Technology Investigation

Summary

for

Group Testing Environment

v2.0

15/04/2022

Prepared by

Filipe Lagrenade,

Sanjay Williams,

Jia Son Pow, and

Isaac Ellis

Capstone Group I3

**Table of Contents:**

[**Front End**](#_fvwpad7xosyt) **3**

[**Research Summary Document**](#_mw59wulxdgzy) **3**

[**Introduction**](#_oh96v9foxt98) **3**

[**Criteria**](#_j4ra9aett3uj) **3**

[**Most Suitable Option**](#_drnb6hfm5vy) **3**

[Reasons](#_cfm9ahamc36m) 4

[Other Options Considered](#_i6jr3lm8n36x) 4

[**Database**](#_fa9ffjhcgkk) **4**

[**Research Summary Document**](#_euqqfcjecq7n) **4**

[**Introduction**](#_mplmosovbv6g) **4**

[**Criteria**](#_pniqiqu5y2rs) **5**

[**Most Suitable Option**](#_5jii04ub0evf) **5**

[Reasons](#_o8b04ob0222) 5

[Other options Considered](#_cd8p0e6luk1z) 5

[**Middleware**](#_7x081brt3a9i) **5**

[**Research Summary Document**](#_yfmctkyty9nj) **6**

[**Introduction**](#_9qee5ru5uf0x) **6**

[**Criteria**](#_gye4r3fuc46e) **6**

[**Most Suitable Option**](#_dwmku3ctplhv) **6**

[Reasons](#_l3b2hax4tknp) 6

[Other Options Considered](#_fxvucu1jwodg) 7

[**Syncing**](#_y0kp1m4ckie6) **7**

[**Research Summary Document**](#_nwo8vxkrhajp) **7**

[**Introduction**](#_59oicjh7qci5) **7**

[**Criteria**](#_hdr0jgpeuzw) **8**

[**Most Suitable Option**](#_5vamozdvgr8) **8**

[Reasons](#_octqbv56c8x7) 8

[Other Options Considered](#_t8l02nhxllm0) 8

[**Client Development Framework**](#_2vu2p6tcdwoi) **8**

[**Research Summary Document**](#_sgw0548uukdu) **8**

[**Introduction**](#_nqz2trfda39f) **9**

[**Criteria**](#_iglr12t9cd6o) **9**

[**Most Suitable Option**](#_2fa4j1qgptc7) **9**

[Reasons](#_ezpbrdx1eknm) 9

[Other Options Considered](#_ctt2s7p8xzyd) 9

[**Group Agile Report**](#_xy3xf5s5z37q) **10**

# Front End

Contributors: **Jia Son Pow**

## Research Summary Document

[Front End Research Summary](https://docs.google.com/document/d/1yjSbL1n3t0OHq7UmkSfqJjSaOYrfy1h96avBvsCafBE/edit#)

## Introduction

For this project’s front-end architecture, there will be two applications. They are:

* + 1. A desktop-based application will be the interface through which the students can access the test papers and submit their results.
    2. A web-based application where the invigilators/unit coordinators can answer questions and see what computers are active/inactive as well as a dashboard from which the unit coordinator will be able to send out messages to every student sitting the test at that given time.

## Criteria

* Security
* Reliability
* Ease of Development
* Liveliness
* Open-source
* Documentation

## Most Suitable Option

[Vue.js](https://docs.google.com/document/u/0/d/1oM7sLrIe7dGA7dpQFwd0dPgKNybiCBtcl-EJ6lXdMQA/edit)

### Reasons

* Ease of Development
* Resource Usage
* Reliability
* Community Support

### Other Options Considered

* [React.js](https://docs.google.com/document/u/0/d/1IzKglJuh7Kz1SUON6AHRccE-vDK9n4xPN2ImViPHSLM/edit)
* [Angular.js](https://docs.google.com/document/u/0/d/1YxOz1NYXW7rc4WDgwbiDtsVkjsyE6TOjrSFMkTOc5nk/edit)
* [jQuery](https://docs.google.com/document/u/0/d/1LlV5FFUUikuSfxMJsEcs2akEk22WVG3KWt7qdAFBLzk/edit)

# 

# Database

Contributors: **Isaac Ellis**

## Research Summary Document

[Database Research Summary](https://docs.google.com/document/u/0/d/136athYWByb6lyE9svDQ9vfI34G8RKvWD-UH1yK45rm4/edit)

## Introduction

For the project's architecture, a database has been chosen to store metadata related to

* Messages sent between students and UC/invigilators
* Tests (times, dates, students, test papers)
* Auditing events
* Valid test computers
* Users' accounts and password
* Security Roles

The database will be accessed by our middleware software which will allow access to relevant data through an application interface. The scale of the data that needs to be stored and queried we predict to be small as the project is only being used within Curtin.

## Criteria

* Security
* Reliability
* Ease of development
* Liveliness
* Opensource

## Most Suitable Option

* [PostgreSQL](https://docs.google.com/document/u/0/d/1sez7MUTN4SW3lk3ZwhBD7yrJjMMUpuctuIinY8-ZOlA/edit)

### Reasons

* Presents as the correct choice of database. As it meets all of our criteria without any glaring disadvantages.
* MongoDB follows as a close second; it meets our criteria however it does not enforce a schema.

