



UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

EASY WRITTEN EXAM
PLATAFORMA PARA FACILITAR EL DISEÑO, LA ENTREGA Y LA EVALUACIÓN
DE EXÁMENES EN LÍNEA ESCRITOS

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN COMPUTACIÓN

MATÍAS EDUARDO PIZARRO ORELLANA

PROFESOR GUÍA:
JÉRÉMY BARBAY

MIEMBROS DE LA COMISIÓN:
FEDERICO OLMEDO BERÓN
PATRICIO INOSTROZA FAJARDIN

SANTIAGO DE CHILE
2022

RESUMEN

El presente trabajo muestra el desarrollo de una plataforma web para diseñar, responder y evaluar pruebas en línea. Esta plataforma intenta solventar tanto la falta de una estandarización sobre la forma en la que se han llevado a cabo las evaluaciones en línea durante la pandemia como la dificultad que, a veces, se presenta para los ayudantes (evaluadores) al corregir las evaluaciones sin tener una forma eficiente de dar *feedback*, lo que se traduce en una inversión mayor de tiempo.

La plataforma desarrollada tiene por nombre *EasyWrittenExam*, y cuenta con cuatro vistas principales: *Home*, *TestDesign*, *TestAnswering* y *TestEvaluation*, además de varias vistas auxiliares necesarias para hacer que todo se conecte y funcione, tanto a nivel de *front-end* como *back-end*.

La vista *TestEvaluation* implementa una forma de sugerir comentarios al momento de evaluar, con el fin de ahorrar tiempo en la corrección. Además, se ha implementado una forma de presentar las respuestas a los evaluadores de manera ordenada en base a la cantidad de palabras que tiene cada respuesta (*clustering*).

Las validaciones realizadas constan de dos fases. La primera fase de vistas, donde la plataforma no tiene implementada ninguna lógica ni funcionalidades, es decir, solo considera los *mockups* que muestran cómo se vería la plataforma. La segunda fase, donde ya se han implementado las funcionalidades básicas, es decir, un usuario, puede crear, responder y evaluar controles. Lamentablemente, solo una profesora se ofrece para probar la plataforma.

Los resultados obtenidos constan de una plataforma funcional que cumple con alguno de los objetivos, esto es, la implementación de las funcionalidades básicas. No se alcanza a implementar el *clustering* avanzado de repuestas ni el sugeridor de *feedback* basado en etiquetas para agilizar la evaluación de las respuestas.

*Dedico mi trabajo de memoria a mis padres, quienes siempre han soñado con que mi
hermana y yo seamos profesionales*

Agradecimientos

A mi mamá, papá y hermana, por ser un apoyo emocional enorme en este proceso. A mi abuelita Nancy, quien siempre ha estado ahí para lo que necesite. A Les Dines, por ayudarme siempre que lo necesité, en todos los ámbitos. A Ana Quezada, por subirme el ánimo siempre que lo necesité, y hacerme tirar para adelante. A Jérémy Barbay, por darme los mejores *tips* para mejorar mi manera de escribir reportes y por su excelente labor como guía. A mis amigos en general, y a las personas que me dieron una mano cuando lo necesité.

Tabla de contenido

Índice de tablas	vi
Índice de ilustraciones	vii
1. Introducción	1
1.1. Objetivos de la memoria	2
1.1.1. Objetivo General	2
1.1.2. Objetivos Específicos	2
1.2. Estructura general de la solución	3
1.3. Estructura del documento	4
2. Antecedentes	5
2.1. Situación Actual	5
2.1.1. Metodología Tradicional	5
2.1.2. Alternativas de plataformas de evaluaciones en línea	6
2.2. Tecnologías para el desarrollo de aplicaciones web	8
2.3. Oportunidades Digitales	10
3. Solución	11
3.1. Funcionamiento de la plataforma	11
3.1.1. Usuarios	11
3.1.2. Controles	12
3.1.3. Creación de un control	12
3.1.4. Evaluación de un control	13
3.1.5. Responder un control	13
3.2. Arquitectura de la aplicación	14
3.3. Modelo de la base de datos	15
3.4. Producto Mínimo Viable	16
3.4.1. Test Design	16
3.4.2. Test Answering	17
3.4.3. Test Evaluation	17
3.4.4. Home	18
4. Implementación	19
4.1. Funcionamiento general	19
4.1.1. LogIn	19

4.1.2.	Idiomas	19
4.1.3.	Apache	20
4.1.4.	React	20
4.1.5.	Express	20
4.1.6.	Postgresql	20
4.1.7.	Deployment	21
4.2.	Vistas Principales	21
4.2.1.	Home	21
4.2.2.	TestDesign	22
4.2.3.	TestAnswering	23
4.2.4.	TestEvaluation	24
4.3.	Flujo de la plataforma	26
5.	Evaluación de la solución	29
5.1.	Validación	29
5.2.	Encuesta	31
5.3.	Análisis de resultados	31
6.	Conclusiones y trabajo futuro	32
6.1.	Trabajo realizado	32
6.2.	Discusión y aprendizaje	33
6.3.	Trabajo futuro	33
	Bibliografía	35
	Anexos	38
A.	Entrevistas preliminares	39
A.1.	Análisis encuesta	39
A.2.	Entrevistas	40
B.	Proceso	41
B.1.	Calendario de Trabajo	41
B.2.	Desarrollo	42
C.	Otras Vistas	45
D.	Resultados de la encuesta de usabilidad EWE	47

