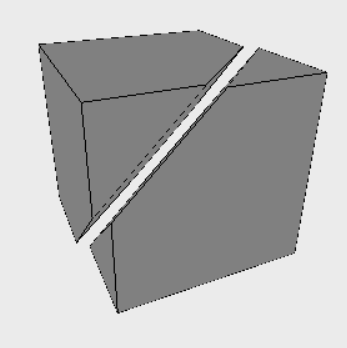
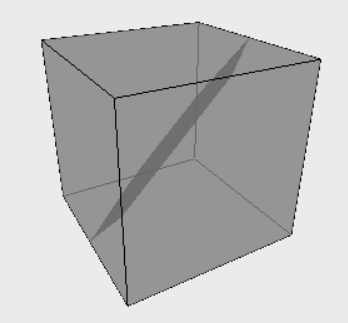


Рис. 1.7 Результат тестування функції з побудови перерізу об’єкту