Orange and white text on a black background

Description automatically generated

|  |  |
| --- | --- |
|  | **ASSESSMENT 1 – Group Assessment** |
| **Student Name** | Mohammaduzair Mohammadyahya Shaikh | A00126661 |
| Student Name | Mohammed Isa | A00118725 |
| **Subject Code and Title** | PBT205—Project-based Learning Studio: Technology |
| **Assessment** | Prototype Development |
| Learning Facilitator | Nyasha Shoniwa |

# Task – 3 Contact Tracing App

### Application Overview

Using RabbitMQ, this application maintains the positions of users and replicates their movements on a grid. Position updates are sent to a tracker, which monitors the grid for interactions between various users. This simulation offers a realistic setting for learning about message queues and grid-based simulations. It includes real-time messaging, validation, and error handling.

### Functionality

Person Simulation: A person's movements on a grid are simulated using the person.js file. Every second, it starts a grid with a scalable size and transfers the subject to arbitrary locations. Updates on positions are delivered to a RabbitMQ queue.

Position Tracking: Every user's position updates are listened for by the tracker.js code. It checks to see if users on the grid meet one another and updates an in-memory object with the most recent positions.

Grid Management: Positions are checked to make sure they are inside the grid by using the grid.js file, which also controls grid boundaries. Additionally, it generates spots at random for the user to move to.

Error Handling: RabbitMQ connection errors, application errors, and validation problems are handled by custom error classes and functions defined in the errorHandler.js file. It makes ensuring that processes shut down gracefully when needed and that thorough error messages are recorded.

### Executing the Application

Install required Dependencies: Install the Node.js package.

**Terminal: npm install**

Start the Tracker: Execute the tracker.js by this command to listen for the position updates.

**Terminal: node src/tracker/tracker.js**

Start the Person Simulation: Execute the person.js Script to simulate the person’s movement.

**Terminal: node src/person/person.js <username> <gridSize>**

**Terminal: node src/person/person.js Uzairr 10**

### Error Handling

There are several levels of robust error handling in the program. Invalid user inputs, setup difficulties, and RabbitMQ connection problems are logged with comprehensive error messages. This guarantees that the system can maintain consistent states and gracefully handle faults.

### Usefulness

This application offers a practical way to learn about error management, grid-based simulations, and real-time messaging. It is an effective teaching tool for people interested in system design and message technologies since it can be used to learn and test scenarios involving user interactions on a grid.

# References:

**RabbitMQ Documentation Team. (n.d.). *RabbitMQ documentation* [Website]. Retrieved from**[**https://www.rabbitmq.com/docs**](https://www.rabbitmq.com/docs)

[**Node.js**](https://node.js/)Foundation. (n.d.). [***Node.js***](https://node.js/)*documentation* [Website]. Retrieved from [**https://nodejs.org/en/docs/**](https://nodejs.org/en/docs/)

Docker Inc. (n.d.). *Docker documentation* [Website]. Retrieved from [**https://docs.docker.com/**](https://docs.docker.com/)

# Project Overview

**Project Name:** Contact Tracing Application Using RabbitMQ and Node.js

**Project Manager:** Mohammaduzair & Mohammed Isa

**Start Date: 3 October**

**End Date: 26 October**

### Purpose

The goal of this project is to use Node.js and RabbitMQ to simulate a contact tracing system. The application uses message brokering and a grid-based simulation to watch people's movements on a grid and keep an eye out for any contact events in real-time.

### Objectives

* Develop a contact tracing simulation to tract users on a grid.
* Use RabbitMQ for real-time position updates and message handling.
* Enable error handling for reliable data communication and application stability.

### Scope

The scope encompasses the creation of the contact tracing application, the design of the grid environment, the setting up of RabbitMQ for message queuing, and thorough testing of contact detection and tracking.

### Deliverables

* Contact tracing application code for position tracking and contact detection.
* Documentation covering setup, functionality, error handling, and user instructions.
* Final report outlining the application overview, error handling strategies, and the application’s usefulness.

### Stakeholders

Mohammaduzair: Project Manager and Reviewer

Mohammed Isa: Developer and Designer

### Risks

* Dependency Issues: Possible problems setting up RabbitMQ.
  + Mitigation: Verify that environment and Docker variables are set up appropriately.
* Connectivity Issues: RabbitMQ may fail to connect if the server is not running. Mitigation: Document steps to troubleshoot RabbitMQ setup.

