A School Operation And Management System

Keywords: SaaS, application, management, operation, school, etc.

Last revision: on May 7th, 2019

Counters: 2889 words or 15909 characters (not including spaces and sections before introduction included) in 31 pages

18016754 - Ranavann CHHEM
18016826 - Quentin REVEL
18027317 - Anudeep CHIKKAM
18027425 - Jeiman JEYASINGAM
18027434 - Parijat PATEL
18079563 - Antoine YREUX

School of Engineering, Computing and Mathematics (ECM)

Oxford Brookes University

A coursework report (Assessment 2) submitted for the module

P00412 - Big Data and the Cloud

on May 7th, 2019

as part of the degree

CM80A - Master of Science in Computer Science

September 2019

Declaration of authorship

We, Ranavann CHHEM, Anudeep CHIKKAM, Jeiman JEYASINGAM, Parijat PATEL, Quentin REVEL and Antoine YREUX, certify that this coursework report titled, “A School Operation And Management System” and the work presented in it are our own.

We swear that:

- This coursework report was done as part of our postgraduate studies in Master of Science of Computer Science or Computing at Oxford Brookes University.
- If any component of this coursework report had already been transmitted for any kind of qualification, certification or degree at any institution including this university, this has been plainly declared.
- When we have considered / examined and used every kind of source or that the coursework report is based on collaborative work, this is always precised and properly cited and referenced otherwise, this coursework report is completely our own work.

Signed:

Date: May 7th, 2019

Acknowledgements

We would like to thanks Doctor Hong ZHU for his lectures, practical lessons and guidance.

The application will:

- be developed with:
  - Bootstrap-Vue made by Twitter (2016), a CSS Framework for web design.
  - Express.js, a JS Framework for data management and security purposes.
  - Vue.js made by YOU (2014), a JavaScript (JS) Framework for optimale performance.
- use MongoDB as a database due to Big Data.
- be versioned on Gitlab made by Gitlab Inc. (2011).
- be deployed:
  - on Docker (2013) Containers on nodes pools on Kubernetes clusters for scalability (high-availability, load balancing) and security purposes on Google Cloud Platform made by Google (2011).
  - through Gitlab for one cluster Codefresh made by Codefresh (2017) for multiple clusters (Auto DevOPS (Continuous Integration / Continuous Deployment) platform).
  - with TLS certificate and encryption from Electronic Frontier Foundation, Mozilla Foundation, University of Michigan, Akamai Technologies, and Cisco Systems (2014).
- store documents (contracts, cover letters, CV, etc.) on Google Cloud Filestore made by Google (2011).

Acronyms

GDPR . *Glossary:* General Data Protection Officer (GDPR), 4

Glossary

General Data Protection Officer (GDPR) is the European regulation which frames everything regarding private data. 4

List of Figures

Contents

Introduction - Group

With the advent of mobile and cloud computing technologies, it is perceived that a new generation of school management systems can be developed to automate school operations by taking advantage of frontier IT technologies such as big data, cloud computing and mobile computing. It is decided to develop a multi-tenant SaaS application with Google Cloud platform to illustrate the functions of school management and operation.

This document provides the architectural design and specifications for the user to understand how the system operates, and for the engineers to amend the changes as per the requirements. Section 1 states designed components, common features developed for each user and individual features developed per each user type. It also analyses the architectural design, security features, software used and detailed analysis of each specification. Section 2 defines the setup of run-time environment, tools and technologies used in building this project, configuration and implementation details.

1 Design specification

1.1 Scope

The software design document will provide readers an insight in meeting client's needs effectively and efficiently.

Subsequently the document facilitates understanding and communication of the system by providing several views of the system design.

This is achieved through the use of comprehensive description, architectural patterns, sequence diagrams, class diagrams, user interfaces and relational models.

1.2 Purpose

The purpose of this document as a critical part of the effective design was to guide the development phase and to satisfy the brief of functional and non-functional requirements of the software.

It also allows the system to be offered to the existing users.

It defines the scope of data models, structures, architecture, interfaces and component level design.

This document can also serve a reverse engineering asset for users and stakeholders that will assist the system in the future.

1.3 Project overview

The system is a web-based application designed for management of schools and their operations.

It is a six-sided platform for school managers, students, teachers, parents, event organizers, and system operators.

Agile methodology development was practised to produce app revisions in sprints as set in the project plan.

For design phase, critical components of the software system was specified, indicating its functionality and usability.

Furthermore, data migration and interfaces were defined and considered into available resources.

1.4 Related software

- Google Drive was used to share resources and each student's contribution.
- Asana was used for basing the system on individual tasks framed into bigger projects.
- GitLab was used for code versioning and test analysis.
- Bootstrap-Studio was used to create a responsive web design.

1.5 Related Documents

- Project Plan
- Requirements Specification
- Test and Configuration Plan
- Test and Evaluation Results
- Configuration Management Plan and Implementation Details
- Quality Factors and Risk Analysis

1.6 Database

- MongoDB

1.7 Icons

- FontAwesome

1.8 Programming languages

- Front-End: HTML5, SCSS (converted to CSS3), TypeScript (converted to JS)
- Back-end: NoSQL, TypeScript (converted to JS)

1.9 Frameworks

- Bootstrap v4.3.1 / Bootstrap-Vue.js v2.0.0-rc19
- Express.js v4.16.4
- Vue.js v2.6.10

1.10 High Level components and interfaces

1.10.1 Graphical User Interfaces

- Overview

  The development of the interfaces were performed after gathering requirements from the specification.

