

Florida International University
School of Computing and Information Sciences

Software Engineering Focus

Final Deliverable

Project Title: Cloud Enabling Work Queue

Team Members: Osniel Valdivia
Francisco Espinosa Castillo
Product Owner(s): Norbert Monfort
Mentor(s): Michael Reyerros
Instructor: Masoud Sadjadi

The MIT License (MIT)

Copyright (c) 2016 *Florida International University*

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Abstract

Florida International University's College of Engineering and Computer Science is working with Assurant, Inc., which is a global company that provides a diverse set of insurance products. The target is to create a cloud based work queue that they could implement in their facilities.

This project was built using REST API and Docker. It is also an open source and can be deployed privately on premise, or it can be used through our cloud platform which is accessible through the internet. You will find the documentation about the first version and attempt for this project. This packet includes the Scrum software development procedure used by this team, implemented user stories are described, justified with use cases and sequence diagrams, and tested with system test cases. This document also includes: User Stories, Project Plan, System Design, among others that were present in the project development. You can also find a section on this document where the non-implemented features will be stated as a stand point for any future developer that will continue to work on this project.

Table of contents

Introduction.....	4
Current System.....	5
User Stories.....	6
Implemented User Stories.....	6
Pending User Stories.....	10
Project Plan.....	11
Hardware and Software Resources.....	11
Sprints Plan.....	11
Sprint 1.....	11
Sprint 2.....	12
Sprint 3.....	13
Sprint 4.....	14
Sprint 5.....	15
Sprint 6.....	16
System Design.....	17
Architectural Patterns.....	17
System and Subsystem Decomposition.....	17
Deployment Diagram.....	18
Design Patterns.....	18
System Validation.....	19
Glossary.....	21
Appendix.....	22
Appendix A - UML Diagrams.....	22
Appendix B - User interface Design.....	24
Appendix C - Sprint Review Reports.....	25
Appendix D - User Manuals, Installation/Maintenance Document/Shortcomings/Wishlist Document and Other Documents.....	32
References.....	32

Introduction

Companies often have a tremendous amount of support tickets, where they are efficiently queued and the workload is unevenly distributed. Queue systems are used to manage workload in a more efficient manner; however, some of these queue systems are not designed for companies that take their resources from different sources and cannot handle large amounts of data. This project is a cloud-based solution originated by our product owner who works at Assurant, Inc., to the modern ticketing workflow problem.

Here we present the first version of this project, which is open source and can scale its backend to support most companies' use case. It can also be deployed privately on premise or through our cloud platform which is accessible through the internet.

Current System

Fall 2019 is the first version of the project. The main priority of the system is so developers can work on the project with test-driven development and proper documentation. These features are very sought after in the development community but are virtually unattainable once a project incurs significant technical debt. Technical debt comes from prioritizing business features before sustainable software. This incurs a mindset of ‘if it works, that’s all that matters’ which can quickly kill the maintainability of the project, leading to refactoring and wasted time. This project is different. We follow the principles of Dev Ops to provide a stable system to work on. A system that once you make a code change there is a test suite that tells you what routes are failing. A system that provides integration built into the pipeline. Raising a system with these principles at birth is how you get highly-scalable and maintainable system that are so powerful they become a software standard. A self-debugging system, where exceptions and errors are properly given to developers instead of silently failing with an obscure error.

The current system is an Azure App Services Docker container that houses a Node API. There is user authentication as well as user authorization based on user roles. The Node API contains routes for user registration, company registration, and a Swagger UI. Swagger is a documentation framework based on the OpenAPI standard; our entire API is documented using this software. Our application has continuous integration with our docker image that contains a full test suite of all our routes which run whenever our application is built. This means that our deployment pipeline is not tied to any cloud vendor, which means a company can deploy this application whether they use Azure, Aws, Google Cloud Platform, etc. The summary of technologies used are MongoDB Atlas data store, a NodeJS Express REST backend, Docker containerization, chai, mocha, and Supertest for automated testing of REST endpoints, and Open API doc spec with a hosted Swagger UI.

User Stories

Implemented User Stories

CEWQ0001 - Setup cloud database - As a user, I would like to have a cloud database so that I can persist data and access data from anywhere.

CEWQ0002 - Create admin data model - As an admin, I would like to have a data model representation so that I can access the system in the way intended for admins.

CEWQ0003 - Create ticket data model - As a user, I would like to have a data model representation of a ticket so that I can store and access relevant ticket information in the system.

CEWQ0004 - Create employee data model - As an employee, I would like to have a data model representation so that I can access the system in the way intended for employees.

CEWQ0005 - Create owner data model - As an owner, I would like to have a data model representation so that I can access the system in the way intended for owners.

CEWQ0006 - Create customer data model - As a customer, I would like to have a data model representation so that I can access the system in the way intended for customers.

CEWQ0007 - Create company data model - As a user, I would like to have a data model representation of a company so that I can access the system in the way intended for entities corresponding to that company.

CEWQ0008 - Create event data model - As a user, I would like to have a data model representation of an event so that I can have a history of events during a ticket's lifespan.

CEWQ0013 - Add authentication - As a user, I would like to have authentication so that I can access my data and no one else can access data they are forbidden from.

CEWQ0014 - Design landing page mockup - As a designer, I would like to have an initial landing page mockup so that I can do a refined mockup with software.

CEWQ0009 - Create company api - As a user, I would like to have RESTful API routes so that I can get data from the backend using HTTP requests from a front-end client.

CEWQ0015 - Document user api - As a user, I would like to have Swagger documentation of api routes, parameters, request bodies, responses, and authentication so that I can have a reference on how to interact with the backend.

CEWQ0016 - Document company api - As a user, I would like to have Swagger documentation of api routes, parameters, request bodies, responses, and authentication so that I can have a reference on how to interact with the backend.

CEWQ0017 - Create Swagger yaml - As a user, I would like to have a Swagger documentation yaml so that I can have an OpenAPI standard reference on how to communicate with the backend system.

