Definición de endpoints del API

1. Endpoint para recuperación de logros disponibles del usuario al momento de jugar el videojuego

GET http://host:5000/api/videogame/

Input: JSON

```
{
        “userIdentificator”: INTEGER
}
```

Code:

```
videogame.get('/', async (req, res) => {
    try{
    let [rows] = await pool.query("CALL getAvailableUserAchievements(?);", [req.body.userIdentificator]);
    res.set('Access-Control-Allow-Origin', '*');
    res.status(HttpStatus.OK).json(rows[0]);
    }catch(error){
      return res.status(500).json({message: error});
    }
  });
```

2. Endpoint para almacenamiento de cada partida del videojuego en la base de datos

POST http://host:5000/api/videogame/

Input: JSON

```
{
        “time”: TIME,
        “wrongAnswers_”: INTEGER,
        “mode”: INTEGER,
        “modulesCompleted_”: INTEGER,
        “userIdentificator”: INTEGER,
        “achievements”: INTEGER LIST
}
```

Code:

```
videogame.post('/', async (req, res) => {

    try{

    let time = req.body.time.split(":");

    let [rows] = await pool.query("CALL registerGame(@id, ?, ?, ?, ?, ?); SELECT @id;",
["00:"+time[0]+":"+time[1], req.body.wrongAnswers_, req.body.mode, req.body.modulesCompleted_,
req.body.userIdentificator]);

    let gameIdentificator = rows[1][0]["@id"];

    await req.body.achievements.forEach(achievement => pool.query("CALL registerUserAchievement(?, ?, ?)",
[achievement, req.body.userIdentificator, gameIdentificator]));

    res.set('Access-Control-Allow-Origin', '*');

    res.sendStatus(HttpStatus.OK);

  }catch(error){

    return res.status(500).json({message: error});

  }

  });
```

3. Endpoint para recuperación de estadísticas personales y globales del videojuego

GET http://host:5000/api/videogame/stats

Input: JSON

```
{
        "userIdentificator": INTEGER
}
```

Code:

```
videogame.get('/stats', async (req, res) => {

    try{

    let [rows] = await pool.query("CALL getUserStatistics(?, 10);", [req.query.userIdentificator]);

    res.set('Access-Control-Allow-Origin', '*');

    res.status(HttpStatus.OK).json(rows);

  }catch(error){

    return res.status(500).json({message: error});

  }

  });
```

4. Endpoint para recuperación de estadísticas generales del videojuego

GET http://host:5000/api/videogame/statistics

Input: *NA*

Code:

```
videogame.get('/statistics', async (req, res) => {
    try{
    let [rows] = await pool.query("CALL getStatistics();");
    res.set('Access-Control-Allow-Origin', '*');
    res.status(HttpStatus.OK).json(rows);
  }catch(error){
    return res.status(500).json({message: error});
  }
  });
```

5. Endpoint para integración de la base de datos de Aulify BlackHole con la base de datos utilizada por todos los demás videojuegos

POST http://host:5000/api/videogame/user

Input: JSON

```
{
        “userIdentificator”: INTEGER,
        “username”: VARCHAR
}
```

Code:

```
videogame.post('/user', async (req, res) => {
      try{
      let [rows] = await pool.query("CALL registerUser(?, ?, NULL, NULL, NULL, NULL, NULL, NULL, NULL,
NULL);", [req.body.userIdentificator, req.body.username]);
      res.set('Access-Control-Allow-Origin', '*');
      res.sendStatus(HttpStatus.OK);
    }catch(error){
      return res.sendStatus(HttpStatus.OK);
    }
    });
```