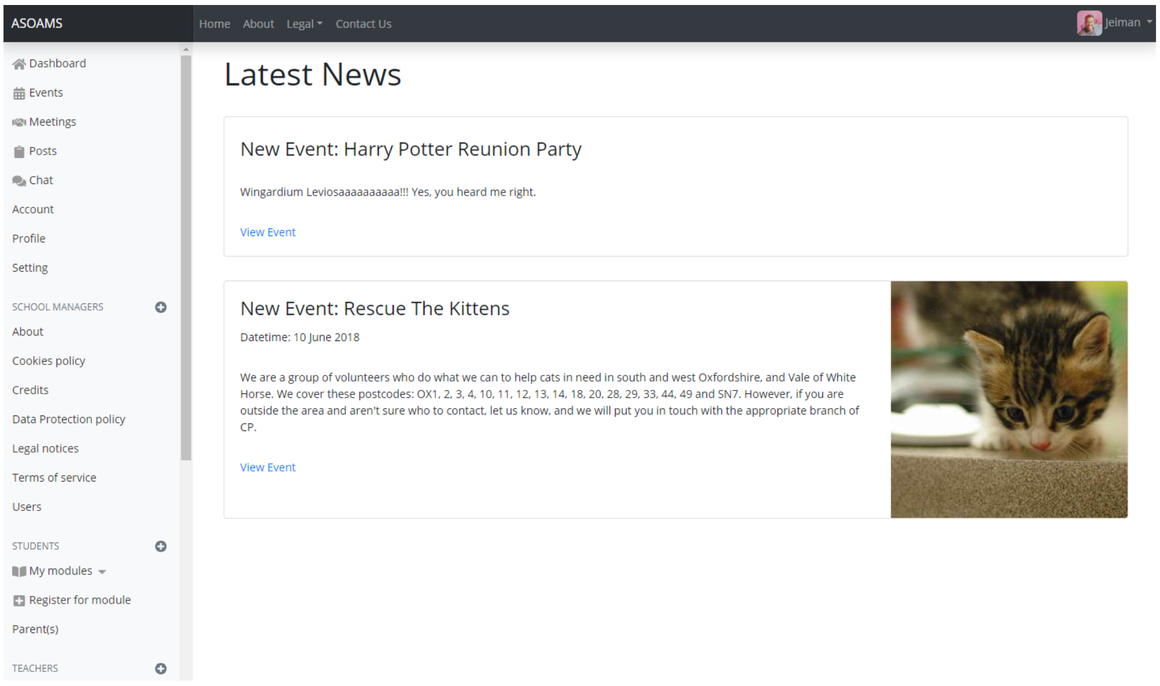
  In terms of usability, various user interface frameworks were considered to facilitate the design process. Overall, Bootstrap was chosen to create the underlying user interface layer to respond to user behaviour and interactions, including responsiveness on various platform screen sizes.

  Considering various types of users: Students, Teachers, Parents, System Operators, School Managers and Event Organizers, as well as various types of devices on which the system will be used, such as smart phones, tablets and computers, the system has an intuitive GUI interface for seamless navigation and input forms (for CRUD operations).

  The conventional interfaces that follow the same graphical template, allow the developers to perform their tasks quickly and efficiently with minimal learning curve while providing basic experience in building a RESTFul micro-service application that interacts with a web-based application.

  This fulfills the non-functional requirement usability.

- Dashboard layout



Latest News

New Event: Harry Potter Reunion Party

Wingardium Leviosaaaaaaaaaa!!! Yes, you heard me right.

View Event

New Event: Rescue The Kittens

Datetime: 10 June 2018

We are a group of volunteers who do what we can to help cats in need in south and west Oxfordshire, and Vale of White Horse. We cover these postcodes: OX1, 2, 3, 4, 10, 11, 12, 13, 14, 18, 20, 28, 29, 33, 44, 49 and SN7. However, if you are outside the area and aren't sure who to contact, let us know, and we will put you in touch with the appropriate branch of CP.

View Event

Figure 1: Front-end layout

This layout is used on every subsequent pages where the user needs to interact with the system. It contains a standard sidebar on the left side of the screen, navigation bar at the top and the page contents in the middle.

The dashboard sidebar menu will contain only the appropriate common and dedicated navigation items and links of the appropriate target users according to their permissions through their roles and are described as follows:

– Dashboard (every spaces)

  Every user will be able to:

  ∗ Manage their accounts and profiles.
  ∗ Exchange with others through the chat / messaging system.
  ∗ Make claims, feedback and support requests on ASOAMS platform.
  ∗ View some security audits about their connections and operations on ASOAMS platform.
  ∗ View some statistics about their use of ASOAMS platform.

– School Managers Space

  School Managers will be able to:

    * Manage ASOAMS platform, and mainly its documentations (users guides) / frequently asked questions and legal information.
    * Manage claims, feed-backs and supports requests on ASOAMS platform.
    * View some statistics of the use of ASOAMS platform.

  - Students Space

    Students will be able to:

    * Schedule 1:1 meetings with teachers, event organizers and school managers. Invitations will be sent out to attendees to be accepted or declined accordingly.
    * Update and delete meetings accordingly with new amendments.

  - Teachers Space

    Teachers will be able to:

    * Create a module for their subject.
    * Edit the module with amendments to marking criteria, modify the syllabus, etc.
    * Delete the module for their subject as per the requirement.

  - Parents Space

    Parents will be able to:

    * Pay the school fees for their children, including paying for trips, events and any other related activity.
    * View the history of all the payments made from beginning, if any payment is unsuccessful, etc.