CEWQ0018 - Document company data model - As a user, I would like to have a Swagger documentation of this data model so that I can represent this model as json through the API.

CEWQ0019 - Document user data model - As a user, I would like to have a Swagger documentation of this data model so that I can represent this model as json through the API.

CEWQ0020 - Document ticket data model - As a user, I would like to have a Swagger documentation of this data model so that I can represent this model as json through the API.

CEWQ0021 - Document event data model - As a user, I would like to have a Swagger documentation of this data model so that I can represent this model as json through the API.

CEWQ0022 - Postman collection for company - As a user, I would like to have a Postman collection of this abstraction so that I can send json data to the API.

CEWQ0028 - NodeJs for API - As a user, I want to have a fast web-scalable application so I want to use NodeJS for the API back end.

CEWQ0030 - Design Ticket API - As a user, I would like to have a RESTful API routes for tickets so that I can get ticket data from the back-end from a front-end client.

CEWQ0031 - Serve Swagger UI on API - As a user, I would like to have a Swagger OpenAPI UI of the API routes so that I can have access to a standard documentation that shows how to interact with the api.

CEWQ0032 - Add npm clean script for packages - As a user, I would like to have a package management script so that I can try out packages and effectively remove them and solve any conflicts.

CEWQ0033 - Add automated database setup for tests - As a user, I would like to have the test database automatically setup so that I can run stateless automated tests that do not rely on manual data.

CEWQ0034 - Restrict owners to only have one company - As a user, I would like to restrict owners to only have one company so that I can control how many companies I have in the system.

CEWQ0035 - Fix incorrect HTTP status code for company already existing - As a user, I would like to get the proper http response on an invalid request so that I can properly know what went wrong in the request.

CEWQ0036 - Fix incorrect HTTP status code for owner already having a company - As a user, I would like to get the proper http response on an invalid request so that I can properly know what went wrong in the request.

CEWQ0037 - Develop test suite for company post route - As a user, I would like to automatically test api route so that I can make changes to code without breaking expected functionality.

CEWQ0038 - Develop test suite for company get route - As a user, I would like to automatically test api route so that I can make changes to code without breaking expected functionality.

CEWQ0039 - Develop automated test for customer registration - As a user, I would like to automatically test api route so that I can make changes to code without breaking expected functionality.

CEWQ0040 - Develop automated test for owner registration - As a user, I would like to automatically test api route so that I can make changes to code without breaking expected functionality.

CEWQ0041 - Create Dockerfile - As a user, I would like to have a Dockerfile so that I can build highly-scalable, high-availability architecture that is cloud platform agnostic.

CEWQ0042 - Add .dockerignore - As a user, I would like to have a .dockerignore file so that I can prevent having sensitive information inside a container running on a cloud system.

CEWQ0043 - Create Dockerfile stages - As a user, I would like to have Dockerfile stages so that I can manage dev, test, and production environments.

CEWQ0044 - Create Dockerhub account - As a user, I would like to have a Docker hub account so that I can manage my Docker images remotely to host them on cloud providers.

CEWQ0045 - Add continuous integration - As a user, I would like to have continuous integration so that I can only make secure and expected changes to production code.

CEWQ0046 - Document Docker pull and run commands - As a user, I would like to have documentation of the docker commands so that I can have a reference on how to interact with our docker system.

CEWQ0047 - Document Docker build and push commands - As a user, I would like to have documentation of the docker commands so that I can have a reference on how to interact with our docker system.

CEWQ0048 - Document Docker remove all containers and their volumes, and delete all images commands - As a user, I would like to have documentation of the docker commands so that I can have a reference on how to interact with our docker system.

CEWQ0049 - Document issue with Docker bash login - As a user, I would like to login to docker via the bash cli so that I can interact with our docker system.

CEWQ0050 - Deploy Docker App to Azure - As a user, I would like to deploy my docker app to the cloud so that I can access my application through the internet.

CEWQ0051 - Create Linux resource - As a user, I would like to have a host os so that I can run my docker container on the cloud.

CEWQ0052 - Create subscription resource - As a user, I would like to pay for my cloud docker app so that I can keep running it with high availability.

CEWQ0054 - Add nom security audit to docker build - As a user, I would like to audit my dependencies on a docker build so that I can fail the build should a vulnerability be exploitable.

CEWQ0055 - Optimize Docker Build - As a user, I would like to have an optimal Docker build so that I can correctly use all of the features of Docker.

CEWQ0056 - Collect all Credentials - As a user, I would like to have all of the credentials of the system so that I can use the cloud based resources next semester.

CEWQ0057 - Record Instruction Videos - As a user, I would like to have instructions of the system so that I can know how to continue building on it.

CEWQ0058 - Document UML Diagrams - As a user, I would like to have UML documentation of the system so that I can have an industry standard overview of the project.

Pending User Stories

CEWQ0010 - Create ticket API - As a user, I would like to have RESTful API routes for tickets so that I can get ticket data from the backend using HTTP requests from a front-end client.

CEWQ0011 - Create employee API - As an employee, I would like to have RESTful API routes so that I can get data from the backend using HTTP requests from a front-end client.

CEWQ0012 - Create admin api - As an admin, I would like to have RESTful API routes so that I can get data from the backend using HTTP requests from a front-end client.

CEWQ0023 - Postman collection for employees - As a user, I would like to have a Postman collection of this abstraction so that I can send json data to the API.

CEWQ0024 - Postman collection for users - As a user, I would like to have a Postman collection of this abstraction so that I can send json data to the API.

CEWQ0025 - Postman collection for admins - As a user, I would like to have a Postman collection of this abstraction so that I can send json data to the API.

CEWQ0026 - Postman collection for tickets - As a user, I would like to have a Postman collection of this abstraction so that I can send json data to the API.

CEWQ0027 - Postman collection for events - As a user, I would like to have a Postman collection of this abstraction so that I can send json data to the API.

EPIC - Webhooks - As a user, I would like to have web hooks of this abstraction so that I can send real time updates to users.

