Functionele analyse dice roll

# Technologiën die gebruikt zijn

Raspberry PI, Python, Node JS, Remix (React server), Express, Cors, Sequelize, MariaDB, JSON en gpiozero

# Client-side/User interface

Gemaakt met Remix oftewel een React server-side framework en CSS.

# Server-side

Python server: Flask webserver met connectie tot de GPIO pins van de Raspberry PI.  
Node JS server: Express web server met connectie tot de MariaDB database.

# Integraties tussen componenten:

|  |  |
| --- | --- |
| Flask | <http://raspberrybouwen:5000> |
| Node JS | <http://raspberrybouwen:4000> |
| Remix | http://localhost:3000 |
| MariaDB | Raspberrybouwen:3306 |

Node JS -> MariaDB: Data ophalen  
Flask -> GPIO: Leds aansturen  
Remix -> Node JS: MariaDB data ophalen  
Remix -> Flask: Leds aansturen vanuit client-side

# Database model

Database diceroll:  
 Tabel Users:  
 Velden:  
 UserID: int primary key auto\_increment not null,  
 Username: varchar(255) not null,  
 timeswon: int not null

# Verloop dice roll

Stap 1: Homepage  
Stap 2:

**Game spelen**:

**Redirect**: /waiting  
**Redirec**t: /game  
 API Call: <http://raspberrybouwen:4000/getrandomnumbers>  
 API Call: [http://raspberrybouwen:5000/leds/(RandomNum 1)/(Random](http://raspberrybouwen:5000/leds/(RandomNum%201)/(Random)Num2)  
 => Blink leds  
 **Gebruiker voert naam in en klikt submit**:  
 API Call: <http://raspberrybouwen:4000/addUser>  
 body: { username: ( Ingevoerde naam )}  
 Check als de user bestaat:  
 Zo niet, maak een nieuwe  
 Zo wel, update de user  
 **Redirect**: /

**Highscores bekijken**:

API Call: <http://raspberrybouwen:4000/topusers>