  - Event Organizers Space

    Event Organizers will be able to:

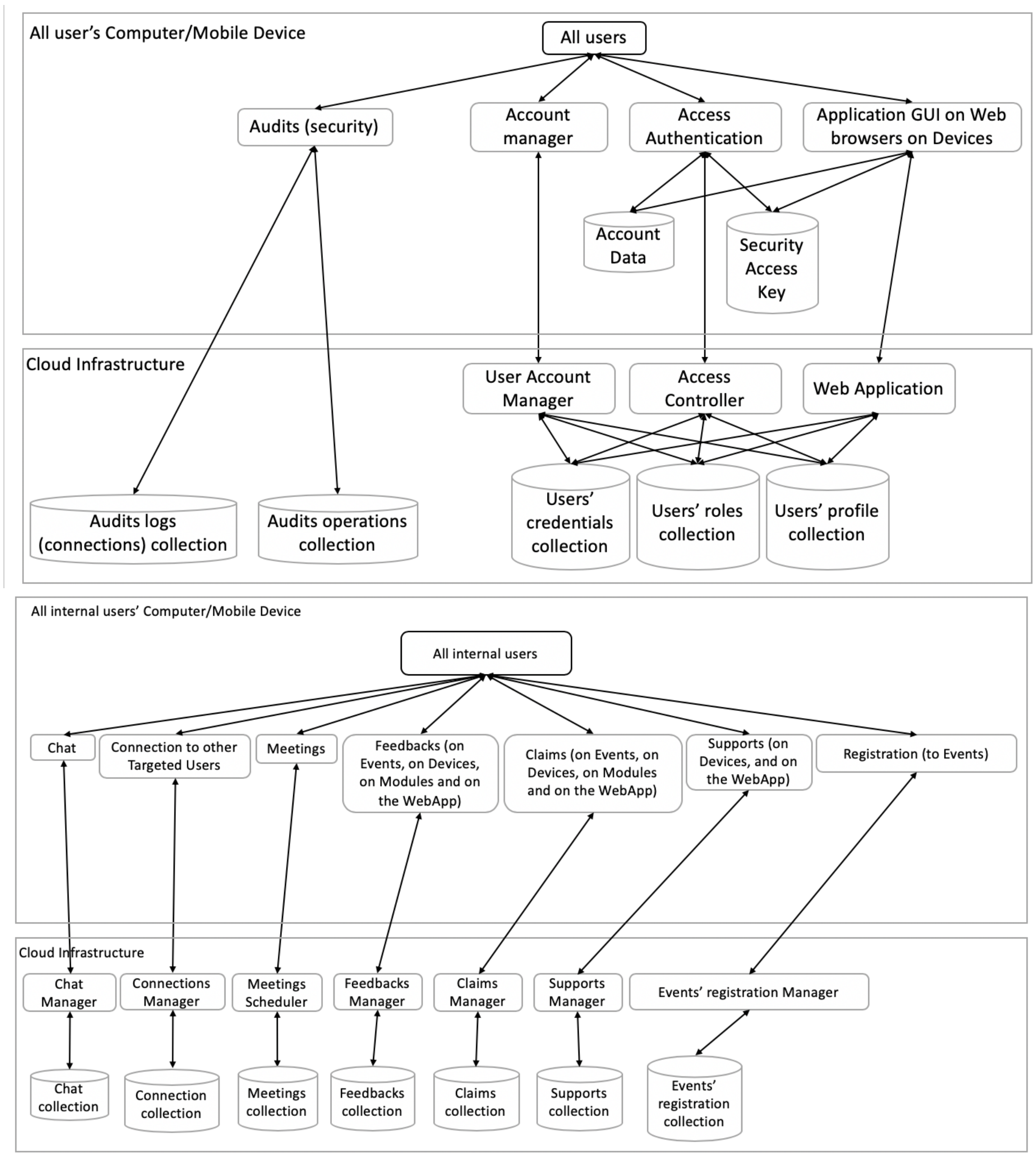
    * To create the event, provide details of the event.
    * Edit the events as per the requirements.

  - System Operators Space

    System Operators will be able to:

    * Manage claims, feedback and supports requests on ASOAMS platform.
    * Manage every users accounts if need.
    * Create users, roles and profiles for every active user in the school.
    * Update and delete users, roles and profiles.

1.11 Architectural Design



All user's Computer/Mobile Device
All users
Audits (security)
Account manager
Access Authentication
Application GUI on Web browsers on Devices
Account Data
Security Access Key
Cloud Infrastructure
User Account Manager
Access Controller
Web Application
Users' credentials collection
Users' roles collection
Users' profile collection
Audits logs (connections) collection
Audits operations collection
All internal users' Computer/Mobile Device
All internal users
Chat
Connection to other Targeted Users
Meetings
Feedbacks (on Events, on Devices, on Modules and on the WebApp)
Claims (on Events, on Devices, on Modules and on the WebApp)
Supports (on Devices, and on the WebApp)
Registration (to Events)
Cloud Infrastructure
Chat Manager
Connections Manager
Meetings Scheduler
Feedbacks Manager
Claims Manager
Supports Manager
Events' registration Manager
Chat collection
Connection collection
Meetings collection
Feedbacks collection
Claims collection
Supports collection
Events' registration collection