Project Plan

In this section you will find the hardware and software that is involved in the generation of this project, as well as the implemented stories divided into their designated sprints.

Hardware and Software Resources

Hardware:

- Any computer that runs Windows, MacOS or Linux
- Cloud-based provider to deploy in the internet

Software:

- Git
- NodeJS
- Docker
- CMD, terminal or bash
- Code editor (Visual Studio)
- MongoDB atlas instance

Sprint Plan

Sprint 1:

CEWQ0001 - Setup cloud database - As a user, I would like to have a cloud database so that I can persist data and access data from anywhere.

CEWQ0002 - Create admin data model - As an admin, I would like to have a data model representation so that I can access the system in the way intended for admins.

CEWQ0003 - Create ticket data model - As a user, I would like to have a data model representation of a ticket so that I can store and access relevant ticket information in the system.

CEWQ0004 - Create employee data model - As an employee, I would like to have a data model representation so that I can access the system in the way intended for employees.

CEWQ0005 - Create owner data model - As an owner, I would like to have a data model representation so that I can access the system in the way intended for owners.

CEWQ0006 - Create customer data model - As a customer, I would like to have a data model representation so that I can access the system in the way intended for customers.

CEWQ0007 - Create company data model - As a user, I would like to have a data model representation of a company so that I can access the system in the way intended for entities corresponding to that company.

CEWQ0008 - Create event data model - As a user, I would like to have a data model representation of an event so that I can have a history of events during a ticket's lifespan.

CEWQ0013 - Add authentication - As a user, I would like to have authentication so that I can access my data and no one else can access data they are forbidden from.

CEWQ0014 - Design landing page mockup - As a designer, I would like to have an initial landing page mockup so that I can do a refined mockup with software.

Sprint 2:

CEWQ0009 - Create company api - As a user, I would like to have RESTful API routes so that I can get data from the backend using HTTP requests from a front-end client.

CEWQ0015 - Document user api - As a user, I would like to have Swagger documentation of api routes, parameters, request bodies, responses, and authentication so that I can have a reference on how to interact with the backend.

CEWQ0016 - Document company api - As a user, I would like to have Swagger documentation of api routes, parameters, request bodies, responses, and authentication so that I can have a reference on how to interact with the backend.

CEWQ0017 - Create Swagger yaml - As a user, I would like to have a Swagger documentation yaml so that I can have an OpenAPI standard reference on how to communicate with the backend system.

CEWQ0018 - Document company data model - As a user, I would like to have a Swagger documentation of this data model so that I can represent this model as json through the API.

CEWQ0019 - Document user data model - As a user, I would like to have a Swagger documentation of this data model so that I can represent this model as json through the API.

CEWQ0020 - Document ticket data model - As a user, I would like to have a Swagger documentation of this data model so that I can represent this model as json through the API.

CEWQ0021 - Document event data model - As a user, I would like to have a Swagger documentation of this data model so that I can represent this model as json through the API.

CEWQ0022 - Postman collection for company - As a user, I would like to have a Postman collection of this abstraction so that I can send json data to the API.

CEWQ0028 - NodeJs for API - As a user, I want to have a fast web-scalable application so I want to use NodeJS for the API back end.

CEWQ0030 - Design Ticket API - As a user, I would like to have a RESTful API routes for tickets so that I can get ticket data from the back-end from a front-end client.

Sprint 3:

CEWQ0031 - Serve Swagger UI on API - As a user, I would like to have a Swagger OpenAPI UI of the API routes so that I can have access to a standard documentation that shows how to interact with the api.

CEWQ0032 - Add npm clean script for packages - As a user, I would like to have a package management script so that I can try out packages and effectively remove them and solve any conflicts.

CEWQ0033 - Add automated database setup for tests - As a user, I would like to have the test database automatically setup so that I can run stateless automated tests that do not rely on manual data.

CEWQ0034 - Restrict owners to only have one company - As a user, I would like to restrict owners to only have one company so that I can control how many companies I have in the system.

CEWQ0035 - Fix incorrect HTTP status code for company already existing - As a user, I would like to get the proper http response on an invalid request so that I can properly know what went wrong in the request.

CEWQ0036 - Fix incorrect HTTP status code for owner already having a company - As a user, I would like to get the proper http response on an invalid request so that I can properly know what went wrong in the request.

CEWQ0037 - Develop test suite for company post route - As a user, I would like to automatically test api route so that I can make changes to code without breaking expected functionality.

CEWQ0038 - Develop test suite for company get route - As a user, I would like to automatically test api route so that I can make changes to code without breaking expected functionality.

CEWQ0039 - Develop automated test for customer registration - As a user, I would like to automatically test api route so that I can make changes to code without breaking expected functionality.

CEWQ0040 - Develop automated test for owner registration - As a user, I would like to automatically test api route so that I can make changes to code without breaking expected functionality.

Sprint 4:

CEWQ0041 - Create Dockerfile - As a user, I would like to have a Dockerfile so that I can build highly-scalable, high-availability architecture that is cloud platform agnostic.

CEWQ0042 - Add .dockerignore - As a user, I would like to have a .dockerignore file so that I can prevent having sensitive information inside a container running on a cloud system.

CEWQ0043 - Create Dockerfile stages - As a user, I would like to have Dockerfile stages so that I can manage dev, test, and production environments.

CEWQ0044 - Create Dockerhub account - As a user, I would like to have a Docker hub account so that I can manage my Docker images remotely to host them on cloud providers.

CEWQ0045 - Add continuous integration - As a user, I would like to have continuous integration so that I can only make secure and expected changes to production code.

CEWQ0046 - Document Docker pull and run commands - As a user, I would like to have documentation of the docker commands so that I can have a reference on how to interact with our docker system.

CEWQ0047 - Document Docker build and push commands - As a user, I would like to have documentation of the docker commands so that I can have a reference on how to interact with our docker system.

CEWQ0048 - Document Docker remove all containers and their volumes, and delete all images commands - As a user, I would like to have documentation of the docker commands so that I can have a reference on how to interact with our docker system.

