Tournaaro
Tournament Management Tool

Applied Project & Minor Dissertation

STUDENT: EOIN LERNIHAN

SUPERVISOR : JOSEPH CORR

Agenda

- Project Overview
- Objectives
- Architecture
- System Design
- System Evaluation / Follows-on
- Conclusion

Project Overview

Scope: Serverless webapp to support the management of game tournaments . Serving both small business and customer in an affordable, reliable and simple manner.

Technical Aspects: Java 11/8, JSON, REACT, LAMBDA, MongoDB, CORS,API Gateway, CLOUDWATCH

Status : On Track !

Objectives

Advertisement

- Opportunity for game shops to promote tournaments that they are hosting to a target audience

Time mangement

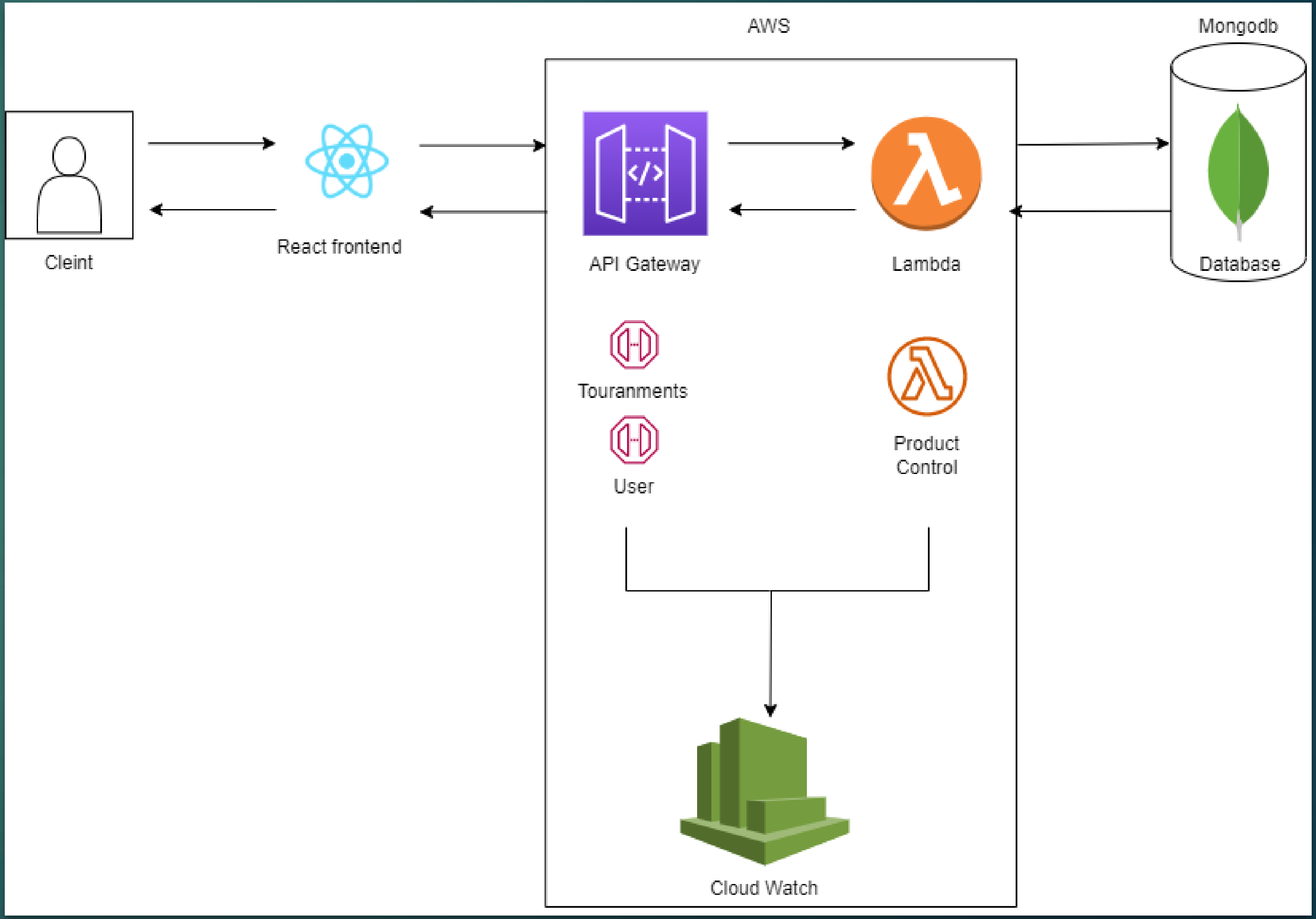
- Allows shop admins to keep track of upcoming tournaments
- Customer are able to join, managed tournaments

ServerLess

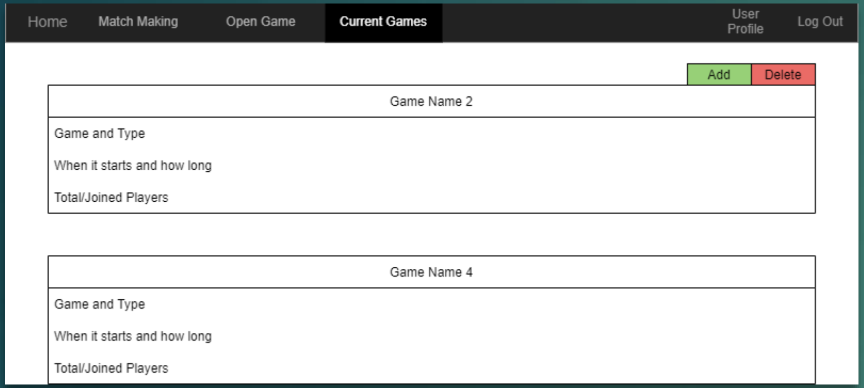
- To show that server-less is cheaper for small time companies
- To show that it is adequately for the task
- To show technique for addressing cold starts