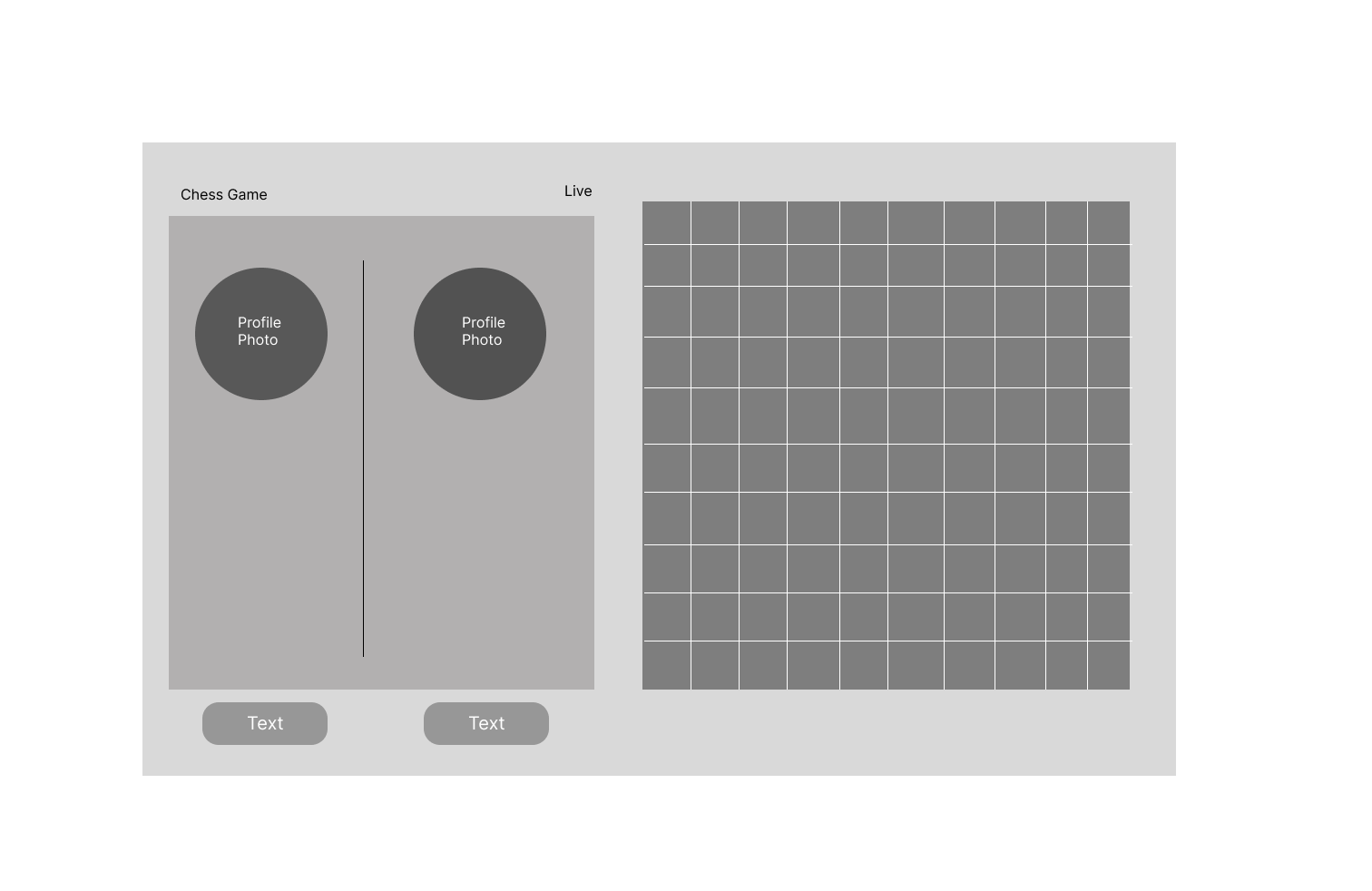
### Team Members

|  |  |
| --- | --- |
| Name | Role |
| Mohammaduzair Mohammadyahya Shaikh | Project Manager and Reviewer |
| Mohammed Isa | Developer and Designer |

Ui and Screenshots of functionality

A screenshot of a game

Description automatically generated



A screenshot of a computer program

Description automatically generated