P2 ImRec Project Planning Document

# Team Members

Emma Sutton

Kieron “Special K” Gillingham

Matt Dear

# Communication Methods

1. Mattermost Group
2. Discord Chat

# Research

* ANPR library <https://github.com/oskopek/javaanpr>
* JMS (Java Messaging System) <https://www.javatpoint.com/jms-tutorial>
* GitHub Actions/Circle CI
* GitHub Projects
* Docker/Docker Compose <https://docs.docker.com/compose/>

# Messages

**IN**: Image(ASCII), Image Reference, Camera Number, Timestamp

**OUT**: Number Plate, Image Reference, Camera Number, Timestamp

# Deployment

* Requirements of ANPR library (Java 1.5 or above)
* Docker/JAR?
* Docker Compose?
* Maven
* Ubuntu?

# Testing Requirements

* Selection of images to test with
* Working ActiveMQ server to test on
* Access to Java ANPR library

# Tests:

* Test scaling up on queue length
* Test tearing down on queue length
* Test JMS messages in and out
* Test getting the image from the API using the Image ID
* Test image recognition

# Questions

1. P1(Image Receiving) - What are they giving us/what we need from them?
2. How are P1(Image receiving) converting images into an ascii format. We will be getting a URL of the image
3. P3(Reconcile Charging), P8 (Check MOT), P9 (Check Speeding) - What we can offer them and what they would like from us?
4. P4(DevOps) - How would they like us to scale? How to package out microservice?
5. What implementation of JMS will be using and whose decision is that? (<http://activemq.apache.org/>) Implementation of JMS needs to support queue length information to allow us to scale.

# Functional Requirements

1. Receive messages from P1(Receiving Image Team) queue.
2. Talk to P1 API to get image of number plate
3. Deconstruct the serialised object
4. Recognise the number plate and add this to the object.
5. Serialise the object with the number plate attached
6. Send objects as a topic.
7. Scale on high traffic.

# Non Functional Requirements

1. Tear down on low traffic from P1(Receiving Image Team) based on what is in the queue. (Docker/VMs)

Things to do in meeting:

* Set up Github project? / No we are waiting for Craig to do this...
* Create diagram to explain our part of the system / Done Basic Diagram
* Create data flow diagram for our part of the system? / Done Message Diagram
* Complete non functional and functional requirements / Done
* Create a timeline of tasks that need to be completed and in what order. / Done Jira

First Presentation:

* How your component fits into the overall system
* Initial System model – information model, messaging and components (Class diagram)
* Functional and non-functional feature requirements
* Test requirements
* Deployment requirements
  + Describe the precise, desired configuration of a software system. They relate the system's non functional **requirements** to its architecture, providing a basis for making decisions about design trade-offs in terms of the resulting system's non functional properties.
* Outline project plan

Presentation one:

# How our part of the system fits into the overall project:

Questions to be asked

Who will code and de code the images?

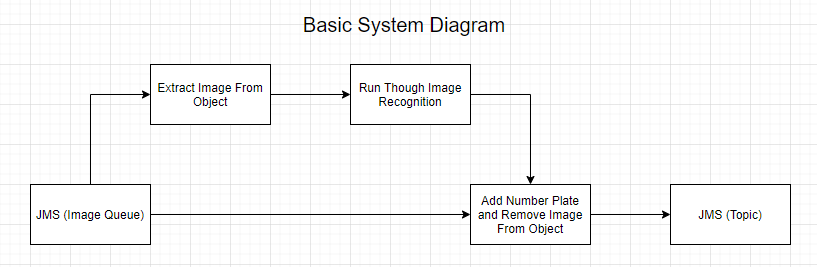
# Deployment Requirements

* ActiveMQ server
* Deployment location
* Access to Java ANPR library
* We need to make a docker image for use downstream
* Need a properties file that can be injected to talk to the ActiveMQ server.
* Properties can setup how many threads we are running.

# Class Diagram:

# Use Case Diagrams:

# Message Model:



# Outline of Project Plan: