Case Study

Group: RDEnvi

Robertas Asvicas

Dmitry Lvov

**Contents**

[Project definition 3](#_Toc62399183)

[Requirements 3](#_Toc62399184)

[Infrastructure Diagram 4](#_Toc62399185)

[Infrastructure Components 5](#_Toc62399186)

[VPC 5](#_Toc62399187)

[Subnets 5](#_Toc62399188)

[NAT subnet: 5](#_Toc62399189)

[Private subnet 5](#_Toc62399190)

[Database subnet: 5](#_Toc62399191)

[Cloudwatch Insights 6](#_Toc62399192)

[Billing 6](#_Toc62399193)

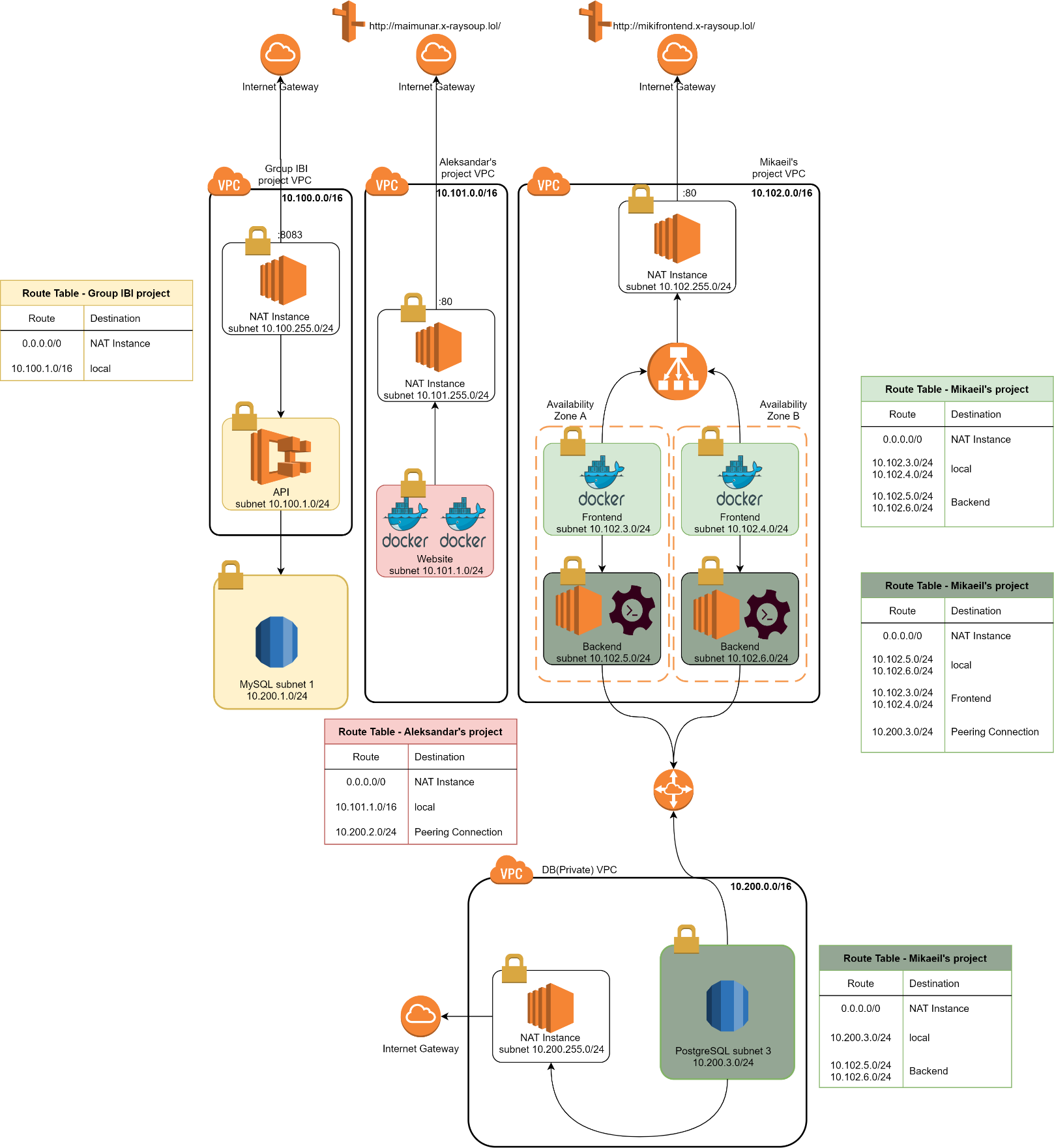
[MoSCoW 7](#_Toc62399194)

# Project definition

Our main goal in this project was to create suitable infrastructure for IBI software students group in order to host API and database for their application. Infrastructure is supposed to be scalable and secure. We were using Amazon Web Services (further as AWS) for infrastructure creation in a conjunction with Docker for scalability.

# Requirements

* Network Infrastructure on AWS
  + Secure – usage of private subnets, private VPCs, NAT instances, configured Security Groups
  + Scalable – usage of Autoscaling Groups and Containers
  + Monitored – usage of Cloudwatch container insights
* Isolated Database (Public)

Infrastructure Diagram

Infrastructure Components

## VPC

Infrastructure is made out of 1 VPCs :

|  |
| --- |
| Main VPC |
| CIDR block: 10.100.0.0/16 |
| Used to host everything on the infrastructure |
| NAT subnet can be accessed from WAN |
| Has Internet Access |
| ECS container instances can be accessed only through SSM or SSH connection from NAT Bastion |
| Implemented ECS cluster for advanced scalability |

## Subnets

There are 3 types of subnets in our infrastructure:

* NAT subnet
* Private subnet
* Database subnet

### NAT subnet:

Is hosting **NAT** **instance** which serve as a **middleman** between **Private** **subnet** and **Internet**. **Route Table** is configured that NAT subnet instances see Internet Gateway (further as IGW) – route to the Internet and the machine + database in private subnet. Has its own **Security Group**, which allows all traffic Inbound into NAT subnet to all ports.

Private subnet:

Are hosting **API** (tomcat). **Route Table** is configured that it sees NAT instance – route to IGW and local VPC instances. Has its own **Security Group**, which allows port **80** and port **443** (HTTP and HTTPS) traffic from NAT instances. **IP** addresses **range** varies **from 10.100.1.0/24 to 10.100.6.0/24.**

### Database subnet:

Database subnet us hosting **database instance**. **Route Table** is configure that it sees IGW – route through NAT instance, machines in the private subnet and local VPC instances. Has its own **Security Group**, which allows port **80** and port **443** traffic. **IP addresses** range varies **from 10.200.1.0/24 to 10.200.3.0/24**.

## Cloudwatch Insights

**All** traffic that is going through EC2 instance that hosts Docker containers is logged through Cloudwatch container monitoring which allowed us to create dashboards for container tasks and structure the logs.

# Billing

Whenever budget exceeds 4.00USD an alarm message is sent to the root account e-mail ([r.asvicas@student.fontys.nl](mailto:r.asvicas@student.fontys.nl)).

# MoSCoW

**Must have:**

* Main VPC
* NAT instance
* Internet Gateway
* 3 Subnets
* ECS(EC2) instance to host containers
* Cloudwatch Insights

**Should have:**

* Systems manager
* Cloudwatch Insights logging

**Could have:**

* Separate VPC for database
* Peering connection
* Fluentd logging implementation in a conjunction with elastic.co

**Won’t have (this time):**

* Site to Site VPN connection
* Terraform configuration
* Ansible automation
* Transit gateway
* Auto scaling groups