```
const xlsxFile = require("read-excel-file/node");
const bodyParser = require("body-parser");
const readline = require("readline");
const express = require("express");
const path = require("path");
const fs = require("fs");
const app = express();

const http = require("http").createServer(app);
const webPort = 8080;

const excelPath = path.join(__dirname, "archivos", "ajedrez.xlsx");

var transitionTree;

// EJS INIT
// Set the view engine to ejs
app.set("view engine", "ejs");
// Set ejs files path
app.set("views", __dirname + "/dist/pages");
// Set body parser
app.use(bodyParser.urlencoded({ extended: true }));

// REQUESTS
app.get("/", (req, res) => {
  res.render("index");
});

app.post("/procesar/cadena/", (req, res) => {
  let response = {};
  let output;

  if (!req.body.auto) {
    if (req.body.input.length == 0) {
      req.body.input = generateMoves(20);
    }
  } else {
    req.body.input = generateMoves(10);
  }

  output = automata(req.body.input, 1, 16);

  response.moves = req.body.input;
  response.animations = output.animations;
  response.winner = output.winner;

  if (output.winner) {
    result.message = "Cadena ganadora!";
    fs.writeFile("./archivos/ganadores.txt", cadena, () => {});
  }

  // Send response
  res.send(JSON.stringify(response));
});
```

```
55
56  function generateMoves(num) {
57    let resultado = "";
58    for (let i = 0; i < num; ++i) {
59      resultado += Math.floor(Math.random() * 2) == 1 ? "r" : "b";
60    }
61    return resultado;
62  }
63
64  // AUTOMATA
65  // Main automata functionality
66  function automata(cadena, startNode, winNode) {
67    let current, next;
68    let animations = [];
69    let result = {};
70
71    current = [startNode];
72    animations.push(current);
73
74    cadena.split("").map((currentChar) => {
75      next = [];
76      current.forEach((nodo) => {
77        next = next.concat(processNode(currentChar, nodo));
78      });
79
80      // Removing duplicates
81      current = [...new Set(next)];
82
83      // Adding step to animation queue
84      animations.push(current);
85    });
86
87    // Check if winning condition
88    result.winner = current.includes(winNode);
89
90    // Add animation list
91    result.animations = animations;
92
93    return result;
94  }
95  // Node processing
96  function processNode(currentChar, nodeName) {
97    let resultado = [];
98
99    transitionTree.forEach((transition) => {
100     if (transition.nombre == nodeName) {
101       transition.pasos.forEach((paso) => {
102         if (paso.origen == currentChar) {
103           resultado = resultado.concat(paso.destinos);
104         }
105       });
106     }
107   });
108
```

```
  return resultado;
}
// Generate transition tree
function generateTree() {
  xlsxFile("./table.xlsx").then((rows) => {
    transitionTree = [];

    // Removing first row
    rows.shift();

    // Add a transition for each row
    rows.forEach((row) => {
      transitionTree.push({
        nombre: row[0],
        pasos: [
          {
            origen: "r",
            destinos: row[1]
              .toString()
              .split(",")
              .map((x) => +x),
          },
          {
            origen: "b",
            destinos: row[2]
              .toString()
              .split(",")
              .map((x) => +x),
          },
        ],
      });
    });
  });
}

// SERVER SET-UP
app.use(express.static(__dirname + "/dist/public/"));

// SERVER LISTEN INIT
http.listen(webPort, () => {
  console.log("Listening on port: " + webPort);
});

// Init
generateTree();
```

```
@media only screen and (min-width: 768px) and (max-width: 991px) {

  #main {
    width: 712px;
    padding: 100px 28px 120px;
  }

```

```
  /* .mono {
    font-size: 90%;
  } */

  .cssbtn a {
    margin-top: 10px;
    margin-bottom: 10px;
    width: 60px;
    height: 60px;
    font-size: 28px;
    line-height: 62px;
  }
```

```
class TelgramRequestHandler(object):
    def handle(self):
        addr = self.client_address[0]          # Client IP-adress
        telgram = self.request.recv(1024)      # Recieve telgram
        print "From: %s, Received: %s" % (addr, telgram)
        return
```

```

@media only screen and (min-width: 768px) and (max-width: 991px) {

  #main {
    width: 712px;
    padding: 100px 28px 120px;
  }

  /* .mono {
    font-size: 90%;
  } */

  .cssbtn a {
    margin-top: 10px;
    margin-bottom: 10px;
    width: 60px;
    height: 60px;
    font-size: 28px;
    line-height: 62px;
  }
```