Figure 2: Architectural Design of all users

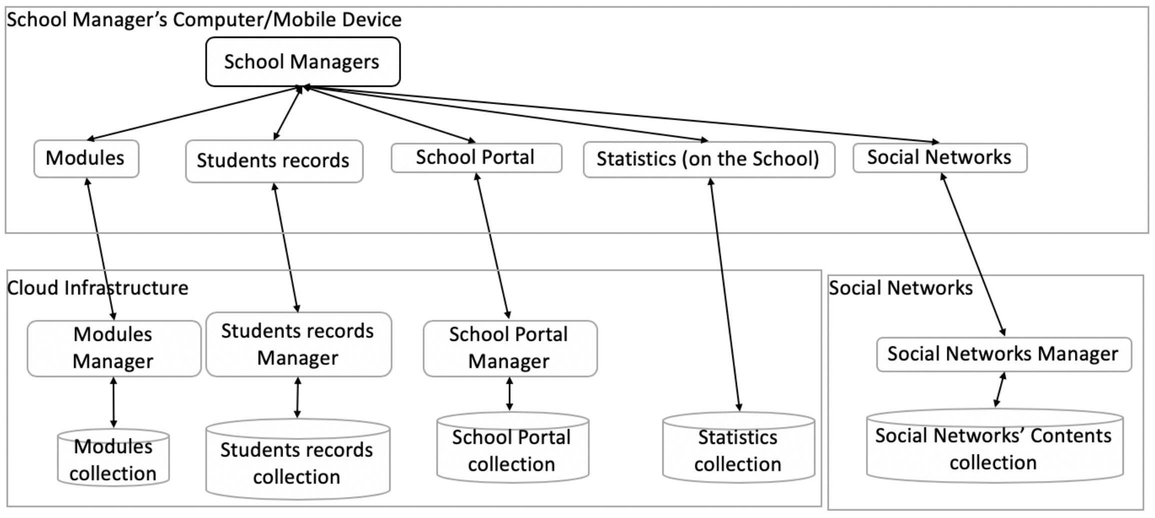


Figure 3: Architectural Design of School Managers

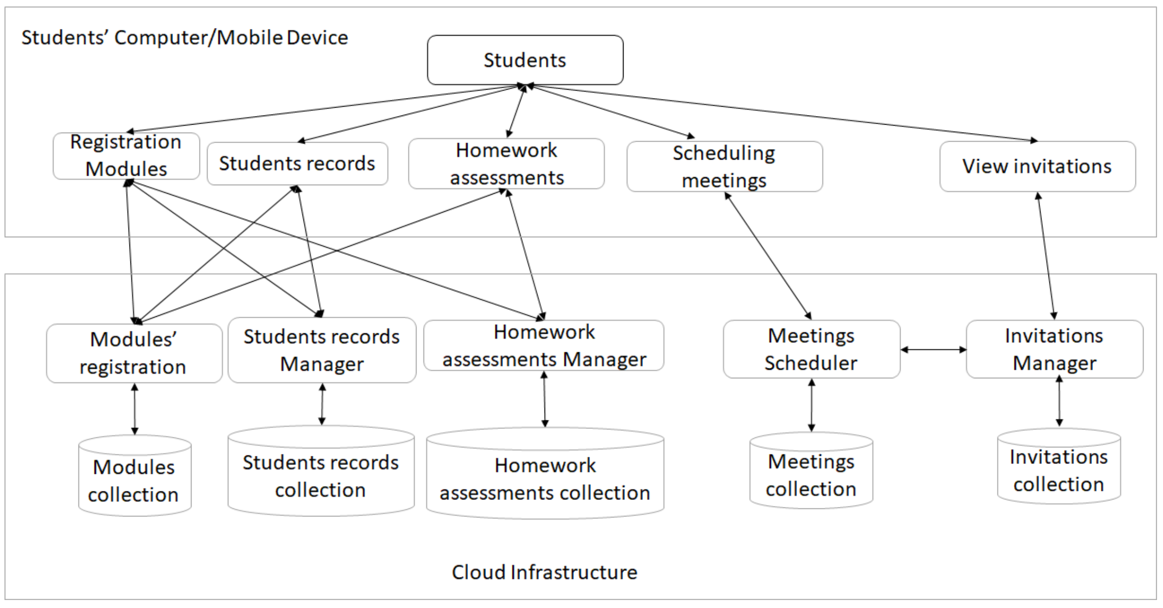


Figure 4: Architectural Design of Students

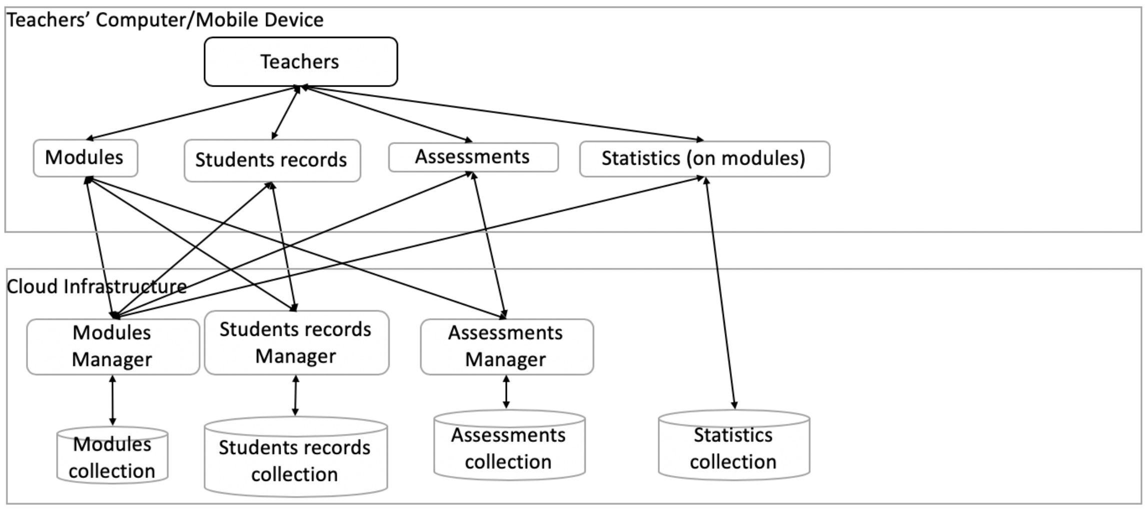


Figure 5: Architectural Design of Teachers

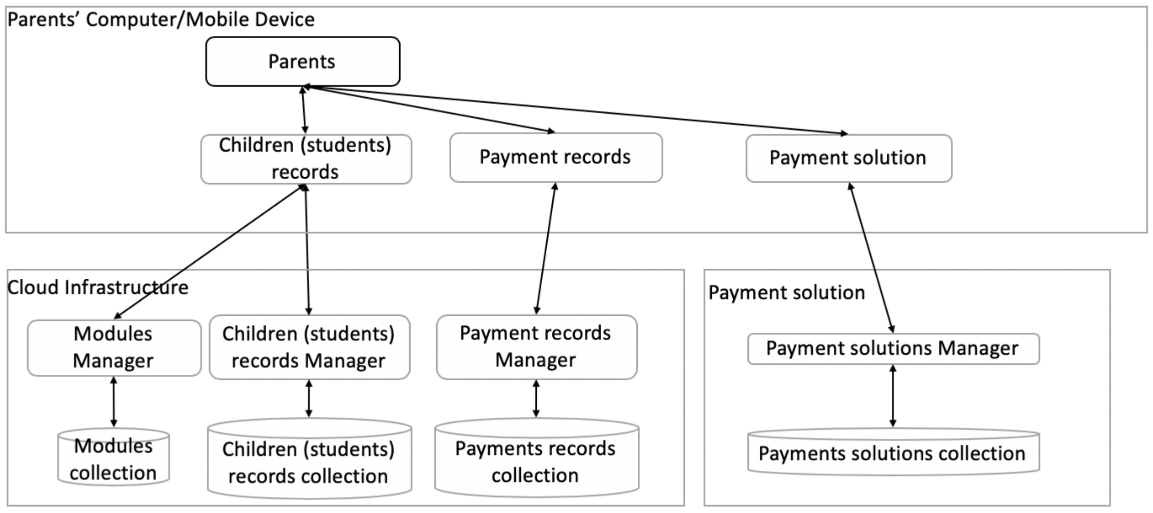

Parents' Computer/Mobile Device
Parents
Children (students) records
Payment records
Payment solution
Cloud Infrastructure
Modules Manager
Children (students) records Manager
Payment records Manager
Modules collection
Children (students) records collection
Payments records collection
Payment solution
Payment solutions Manager
Payments solutions collection