CEWQ0049 - Document issue with Docker bash login - As a user, I would like to login to docker via the bash cli so that I can interact with our docker system.

Sprint 5:

CEWQ0050 - Deploy Docker App to Azure - As a user, I would like to deploy my docker app to the cloud so that I can access my application through the internet.

CEWQ0051 - Create Linux resource - As a user, I would like to have a host os so that I can run my docker container on the cloud.

CEWQ0052 - Create subscription resource - As a user, I would like to pay for my cloud docker app so that I can keep running it with high availability.

CEWQ0054 - Add nom security audit to docker build - As a user, I would like to audit my dependencies on a docker build so that I can fail the build should a vulnerability be exploitable.

Sprint 6:

CEWQ0055 - Optimize Docker Build - As a user, I would like to have an optimal Docker build so that I can correctly use all of the features of Docker.

CEWQ0056 - Collect all Credentials - As a user, I would like to have all of the credentials of the system so that I can use the cloud based resources next semester.

CEWQ0057 - Record Instruction Videos - As a user, I would like to have instructions of the system so that I can know how to continue building on it.

CEWQ0058 - Document UML Diagrams - As a user, I would like to have UML documentation of the system so that I can have an industry standard overview of the project.

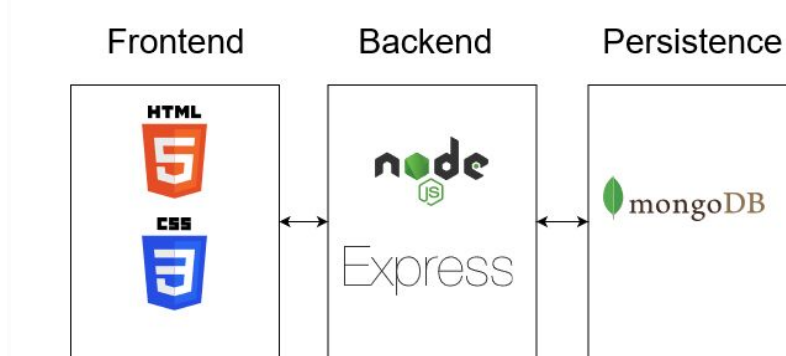
System Design

This section discusses the design decisions, and architecture patterns implemented in this project.

Architectural patterns

The architectural pattern is 3 tier.

Architectural Diagram



System and Subsystem Decomposition

Front-End:

Greenfield project that is open to any framework or technology

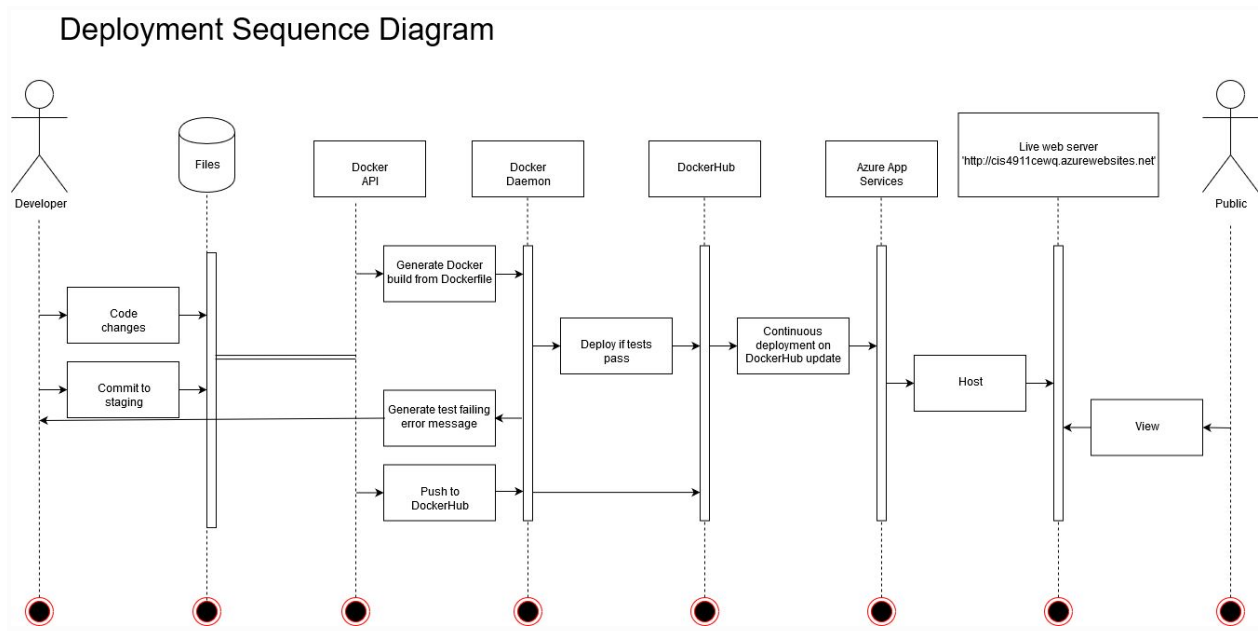
Back-End:

