



Федеральное государственное бюджетное образовательное учреждение высшего образования

**«МИРЭА – Российский технологический университет»**

**РТУ МИРЭА**

Институт информационных технологий

Кафедра инструментального и прикладного программного обеспечения

Дисциплина «Разработка серверных частей интернет-ресурсов»

## **Курсовая работа**

**Веб-приложение «Крестики-нолики»**

Студент: Хречко С.В.

Группа: ИКБО-03-21

Руководитель: доцент Куликов А.А.

# Цель

Разработать серверную часть веб-приложения «Крестики-нолики»

## Задачи

1. Провести анализ предметной области по тематике разработки
2. Обосновать выбор технологий
3. Разработать архитектуру приложения
4. Реализовать слой логики базы данных
5. Разработать слой серверной логики
6. Провести тестирование разработанного веб-приложения
7. Составить отчёт и презентацию

# Технологии разработки



Хранение данных



Express

JS

Разработка серверной части

Настройка проекта



Написание кода















Visual Studio Code

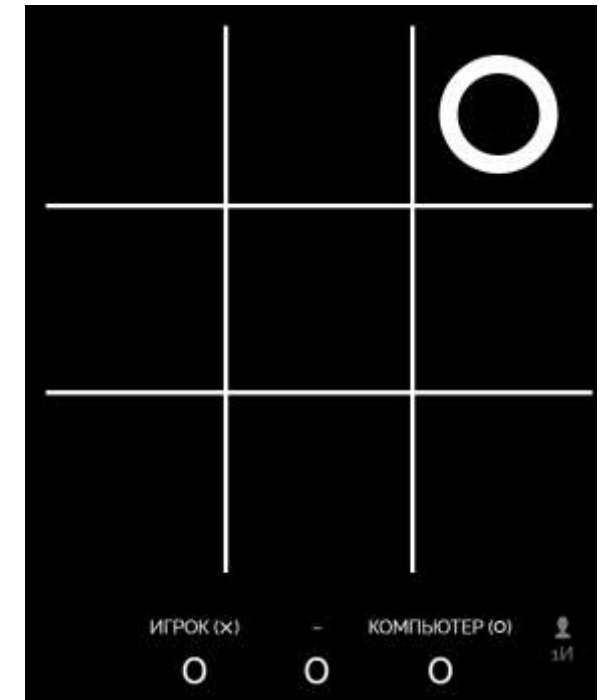
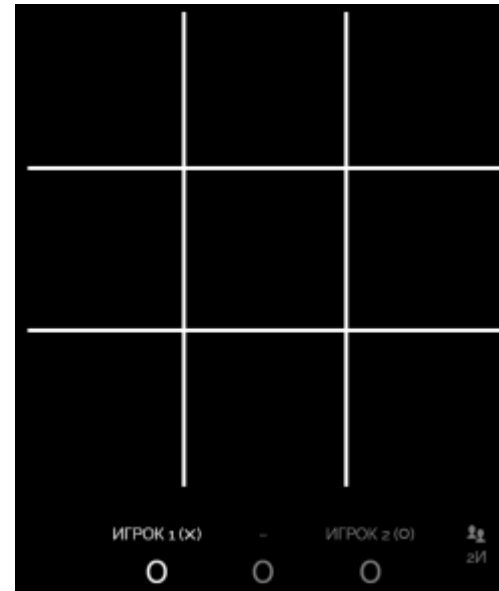
Тестирование  
приложения