Figure 6: Architectural Design of Parents

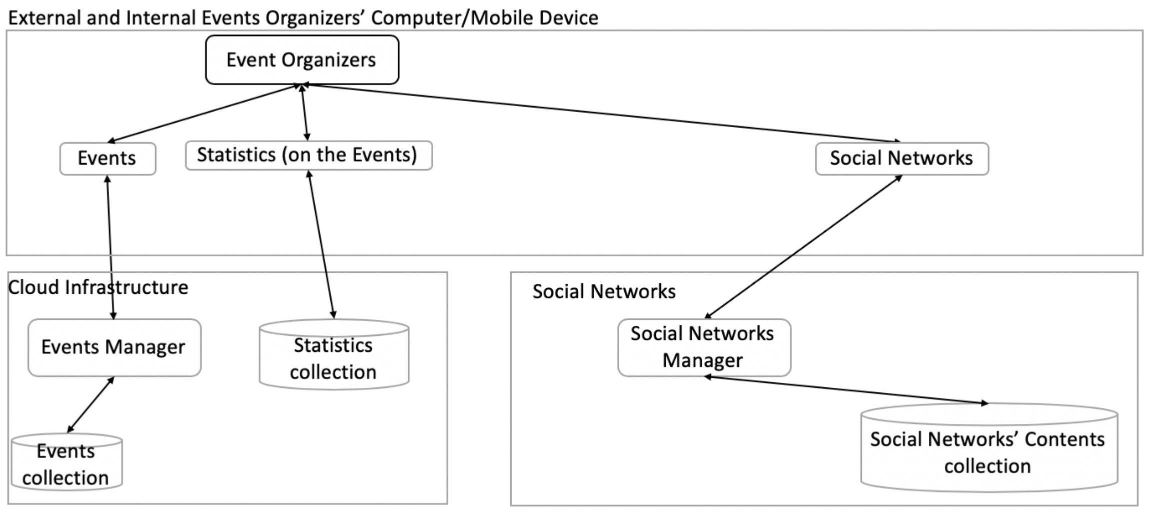


Figure 7: Architectural Design of Event Organizers

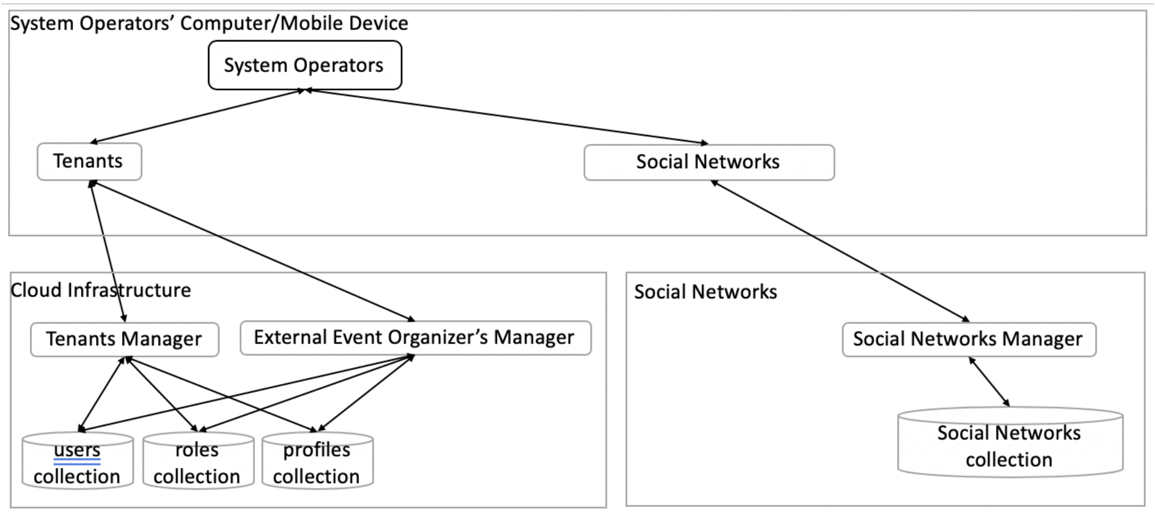


Figure 8: Architectural Design of System Operators

1.12 Style/Methodology

ASOAMS has been developed in a MVC (Model-View-Controller) architectural style with the addition of Three-Tier Client/Server Architecture.

MVC style separates business logic code and presentation code from the system data. This will increase system maintainability and scalability.

Model (from MVC) allowed us to develop in parallel, introducing code re-usability.

The online application is structured into three logical components that interacts with one another.