Middleware: NodeJs for runtime and Express for http server

Routes: Express-router that routes url to corresponding controller class

Database: Cloud MongoDB Atlas

Deployment Sequence Diagram

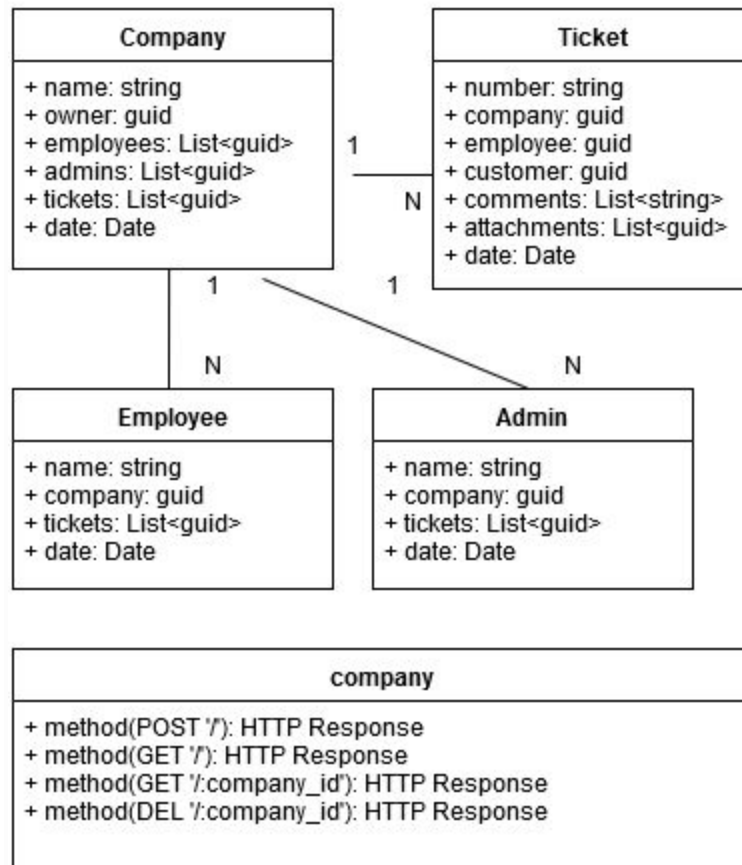


Design Pattern

We have:

- Company model
- Auth. middleware
- Company API
- Azure web Service
- Http req.

Class Diagram



System Validation

The following are documentation of the test cases.

Test Case	Setup Database
Purpose	Should empty test db
Expected Result	Empty MongoDB Atlas

Test Case	Register Owner
Purpose	Should register owner account
Expected Result	A signed user jwt

Test Case	Update user type to owner
Purpose	Change user type to owner
Expected Result	Mongo doc to not be null

Test Case	Validate create a company
Purpose	Validate http request body
Expected Result	Errors array in response

Test Case	Create a company
Purpose	Create a company corresponding to the owner user
Expected Result	Company json

Glossary

Cloud-based - refers to applications, services or resources made available to users on demand via the Internet from a cloud computing provider's servers.

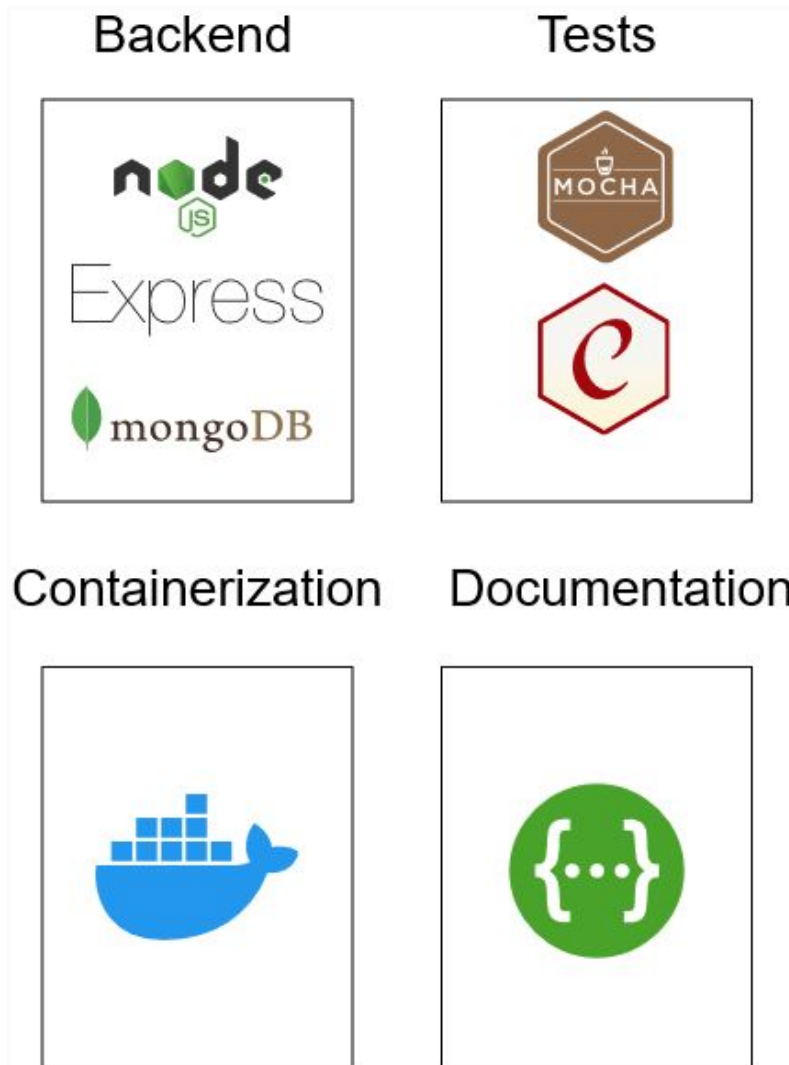
Docker - tool designed to make it easier to create, deploy, and run applications by using containers. Containers allow a developer to package up an application with all of the parts it needs, such as libraries and other dependencies, and ship it all out as one package.

Open Source - denotes to software for which the original source code is freely available and can be redistributed and modified.