# Анализ предметной области

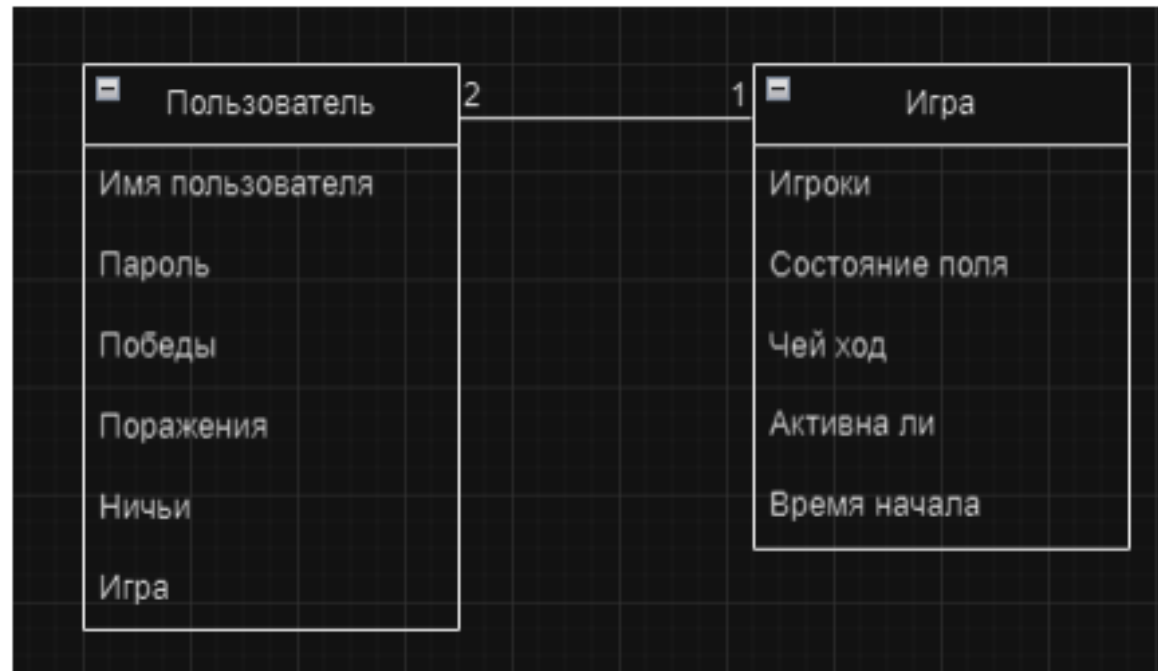
- Игра двух человек
- Сохранение результатов
- Вывод таблицы рекордов

		W/L/D	⌚	Score
1.	 Ludi 	79 / 8 / 123	00h 25m	2647
2.	 Susi 	43 / 29 / 142	00h 32m	2155
3.	 FOOGOL 	47 / 2 / 124	00h 34m	2103
4.	 SUGARDOLCE 	43 / 7 / 97	00h 17m	2011
5.	 xboxingZ 	37 / 3 / 78	00h 20m	1805
6.	 	37 / 13 / 82	00h 23m	1655



Странные ошибки

# Структура базы данных



Здесь будут  
небольшие изменения

# Чистая архитектура



# Ядро – основная логика

```
class TicTacToeGame {
  constructor(state = null) { ...
  }

  // Function to make a move on the board
  makeMove(row, col) { ...
  }

  // Function to switch to the next player
  switchPlayer(player_x, player_o) { ...
  }

  // Function to check if the game is over
  isGameOver() { ...
  }

  // Function to check if a player has won
  checkWin() { ...
  }

  // Function to check if the board is full
  isBoardFull() { ...
  }

  // Function to reset the game
  resetGame() { ...
  }

  // Function to get the current state of the game
  getGameState() { ...
  }
}
```

Ядро не зависит от внешних сущностей, логики и прочего

```
class TicTacToeGame {
  constructor(state = null) {
    if (state) {
      // If a state is provided, use it
      this.board = state.board;
      this.currentPlayer = state.currentPlayer;
    } else {
      // Otherwise, initialize with default values
      this.board = [
        [EMPTY, EMPTY, EMPTY],
        [EMPTY, EMPTY, EMPTY],
        [EMPTY, EMPTY, EMPTY],
      ];
      this.currentPlayer = PLAYER_X;
    }
  }
}
```

# Как храним данные?

Пользователей

```
const userSchema = new mongoose.Schema({
  username: { type: String, required: true, unique: true },
  password: { type: String, required: true },
  wins: { type: Number, default: 0 },
  losses: { type: Number, default: 0 },
  draws: { type: Number, default: 0 },
  game: { type: mongoose.Schema.Types.ObjectId, ref: 'TicTacToeGame' },
});
```

Игры

```
const gameSchema = new mongoose.Schema({
  players: [String],
  gameInstance: ticTacToeGameSchema,
  active: { type: Boolean, default: true },
  startTime: { type: Date, default: Date.now },
});
```

```
const ticTacToeGameSchema = new mongoose.Schema({
  board: [[String]],
  currentPlayer: String,
});
```

А вот и те самые изменения



# Слой работы с данными

Позволяет отделить логику от данных

```
gameSchema.methods.resetGame = function () {  
    const ticTacToeGame = new TicTacToeGame();  
    this.gameInstance.board = ticTacToeGame.board;  
    this.gameInstance.currentPlayer = this.players[0];  
    this.gameInstance.save(); // Save the updated game state  
    this.active = true;  
};
```

```
gameSchema.methods.applyMove = function (row, col, player_x, player_o) {  
    const ticTacToeGame = new TicTacToeGame(this.gameInstance);  
    const moveResult = ticTacToeGame.makeMove(row, col);  
  
    if (moveResult) {  
        if (!ticTacToeGame.isGameOver()) {  
            ticTacToeGame.switchPlayer(player_x, player_o);  
        }  
        this.gameInstance = ticTacToeGame.getState();  
        this.save(); // Save the updated game state to the database  
    }  
  
    return moveResult;  
};
```