### Other options Considered

* [MongoDB](https://docs.google.com/document/u/0/d/17OUx5HefgDXti2EnZc4j7qunYncpISswf3dKLZZMPQw/edit)
* [SQLite](https://docs.google.com/document/u/0/d/13SNpSu9jdEwOUlPuVXYpisd9dX-kUI28-l2ce25iMuI/edit)

# Middleware

Contributors: **Sanjay Williams**

## Research Summary Document

[Middleware Research Summary](https://docs.google.com/document/d/18MV9-07i_TCedu1KtCF3t1Obr9ji0Vc-dbswEx9UPIE/edit?usp=sharing)

## Introduction

Middleware provides communication between applications, databases, and web servers. As it typically takes the load off the operating system and the developers themselves, finding the right middleware software for a project is essential. Most, if not all, criteria should be met, with cost-efficiency at the forefront given the relatively wide range of available choices.

## Criteria

* ‘Out-of-the-box’ Integration
* Open-Source
* Cost-effective
* Scalable, Robust Architecture

## Most Suitable Option

[Apache Tomcat](https://docs.google.com/document/d/1YI5tvluO6OaSTv3yoMp78zKG4dzmFeo2Hq_Q9W_y-d8/edit?usp=sharing)

### Reasons

* + - * Out-of-the-box setup and integration
      * Open-source
      * Free
      * Lightweight
      * Stable
      * Scalable
      * Secure

### Other Options Considered

* + - * [Apache Camel](https://docs.google.com/document/d/1DFNQ6epfwUh6LpOvj76ALtRyrdH55zXWQF_HaZ30xss/edit?usp=sharing)
      * [Apache Kafka](https://docs.google.com/document/d/1wQlCshBtpTgyDeBw92XieFcDIxgfoT9XOdFTpUIKM4Y/edit?usp=sharing)
      * [JBoss EAP](https://docs.google.com/document/d/1v7mENRayrElXLfNaa6z7Ihqt6z-_b7Cg4FPwIMR-S80/edit?usp=sharing)

# Syncing

Contributors: **Isaac Ellis**

## Research Summary Document

[Syncing Research Summary](https://docs.google.com/document/u/0/d/1pZgD3Wd2Vj0Z473vc1uPli9P5RUfnw0SU9wHCoapKM8/edit)

## Introduction

For the student application, we are required to synchronously backup files, to an external server, that a student creates and edits during a test.

This process fulfils **two goals:**

* Making sure that student files are saved so that UC/Invigilators can access them after the test.
* The ability for a student, during a test, to change computers and still access their files.

We would also like to meet **a third lower priority goal:**

* To create a backup system that implements version control on the files created and edited during a test. This would provide students and UC/Invigilators with a history of document changes.

## Criteria

* Security
* Reliability
* Liveliness
* Linux Support
* Opensource

## Most Suitable Option

* [Rsync](https://docs.google.com/document/u/0/d/16LFsHIRt51Ez6rt9adaY28N1CwugFCfuNPcFBzHKXfc/edit)

### Reasons

* Rsync presents a simple solution that would allow the developer to focus on other areas of the software project.

### Other Options Considered

* + - [Writing it ourselves](https://docs.google.com/document/u/0/d/1UMNfdEF3IhNzFCog2sU8QJyEDQv8seF5fD1MYhmU4Vg/edit)
* [Syncthing](https://docs.google.com/document/u/0/d/1uPKVlUdfpPbmOUD8VypHA63tf5TsdBaCLqhRqrkGU6k/edit)

# Client Development Framework

Contributors: **Filipe Lagrenade**

## Research Summary Document

[Client Development Framework Research Summary](https://docs.google.com/document/d/1kOGZ7IUhbYq6_Ro7tDlxBRFLAeM9_2_h_bLG4I4GaG0/edit?usp=sharing)

## Introduction

The Client Development framework will assist in creating native applications in Linux. Ideally, it should allow for the development of an application in any platform, including web, windows, Linux, and MacOS for future scalability if the client chooses to do so.

## Criteria

* Security
* Reliability
* Ease of development
* Cross-platform
* Open-source
* Liveliness

## Most Suitable Option

[Electron](https://docs.google.com/document/d/1H68XbAqdFXQyReBPMYR664th3UeIOQCEhZtKBIESqRE/edit?usp=sharing)

### Reasons

* Large Community Support
* Ease of Development
* Cross-Platform
* Compatibility with Node.JS
* Reliable
* Widely used

### Other Options Considered

* [Tauri](https://docs.google.com/document/d/1Y0MdbXR1xfvjcks5jkKcID_kWC4bMmUMd5FsnSWbGaw/edit?usp=sharing)
* [Neutralino](https://docs.google.com/document/d/10_Foj72da1KqSTxTbKOVNTLFF9DZYytv604er7SvwzY/edit?usp=sharing)

# Group Agile Report