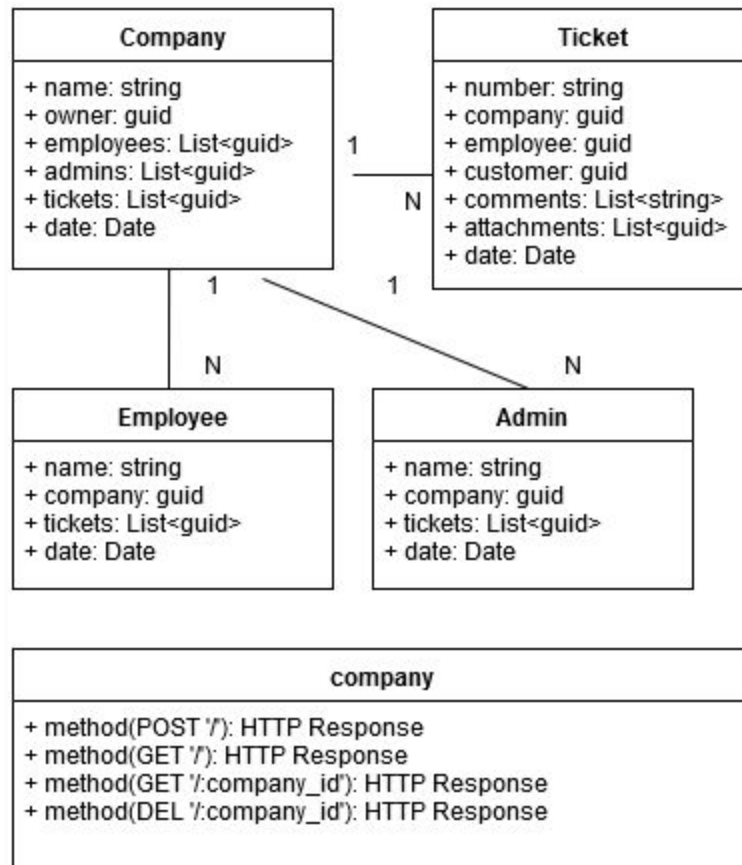
Swagger UI - use to visualize and interact with the API's resources without having any of the implementation logic in place.

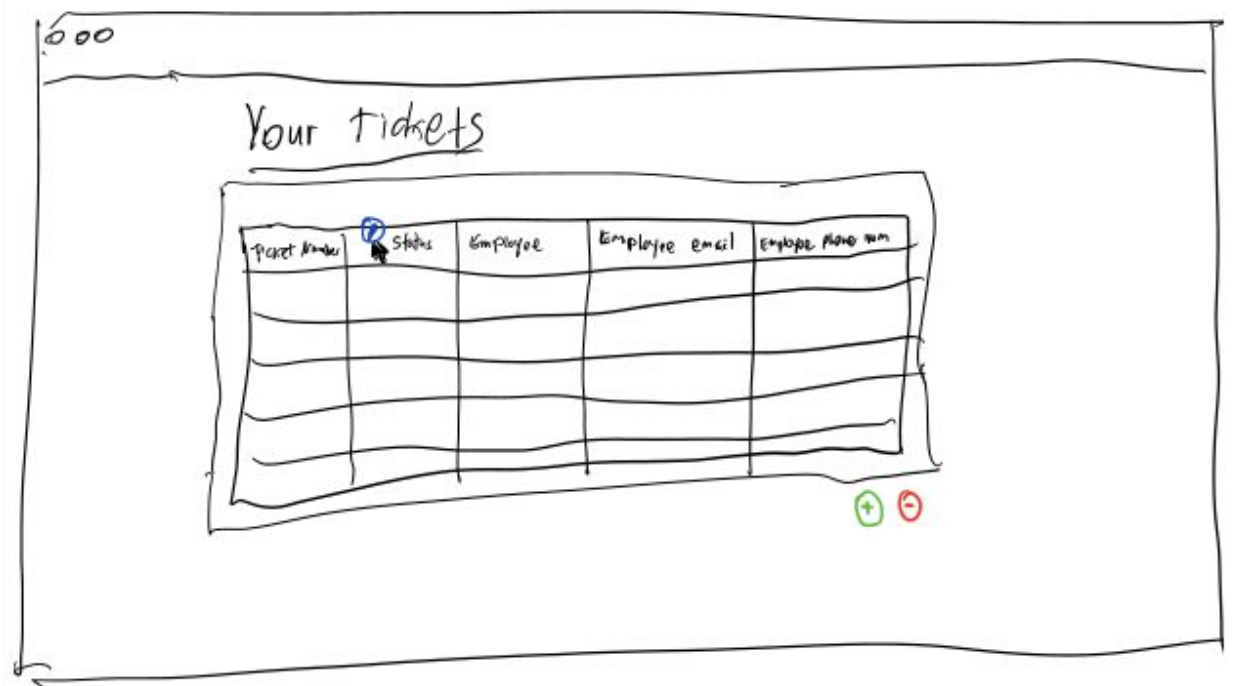
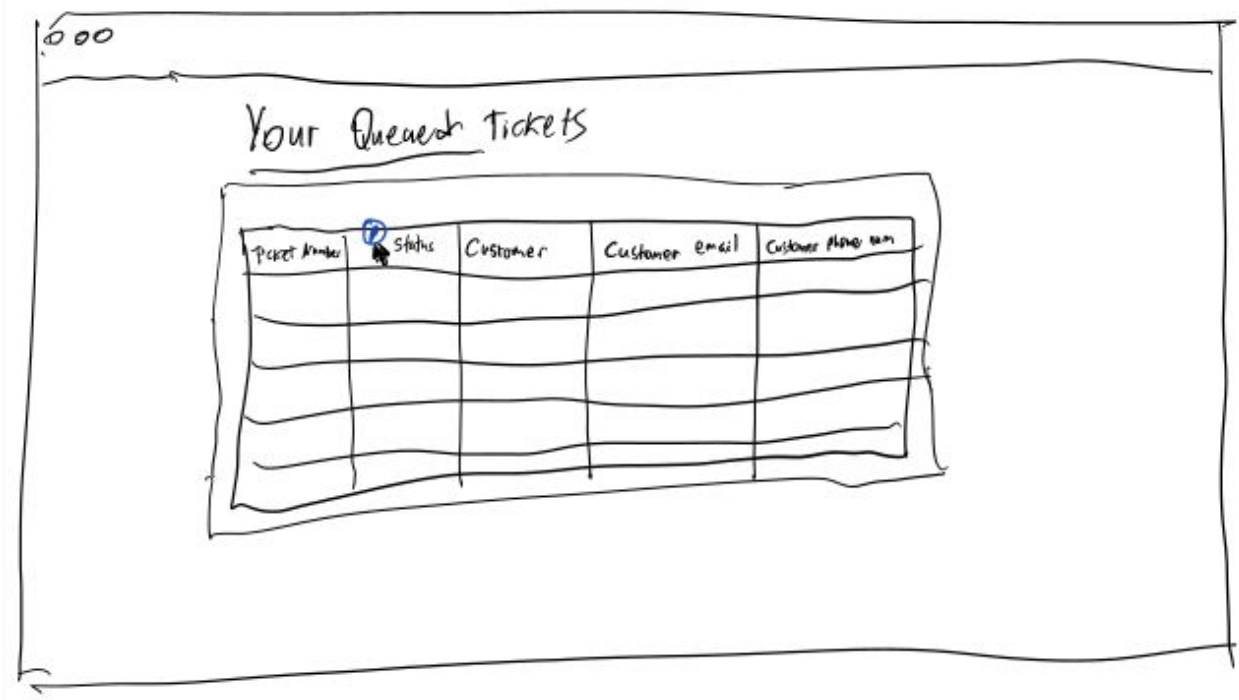
Appendix