# API

В целом, все методы выглядят подобным образом. Разве что с другой логикой

```
app.post('/api/start-game', verifyToken, async (req, res) => {
  try {
    const { player2 } = req.body;
    const player1 = req.user.username;

    const newGame = new Game({
      players: [player1, player2],
      gameInstance: {
        board: [
          [' ', ' ', ' ', ' '],
          [' ', ' ', ' ', ' '],
          [' ', ' ', ' ', ' '],
        ],
        currentPlayer: player1,
      },
    });

    await newGame.save();

    // Update users with game reference
    await User.updateMany(
      { username: { $in: [player1, player2] } },
      { $set: { game: newGame._id } }
    );

    res.status(201).json({ message: 'Game started successfully.', gameId: newGame._id });
  } catch (error) {
    res.status(400).json({ error: 'Failed to start a game. ' + error });
  }
});
```

Вот так запросы отправляются

POST http://localhost:80/api/start-game

Params Authorization Headers (9) Body Pre-request Script Tests Settings

☒ Connection keep-alive

☒ Authorization

Key

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6I6InNlcmdraHliLCJpYXQiOiE3MDMxOTM2NDIsImV4cCI6MTcwMzIzNjg0Mn0.97lkxrm9upAikLuMDFZ4hdUDVBxSaud6nH3zf6ohhMU

POST http://localhost:80/api/start-game

Params Authorization Headers (9) Body Pre-request Script Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL JSON

```
1 {
2   ... "player2": "admin"
3 }
```

# Игра

```
{
  "_id": "6584b0c30398b5e2ec640f5a",
  "players": [
    "sergkhr",
    "admin"
  ],
  "gameInstance": {
    "board": [
      [
        "sergkhr",
        " ",
        " "
      ],
      [
        "sergkhr",
        "admin",
        "admin"
      ],
      [
        "sergkhr",
        " ",
        " "
      ]
    ],
    "currentPlayer": "sergkhr",
    "_id": "6584b5cb0398b5e2ec640f6c"
  },
  "active": false,
  "startTime": "2023-12-21T21:40:19.468Z",
  "__v": 0
}
```

Это законченная игра

Да, в API поле выглядит странно, но нужно помнить, что это техническое представление

Пользователь не будет видеть игру именно таким образом. Но это остается на стороне клиента (что не подлежит разработке в этой курсовой)

# Перезапуск игры

Давайте перезапустим игру



Кстати, эта игра закончена

```
{
  "_id": "6584b0c30398b5e2ec640f5a",
  "players": [
    "sergkhr",
    "admin"
  ],
  "gameInstance": {
    "board": [
      [
        "sergkhr",
        " ",
        " "
      ],
      [
        "sergkhr",
        "admin",
        "admin"
      ],
      [
        "sergkhr",
        " ",
        " "
      ]
    ],
    "currentPlayer": "sergkhr",
    "_id": "6584b5cb0398b5e2ec640f6c"
  },
  "active": false,
  "startTime": "2023-12-21T21:40:19.468Z",
  "__v": 0
}
```

```
{
  "_id": "6584b0c30398b5e2ec640f5a",
  "players": [
    "sergkhr",
    "admin"
  ],
  "gameInstance": {
    "board": [
      [
        " ",
        " ",
        " "
      ],
      [
        " ",
        " ",
        " "
      ],
      [
        " ",
        " ",
        " "
      ]
    ],
    "currentPlayer": "sergkhr",
    "_id": "6584b5cb0398b5e2ec640f6c"
  },
  "active": true,
  "startTime": "2023-12-21T21:40:19.468Z",
  "__v": 1
}
```

Теперь игра снова активна

# Победа, ура

```
{
  "message": "Player sergkhr wins!",
  "winner": "sergkhr",
  "updatedGameState": {
    "board": [
      [
        "sergkhr",
        " ",
        " "
      ],
      [
        "sergkhr",
        "admin",
        "admin"
      ],
      [
        "sergkhr",
        " ",
        " "
      ]
    ]
  },
  "currentPlayer": "sergkhr",
  "_id": "6584b5cb0398b5e2ec640f6c"
}
```