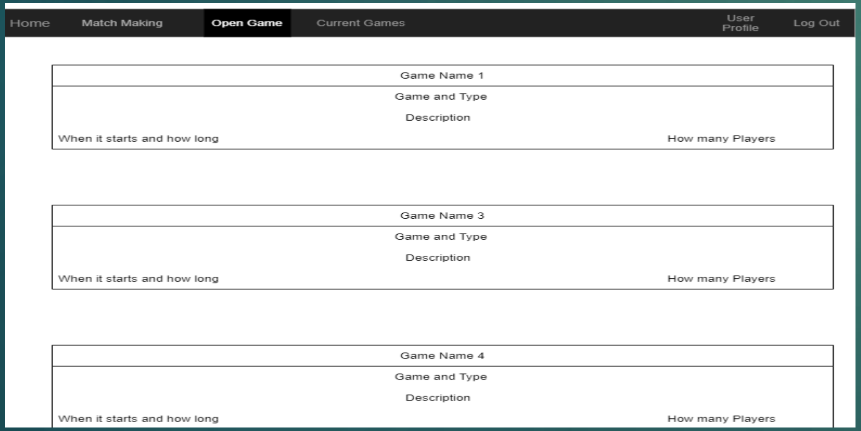
Architecture

1. Client
2. React frontend
   - Wireframes
   - Players to join games
   - Shop to set up tournaments
3. API Gateway (REST API)
   - Directs http traffic form frontend to lambda
   - Json
4. Lambda
   - Serv er-less
   - Jav a (Maven)
   - Writes and reads data form Mongodb
5. MongoDB for the Database
   - Stores data
6. Cloud Watch
   - System console in the Cloud

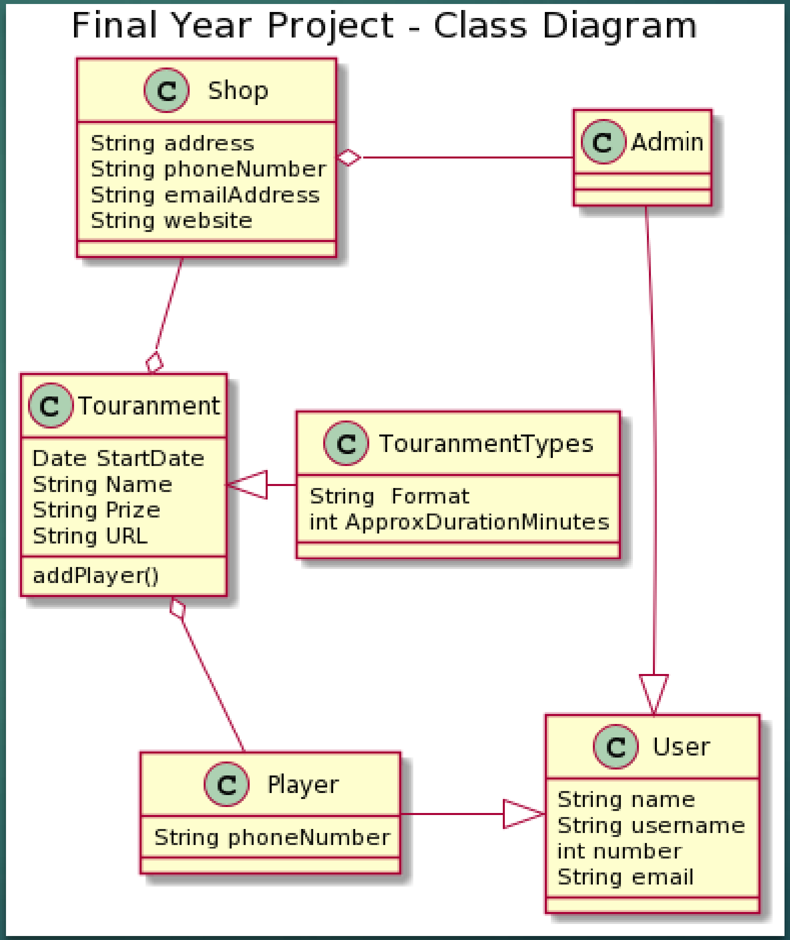


System Design





System Mockup



System Components

1. Shop
   - Owns and creates (via their admin) tournaments
2. Players
   - Is a type of user
3. Tournaments
   - Show

System Evaluation/ Follow-ons

- Investigated lambda and researched serverless
- Able to send a get request form the react client side and able to receive inform form the Mongo dB using a serverless Aws Lambda and API Gateway
  - Reduce the cold start of 8 - 9seconds(8988 avg) down closer to a hot start of 801.37 ms
    - https://www.capitalone.com/tech/cloud/aws-lambda-java-tutorial-reduce-cold-starts/
- Using Maven to build a Fat Jar for AWS Lambda
- Configured API Gateway (json) to interface with Lambda
- Design React Client Application & ensure functionality
  - Need to investigate AWS Amplify in order to go serverless for the React Client on web.
- Evaluated the cost of running serveless it is quite cheap
  - Using free AWS tier for AWS Lambda, API Gateway (cloudwatch 5GB Data)
  - Mongodb will only charge is 27 cents per every million uses

Conclusion

- Conducted productive research into serverless – its benefits & implementation
- Developed a working base in which I can **pull** and **get post requests** using serverless
- Concrete scope & development plan for next semester in what is needed to achieve objectives outlined.
- Draft Wireframes to compliment scope and overall vision
- Combined learnings of new technologies (Serverless & Lambdas) with elements from previous modules.