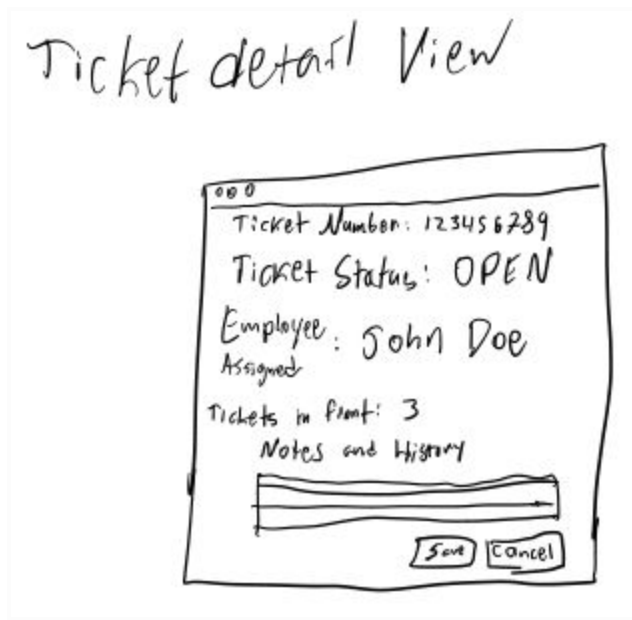
Appendix A - UML Diagrams



Class Diagram



Appendix B - User Interface Design**Employee interface mockup**



Appendix C - Sprint Review Reports

Sprint 1 Review:

Date: 09/26/2019

Start time: 5:40

End time: 6:40

Attendees:

Michael Reyerros

Osniel Valdivia

Francisco Espinosa

The following stories were accepted and discussed with the Product owner.

CEWQ0001 - Setup cloud database - As a user, I would like to have a cloud database so that I can persist data and access data from anywhere.

CEWQ0002 - Create admin data model - As an admin, I would like to have a data model representation so that I can access the system in the way intended for admins.

CEWQ0003 - Create ticket data model - As a user, I would like to have a data model representation of a ticket so that I can store and access relevant ticket information in the system.

CEWQ0004 - Create employee data model - As an employee, I would like to have a data model representation so that I can access the system in the way intended for employees.

CEWQ0005 - Create owner data model - As an owner, I would like to have a data model representation so that I can access the system in the way intended for owners.

CEWQ0006 - Create customer data model - As a customer, I would like to have a data model representation so that I can access the system in the way intended for customers.

CEWQ0007 - Create company data model - As a user, I would like to have a data model representation of a company so that I can access the system in the way intended for entities corresponding to that company.

CEWQ0008 - Create event data model - As a user, I would like to have a data model representation of an event so that I can have a history of events during a ticket's lifespan.

CEWQ0013 - Add authentication - As a user, I would like to have authentication so that I can access my data and no one else can access data they are forbidden from.

CEWQ0014 - Design landing page mockup - As a designer, I would like to have an initial landing page mockup so that I can do a refined mockup with software.

Sprint 2:

Date:10/07/2019

Start time:8:30pm

End time:9:00pm

Attendees:

Osniel Valdivia

Francisco Espinosa

The following stories were accepted and discussed with the Product owner. A live demo was also provided.

CEWQ0009 - Create company api - As a user, I would like to have RESTful API routes so that I can get data from the backend using HTTP requests from a front-end client.

CEWQ0015 - Document user api - As a user, I would like to have Swagger documentation of api routes, parameters, request bodies, responses, and authentication so that I can have a reference on how to interact with the backend.

CEWQ0016 - Document company api - As a user, I would like to have Swagger documentation of api routes, parameters, request bodies, responses, and authentication so that I can have a reference on how to interact with the backend.

CEWQ0017 - Create Swagger yaml - As a user, I would like to have a Swagger documentation yaml so that I can have an OpenAPI standard reference on how to communicate with the backend system.

CEWQ0018 - Document company data model - As a user, I would like to have a Swagger documentation of this data model so that I can represent this model as json through the API.

CEWQ0019 - Document user data model - As a user, I would like to have a Swagger documentation of this data model so that I can represent this model as json through the API.

CEWQ0020 - Document ticket data model - As a user, I would like to have a Swagger documentation of this data model so that I can represent this model as json through the API.

CEWQ0021 - Document event data model - As a user, I would like to have a Swagger documentation of this data model so that I can represent this model as json through the API.

CEWQ0022 - Postman collection for company - As a user, I would like to have a Postman collection of this abstraction so that I can send json data to the API.

CEWQ0028 - NodeJs for API - As a user, I want to have a fast web-scalable application so I want to use NodeJS for the API back end.

CEWQ0030 - Design Ticket API - As a user, I would like to have a RESTful API routes for tickets so that I can get ticket data from the back-end from a front-end client.

Sprint 3:

Date: 10/21/2019

Start time: 8:30pm

End time:9:00pm

Attendees:

Osniel Valdivia

Francisco Espinosa

Michael Reyerros

The following stories were accepted and discussed with the Product owner. A live demo was also provided.

CEWQ0031 - Serve Swagger UI on API - As a user, I would like to have a Swagger OpenAPI UI of the API routes so that I can have access to a standard documentation that shows how to interact with the api.

CEWQ0032 - Add npm clean script for packages - As a user, I would like to have a package management script so that I can try out packages and effectively remove them and solve any conflicts.

CEWQ0033 - Add automated database setup for tests - As a user, I would like to have the test database automatically setup so that I can run stateless automated tests that do not rely on manual data.

CEWQ0034 - Restrict owners to only have one company - As a user, I would like to restrict owners to only have one company so that I can control how many companies I have in the system.

CEWQ0035 - Fix incorrect HTTP status code for company already existing - As a user, I would like to get the proper http response on an invalid request so that I can properly know what went wrong in the request.

CEWQ0036 - Fix incorrect HTTP status code for owner already having a company - As a user, I would like to get the proper http response on an invalid request so that I can properly know what went wrong in the request.

CEWQ0037 - Develop test suite for company post route - As a user, I would like to automatically test api route so that I can make changes to code without breaking expected functionality.

CEWQ0038 - Develop test suite for company get route - As a user, I would like to automatically test api route so that I can make changes to code without breaking expected functionality.

CEWQ0039 - Develop automated test for customer registration - As a user, I would like to automatically test api route so that I can make changes to code without breaking expected functionality.

CEWQ0040 - Develop automated test for owner registration - As a user, I would like to automatically test api route so that I can make changes to code without breaking expected functionality.

Sprint 4:

Date: 11/4/2019

Start time: 8:30pm

End time: 9:00pm

Attendees:

Osniel Valdivia

Francisco Espinosa

Michael Reyerros

The following stories were accepted and discussed with the Product owner. A live demo was also provided.

CEWQ0041 - Create Dockerfile - As a user, I would like to have a Dockerfile so that I can build highly-scalable, high-availability architecture that is cloud platform agnostic.

CEWQ0042 - Add .dockerignore - As a user, I would like to have a .dockerignore file so that I can prevent having sensitive information inside a container running on a cloud system.

CEWQ0043 - Create Dockerfile stages - As a user, I would like to have Dockerfile stages so that I can manage dev, test, and production environments.

CEWQ0044 - Create Dockerhub account - As a user, I would like to have a Docker hub account so that I can manage my Docker images remotely to host them on cloud providers.

CEWQ0045 - Add continuous integration - As a user, I would like to have continuous integration so that I can only make secure and expected changes to production code.

CEWQ0046 - Document Docker pull and run commands - As a user, I would like to have documentation of the docker commands so that I can have a reference on how to interact with our docker system.

CEWQ0047 - Document Docker build and push commands - As a user, I would like to have documentation of the docker commands so that I can have a reference on how to interact with our docker system.

CEWQ0048 - Document Docker remove all containers and their volumes, and delete all images commands - As a user, I would like to have documentation of the docker commands so that I can have a reference on how to interact with our docker system.

CEWQ0049 - Document issue with Docker bash login - As a user, I would like to login to docker via the bash cli so that I can interact with our docker system.

Sprint 5:

Date: 11/18/2019

Start time: 8:30pm

End time: 9:00pm

Attendees:

Osniel Valdivia

Francisco Espinosa

Michael Reyerors

The following stories were accepted and discussed with the Product owner. A live demo was also provided.

CEWQ0050 - Deploy Docker App to Azure - As a user, I would like to deploy my docker app to the cloud so that I can access my application through the internet.

CEWQ0051 - Create Linux resource - As a user, I would like to have a host os so that I can run my docker container on the cloud.

CEWQ0052 - Create subscription resource - As a user, I would like to pay for my cloud docker app so that I can keep running it with high availability.

CEWQ0054 - Add nom security audit to docker build - As a user, I would like to audit my dependencies on a docker build so that I can fail the build should a vulnerability be exploitable.

Sprint 6:

Date: 12/02/2019

Start time: 8:30pm

End time: 9:00pm

Attendees:

Osniel Valdivia

Francisco Espinosa

Michael Reyeros

CEWQ0055 - Optimize Docker Build - As a user, I would like to have an optimal Docker build so that I can correctly use all of the features of Docker.

CEWQ0056 - Collect all Credentials - As a user, I would like to have all of the credentials of the system so that I can use the cloud based resources next semester.

CEWQ0057 - Record Instruction Videos - As a user, I would like to have instructions of the system so that I can know how to continue building on it.

CEWQ0058 - Document UML Diagrams - As a user, I would like to have UML documentation of the system so that I can have an industry standard overview of the project.

Appendix D - User Manuals, Installation/Maintenance Document/Shortcomings/Wishlist Document and Other Documents

1- Introduction Video

- Link: [FIU SCIS-2019 Fall-CEWQ-IntroVideo](#)

2- User Guide Video

- Link: [FIU SCIS-2019 Fall-CEWQ-UserGuide](#)

3- Installation and Maintenance Guide Video

- Link: [FIU SCIS-2019 Fall-CEWQ-InstallMaintenanceGuide](#)

4- Shortcomings and Wish List Video

- Link: [FIU SCIS-2019 Fall-CEWQ-ShortcomingsWishlist](#)

5- Sprints

- [FIU SCIS-2019 Fall-CEWQ-Sprint 1](#)
- [FIU SCIS-2019 Fall-CEWQ-Sprint 2](#)
- [FIU SCIS-2019 Fall-CEWQ-Sprint 3](#)
- [FIU SCIS-2019 Fall-CEWQ-Sprint 4](#)
- [FIU SCIS-2019 Fall-CEWQ-Sprint 5](#)

References

- <https://docs.docker.com/get-started/>
- <https://apihandyman.io/writing-openapi-swagger-specification-tutorial-part-1-introduction/>