Model component - Manages the system data and operations associated on the data.

View component - Manages and defines how the data is presented to the user.

Controller component - Manages user interaction and passes interactions to View and Model.

The design of ASOAMS is based on Object Oriented Programming Methodology.

The main reason for choosing this methodology is to improve software productivity and development process, provide lower development cost, overall providing a higher quality software.

1.13 Layers models

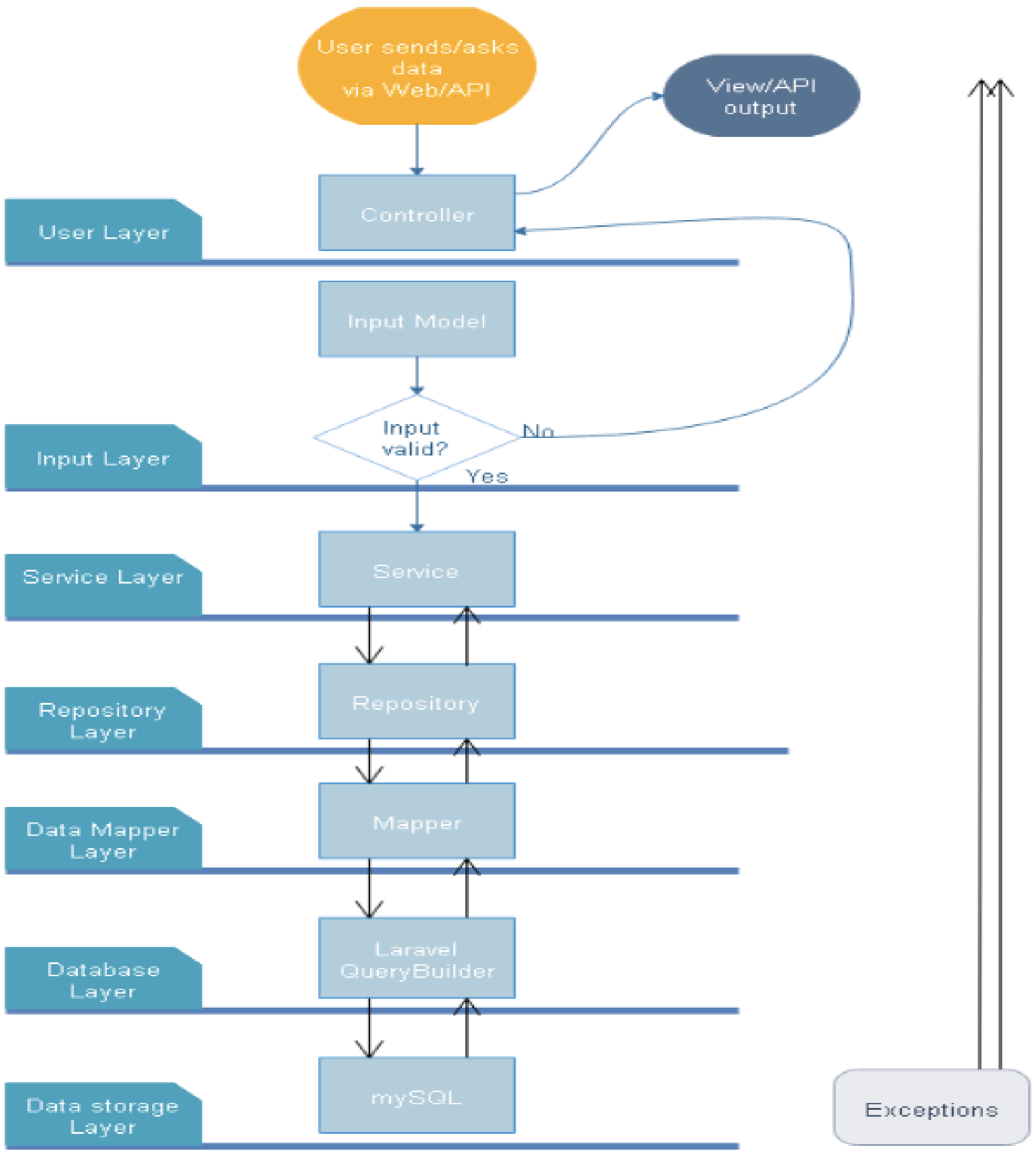


Figure 9: Layers models

1.14 Security Features and Layers

In order to protect the web application, several security layers were used, providing a high-quality web application security.