Вот так выглядит сообщение, отправляемое, когда игра оканчивается победой

После конца игры ходить нельзя

```
Pretty Raw Preview Visualize JSON v
1 {
2   "error": "Game is not active."
3 }
```

# А ничья?

```
{
  "message": "It's a draw!",
  "winner": null,
  "updatedGameState": {
    "board": [
      [
        "sergkhr",
        "admin",
        "sergkhr"
      ],
      [
        "sergkhr",
        "sergkhr",
        "admin"
      ],
      [
        "admin",
        "sergkhr",
        "admin"
      ]
    ],
    "currentPlayer": "sergkhr",
    "_id": "6584d7fe30d6376a03ab5b10"
  }
}
```

Заполнили все поля  
значит — ничья

И игра не активна

```
{
  "_id": "6584b0c30398b5e2ec640f5a",
  "players": [
    "sergkhr",
    "admin"
  ],
  "gameInstance": {
    "board": [
      [
        "sergkhr",
        "admin",
        "sergkhr"
      ],
      [
        "sergkhr",
        "sergkhr",
        "admin"
      ],
      [
        "admin",
        "sergkhr",
        "admin"
      ]
    ],
    "currentPlayer": "sergkhr",
    "_id": "6584d7fe30d6376a03ab5b10"
  },
  "active": false,
  "startTime": "2023-12-21T21:40:19.468Z",
  "__v": 3
}
```

# Таблица рекордов

По запросу на получение  
таблицы рекордов будет получена  
таблица со всеми пользователями,  
отсортированными по победам

```
const users = await User.find({}, { _id: 0, username: 1, wins: 1, losses: 1, draws: 1 })  
  .sort({ wins: 'desc' });
```

GET http://localhost:80/api/users/leaderboard

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

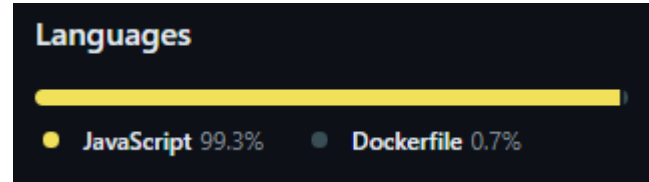
Key	Value
-----	-------

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "draws": 0,
4     "username": "test",
5     "wins": 4,
6     "losses": 1
7   },
8   {
9     "draws": 0,
10    "username": "test2",
11    "wins": 1,
12    "losses": 4
13  },
14  {
15    "username": "sergkhr",
16    "wins": 1,
17    "losses": 0,
18    "draws": 3
19  },
20  {
21    "username": "admin",
22    "wins": 0,
23    "losses": 1,
24    "draws": 3
25  }
26 ]
```

# Результаты

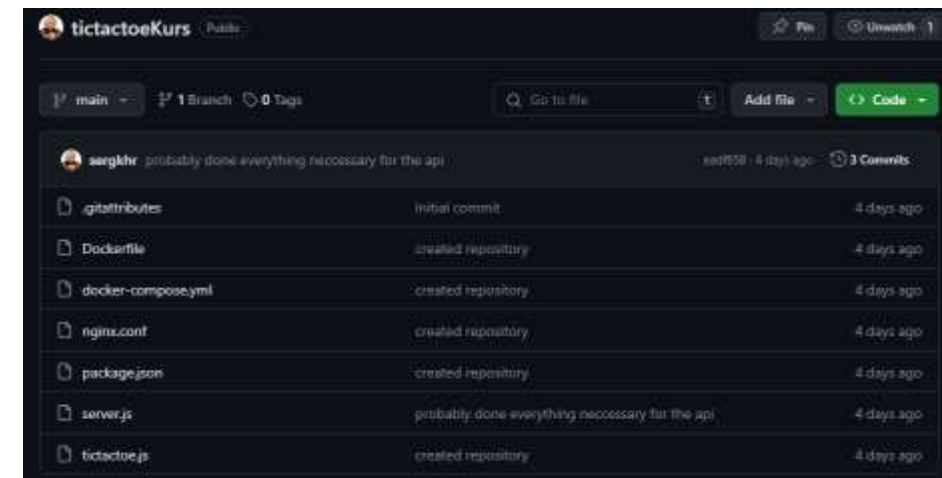


- Разработано веб-приложение «Крестики-нолики»
- Реализован паттерн Чистая архитектура
- База данных работает корректно
- Получен опыт веб-разработки серверных частей

## GitHub

Ссылка на исходный код:

<https://github.com/sergkhr/tictactoeKur>





Спасибо за внимание!