LineUp Software Requirements and Architecture

Team Aphrodite

Purpose

Application for creating, managing and joining queues on the fly

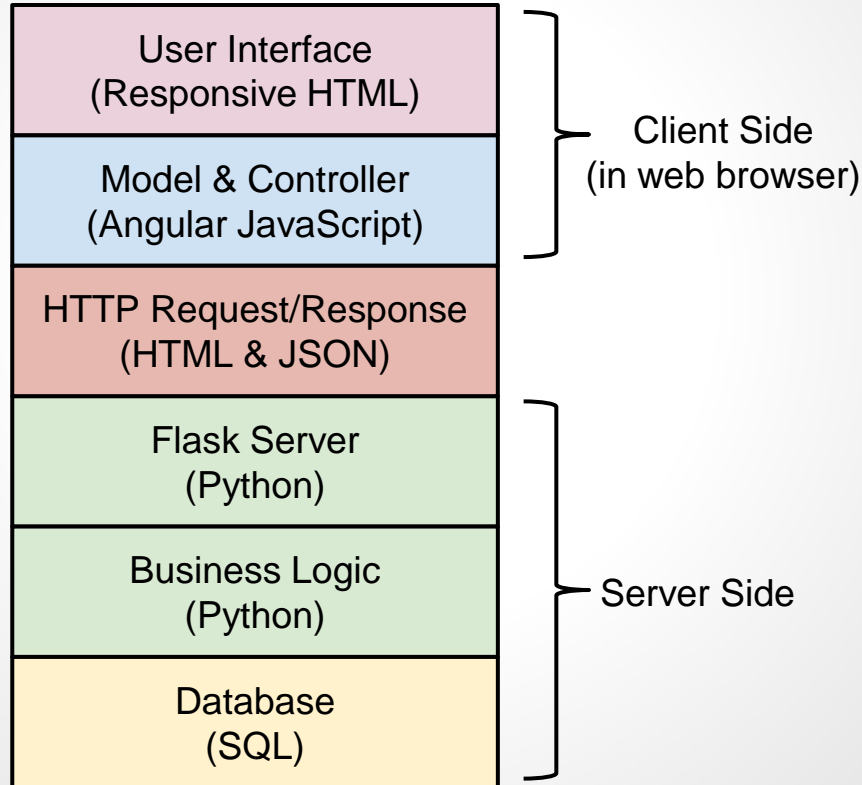# Features

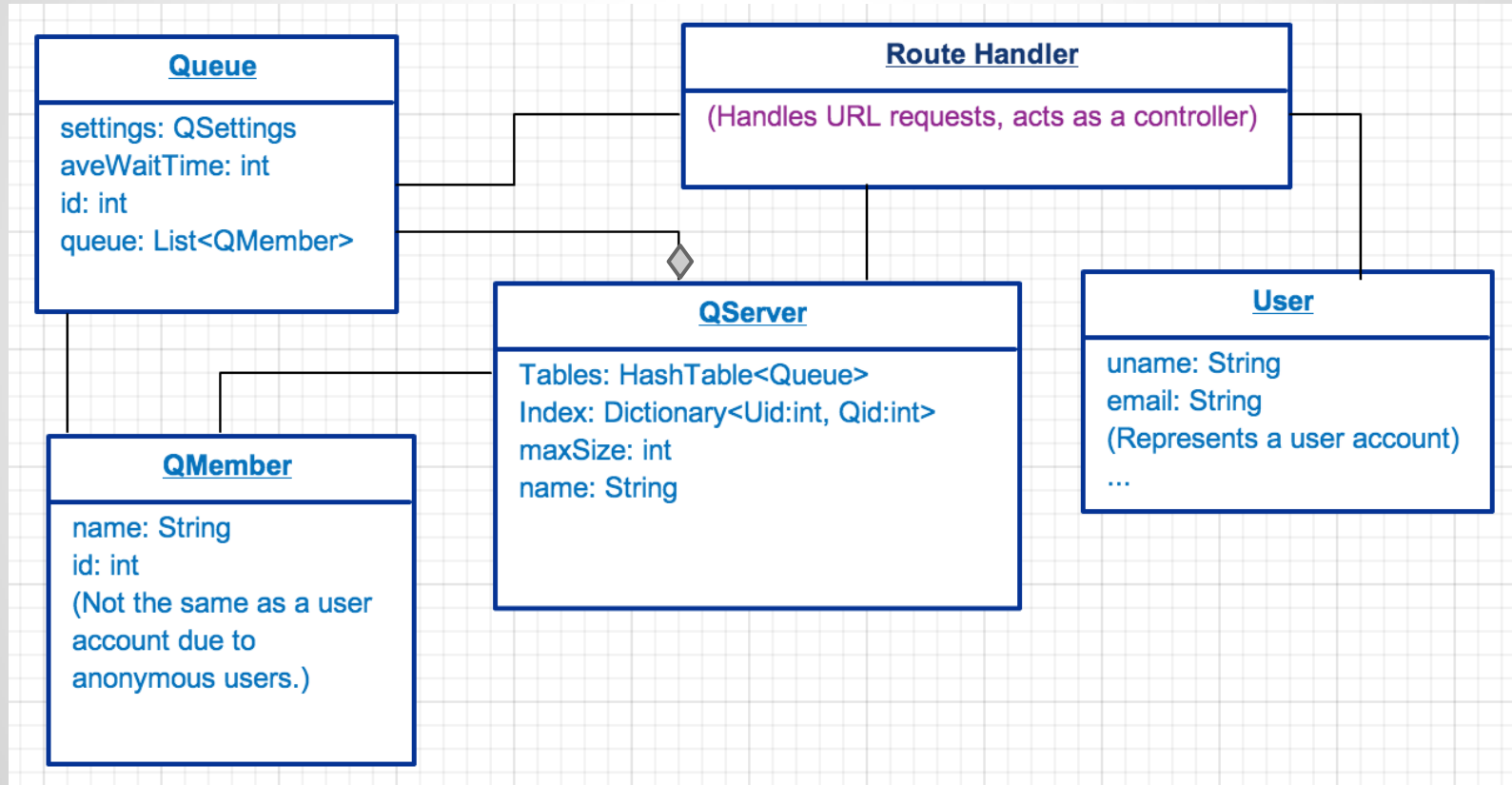- Persistent Queues
- Position Control
- Analytics

Design Goal

- Versatility

Planned Implementation

Simplified Class Diagram

Queue

settings: QSettings
aveWaitTime: int
id: int
queue: List<QMember>

Route Handler

(Handles URL requests, acts as a controller)

QServer

Tables: HashTable<Queue>
Index: Dictionary<Uid:int, Qid:int>
maxSize: int
name: String

User

uname: String
email: String
(Represents a user account)
...

QMember

name: String
id: int
(Not the same as a user
account due to
anonymous users.)

Design Decisions

Anonymous users vs accounts

- Displaying user names in the queues vs. only displaying position

- Generating a user id for anonymous users to allow them to access the queue they joined.

Risk Handling

1. Handling member types
2. Choosing the right framework
3. Organizing teamwork & platform consistency
4. Managing feature creep
5. Implementing user accounts in zero-feature

Timeline/Team Structure

- Zero-Feature

- Beta Release

- Feature-Complete Release

- Release Candidate

- 1.0 Release

Testing

- Server side unit testing and stress testing

- Client side system tests

- Usability tests

LineUp

We know you will all be lining up to use our product...