As the web application will gather a large number of sensitive, confidential information, for instance, in the event of an attack on the database, it would result in the database exposing sensitive information to attackers.

1.14.1 Configuration

With this in mind, authentication facilities were implemented to negate attacks. Facilities such as requesting an Authorization bearer token that is unique to each authenticated user every time a request is being made to any endpoint in the API. This results in denying unauthorised access to attackers.

In regards to registration and login authentication, a simple, easy to use system was implemented.

1.14.2 Storing Passwords

Passwords were hashed using a node package known as “Bcrypt”, that provided the extended capability of setting various salt levels when the password is being hashed.

Moreover, the node package provides flexibility in hashing passwords in the event of system expansion and scalability.

To provide an example, the default values have been increased to provide better security (bcrypt rounds from 10 to 31, argon memory from 1024 to 4096, argon threads from 2 to 8, and argon time from 2 to 8).

1.14.3 Authenticating Users

The method: Auth::attempt method was used in this case.

Even though email is used as the main entity to allow a user to login, the system can be customised by overriding the method responsible to utilise other forms of authentication, such as username.

Furthermore, it will redirect the user to the URL that they tried to access before trying to pass the authentication filter.

For this, a fallback URL was provided for the destination. When the authentication is successful, the user will be logged in.

A "remember me" option is provided to store the user's login credentials for a said period of 30 days or until he/she manually logs out from the system.

1.14.4 Protecting Page contents and Routes

Users are authenticated on each route based on their login credentials either through checking the Cookie state or checking the mutation from the state management (VueX). This results in the application being protected from cross-site request forgeries.

Furthermore, page contents and routes can be protected with a middleware library that checks the user's permissions through their roles and any other custom middleware such as, whitelisting or blacklisting IP addresses, user age, etc.

1.14.5 Password Reminders in Account Settings

The Account Settings section consists of providing full control to the user in managing their account. Password reminders are sent to every user to indicate that it needs to be changed regularly. This is accompanied by resetting options. The reminders can be sent via email, where users can follow instructions to reset or remember their passwords.

1.14.6 Password Validation

The validation is done by verifying that the password matches the characters. Furthermore, the validation will be done automatically.

1.14.7 Encryption

The framework used (Express.js) provides an encryption using mcrypt extension, that uses a technique that is encompassed in the PHP programming language.

1.14.8 Firewall (soft)

The firewall could manage the user's access to the application according to the whitelist/blacklist IP addresses and any other network parameters.

1.14.9 Form check

Express.js framework provides requests (app/Http/Requests) and rules (app/Rules) to check and validate the different fields of every form request.

This includes the Index Keys (Primary, Unique, and Foreign Keys) which check the relationships between the tables and can be update or delete accordingly in cascade.

1.14.10 Audit

Customs features could also be developed in MVC using the CheckRole middleware with only read/view permission to show users' connections and operations and notify those operations on their dashboard, by email, by push, and by sms.

1.14.11 TLS Encryption

Express.js framework has been configured to use HTTPS (TLS v1.2 and v1.3) instead of the usual HTTP protocol for security reason and mainly while loading the assets or for every routes.

1.14.12 Secured cookies

Express.js provides the possibility to secure the session cookie so it can not be read by anyone who can access it as it is encrypted and has been activated by default.

1.14.13 Kubernetes and Docker

Kubernetes and Docker permit to deploy the application and it's database into separate containers on the servers which can not be accessed the same way as the servers and increase its security.

1.14.14 Environments

Different environments (local, development, testing, staging, and production) are provided which permit to store the common configurations parameters and credentials.

For enhanced security, different credentials have been assigned for each environment and should be initialised during the deployment.

1.14.15 Two-Factor authentication

Two-Factor authentication like Google authentication can be use as a second authentication method to confirm the user authentication.

1.14.16 GDPR and Privacy

By implementing customs statistics features, it could tracked and reported to the users different operations frequency like their connections frequency and their locations which could help to prevent fraud, usurpation, etc.

2 Setup of Run-time Environment

2.1 Introduction

The configuration management describes the tools, frameworks and workflow the team members has used to design and develop the system with along with committing the changes to the project.

Section 2.4 illustrates the implementation steps taken to deploy the project to Google Cloud Platform. Moreover, it serves as a user guide to deploy the system on any cloud provider platform.

2.2 Tools and Technologies

ASOAMS project used the following technologies:

- Development
  - Bootstrap-Vue: Combination of Bootstrap 4 components and grid system integrated into Vue.js framework
  - Express.js: Back-end JavaScript framework
  - Vue.js: Front-end JavaScript framework
  - Miscellaneous: FontAwesome for the icons, SCSS (instead of SASS / CSS), TypeScript (instead of JavaScript)
- Cloud Operations
  - Google Cloud Platform: Low cost Infrastructure as an Enterprise cloud service provider
  - Docker: Containerization technology that provides consistency in virtualisation
  - GitLab: Easy DevOps implementation on top of Git default capabilities of version control management
  - Kubernetes: Container Orchestration Platform that simplify deployments of new code into production
- Database
  - MongoDB: A NoSQL database that is suitable and essential for Big Data

  - Mongoose: An object modeling tool designed to work in an asynchronous environment with MongoDB, ideal for the back-end API

- Environment
  - Asana: SaaS designed to help teams organize, track, and manage their work
  - Bootstrap-Studio: Create responsive designs using the Bootstrap framework
  - Visual Studio Code: Cross-platform IDE that supports every frameworks and languages

2.3 Source Code Architecture

2.3.1 For both micro-services:

- all libraries (node_modules) and compilation/installation instructions (package.json, tsconfig.json, etc.) ;
- the different documentations of the application (CHANGELOG.md, CONTRIBUTING.md, LICENSE.md, README.md) ;
- the environment file (.env) (note: the other environment files are here for easiest development purpose, the final application should only have one .env files and the credentials should be defined and inserted during the deployment) to define specific common credentials and librairies ;

2.3.2 For client micro-service:

- the src/App.vue file is the entrypoint of the application (for the GUI)
- the src/main.ts file is the entrypoint of the application
- the src/router.ts file import all features and their routes
- the src/assets folder contains all files like pictures that need to be compiled
- the src/components folder contains partials code that are part of a view and could be reused in several views
- the src/layouts folder contains template of webdesign for the Frontend and Backend

- the src/views folder contains all vue files for disconnected users also known as Frontend
- the src/views/Dashboard folder contains all vue files for connected users as Backend
- the src/styles folder contains all styles though it can also be include into vue files directly as well but it permits clearer sources codes organization

2.3.3 For server micro-service:

- the src/config folder contains as it suggests the configurations/settings of the application ;
- the src/interfaces and src/models folders contains the definition of collections, which fields they may have and on what types, etc.
- the src/routes folder contains all controllers and routes management of the API
- the server.ts file is the entry point of the application

2.4 Implementation

2.4.1 Source code versioning management

- Open Gitlab account for each team member
- Setup a Gitlab group and invite all team members
- Create the project (repository) that is hosting the code
- Using Sourcetree https://confluence.atlassian.com/get-started-with-sourcetree, a GUI software that enables developers to seamlessly clone, create and add repositories and conducting various actions such as, creating branches, merging working directory changes to those branches and pushing them to a remote origin. Conversely, certain developers prefer to use command line to perform Git actions, an example provided below:

```
git clone https://gitlab.com/oxford-brookes-
    university/2018-2019/P00412/client-microservice.
    git
git clone https://gitlab.com/oxford-brookes-
    university/2018-2019/P00412/server-microservice.
    git
```

or if developers have SSH enabled:

```
git clone git@gitlab.com:oxford-brookes-university
    /2018-2019/P00412/client-microservice.git
git clone git@gitlab.com:oxford-brookes-university
    /2018-2019/P00412/server-microservice.git
```

This enables developers to retrieve the repositories without authenticating themselves each time for any Git action.
Once the developer have made changes they can commit their changes to a branch, an example provided below:

```
git commit -am "My changes"
git push origin myonlinebranch
```

Note: Everyone will have to provide their own respective commit messages that illustrates the changes you have performed on the source code so other developers are aware of such changes. This involves changing the branch name "myonlinebranch" with another name that is unique to the changes.

You may also want to organize your git and follow some best practices.

2.4.2 For all setup

- Install node.js and yarn on your computer / server
- Buy and configure your domain name or subdomain if you want one
- Do the configuration with Cloudflare for more accessibility and security reasons

2.4.3 Setup on your computer / Compute Engine of Google Cloud Platform

- First, let's install the librairies:

```
yarn install
```

- Then, let's build (compile the application):

```
yarn build
```

- Then, let's launch the server microservice:

```
yarn start
```

- Finally, let's launch the client microservice (for development purpose):

```
      yarn serve
```

  Note: In production mode, you could use apache2 and nginx as reverse proxy and have the usual port 80 for http or 443 for https (TLS).

Note: if you are making changes, you need to re-run step 2-4.

You will notice that the server is launching on port 4000 and that the client (GUI for users) is on port 8080 (in development mode).

2.4.4 Setup on Kubernetes Engine of Google Cloud Platform with AutoDevOPS with Docker

- Open Google Cloud Platform account
- Enable AutoDevOps GitLab feature
- Create a Kubernetes cluster on it with the default configuration of their managed services
- Connect Google Cloud Platform account to the group or project in Gitlab
- Once we commit and push on Gitlab, with specific docker, gitlab and kubernetes files, it will deploy automatically on Google Cloud Platform

2.4.5 Implementing code changes

Developers will comply with this Git Workflow to implement any new changes to the project:

- Develop on local computer
- Commit to a separate branch named features/featureindevelopment
- Open a Work In Progres (WIP) Merge Request (MR) into the master (trunk) branch
- Continue to commit until the code is considered to be finished
- Remove the WIP flag on the MR
- Discuss with the team members about technical choices

- Implement modifications discussed in the previous steps
- Repeat steps 6-7 until the MR is approved by a team member

Each time a change is approved by the team and that the tests are validated, the code is merged into the trunk of the git repo. This triggers an AutoDevOps pipeline that executes these steps:

- Continuous Integration tests to avoid regression and make sure no bugs are introduced
- Build a Docker image to package the complete application
- Push it to Gitlab private Docker repository in order to make the image ready for deployment
- Deploy the image onto the Kubernetes cluster so the new code becomes accessible online

2.5 Useful Link

A Successful GIT Branching Model

Google Cloud Kubernetes

Gitlab AutoDevOPS

Bibliography

Codefresh, Inc. (2017). *Codefresh*. URL: https://codefresh.io.

Docker, Inc. (2013). *Docker*. URL: https://www.docker.com.

Electronic Frontier Foundation, Mozilla Foundation, University of Michigan, Akamai Technologies, and Cisco Systems (2014). *Let's Encrypt*. URL: https://letsencrypt.org.

Gitlab Inc. (2011). *Gitlab*. URL: https://gitlab.com.

Google, Inc. (2011). *Google Cloud Platform*. URL: https://cloud.google.com.

Twitter (2016). *Bootstrap-Vue*. URL: https://bootstrap-vue.js.org.

YOU, Evan (2014). *Vue.JS*. URL: https://vuejs.org.