Índice de tablas

2.1. Tabla comparativa de plataformas de evaluación en línea	8
--	---

Índice de ilustraciones

3.1. Arquitectura <i>Easy Written Exam</i>	14
3.2. Modelo relacional de <i>Easy Written Exam</i>	16
4.1. Vista Home	22
4.2. Vista TestDesign superior	22
4.3. Vista TestDesign inferior	23
4.4. Vista TestAnswering	24
4.5. Vista TestEvaluation	25
4.6. Vista SignUp	26
4.7. Vista CourseCreation	26
4.8. Vista TestDesign en flujo	27
4.9. Vista TestAnswering en flujo	27
4.10. Vista TestEvaluation en flujo	28
4.11. Vista TestReview en flujo	28
B.1. Calendario de Trabajo - Carta Gantt	41
C.1. Vista Login	45
C.2. Vista Previsualización	46
D.1. Resultados pregunta 1 de encuesta	47
D.2. Resultados pregunta 2 de encuesta	48
D.3. Resultados pregunta 3 de encuesta	48
D.4. Resultados pregunta 4 de encuesta	49
D.5. Resultados pregunta 5 de encuesta	49
D.6. Resultados pregunta 6 de encuesta	50

Capítulo 1

Introducción

La tecnología es algo que siempre ha ido evolucionando y que, al día de hoy, ha permitido ir cambiando la manera en la que las cosas se llevan a cabo, una de ellas es la educación [26, 28]. Particularmente, el internet ha dado paso a la educación en línea y a distancia, no importando en qué parte del mundo uno se encuentre.

Durante los años 2020 y 2021 se ha vivido una transición acelerada hacia la docencia en línea a nivel mundial debido a la llegada de la COVID-19, una pandemia que ha forzado a los países a tomar medidas para poder frenarla, como lo han sido las cuarentenas. Estas medidas han obligado a docentes y alumnos a dejar los espacios físicos para comenzar a realizar las actividades educativas a través de internet.

Varios aspectos de dicha transición pueden presentar ventajas que se quedarán incluso tras un eventual fin de la pandemia que se vive. Entre estas ventajas se encuentra la digitalización de las evaluaciones. Tal digitalización presenta varias oportunidades para facilitar el trabajo de quienes diseñan y hacen evaluaciones. Por ejemplo, una digitalización completa de las evaluaciones facilitaría su reutilización tanto en tiempo (mismo equipo docente) como en espacio (distinto equipo docente dictando un curso similar).

Dada la tecnología que se tiene a mano, existe un potencial enorme para hacer múltiples mejoras en materias de educación en línea. Continuando en la línea de la digitalización, en los sistemas de evaluaciones en línea se podrían incorporar características que ayuden a automatizar la corrección y la detección de plagio, a agrupar (*clustering*) los trabajos de los estudiantes para dar retroalimentación (*feedback*) grupal, a facilitar la reutilización de *feedback* anterior, entre otras cosas. Para dar un ejemplo, consideremos la situación en donde varios alumnos cometieron errores tan similares que podrían recibir no solamente una nota similar, sino que *feedback* similar para que entiendan el porqué de su error. Otro caso es donde alumnos tuvieron errores similares a los que ya tuvieron otros alumnos que ya pasaron por ese mismo curso, por lo que el *feedback* que se entrega podría ser *feedback* que ya se tenía almacenado.

¿Cuánto más facilitaría la evaluación el preprocesar un conjunto de respuestas de alumnos para proponer a los evaluadores grupos de respuestas similares? ¿Cuáles tipos de soluciones se

pueden preprocesar y con qué técnicas? ¿Cómo se podrían indexar respuestas y retroalimentaciones anteriores para poder proponer reutilizarlos eficientemente en instancias futuras? Aunque las repuestas a estas preguntas no fueron parte de este trabajo, son un interesante conjunto de problemas para abordar en un trabajo de tesis.

En este trabajo de memoria se desarrolla un sistema de gestión de aprendizaje (*learning management system* o LMS), en este caso, una plataforma web de evaluaciones en línea llamada *Easy Written Exam* (EWE). Esta plataforma es *open-source* y tiene como propósito facilitar la labor de profesores, evaluadores y estudiantes a la hora de diseñar, contestar y evaluar exámenes en línea escritos. Las funcionalidades de esta plataforma van desde algunas básicas como poder crear una instancia de evaluación, poder responder y corregir tal evaluación, hasta algunas más complejas como la ordenación de respuestas según algún criterio, el guardado de *feedback* para futuras correcciones y la sugerencia de *feedback* para reutilizar, todo esto apuntando a disminuir el tiempo invertido por parte de los evaluadores en la corrección de las pruebas.

Este trabajo contempla la formalización de la solución, el diseño de *Easy Written Exam*, la creación de un producto mínimo viable (*Minimum Viable Product* o MVP) de la plataforma con sus funcionalidades básicas y la validación de este MVP. Algunas de las posibles funcionalidades más complejas que se mencionan quedan propuestas como trabajo futuro, debido al tiempo acotado con el que se contó para este trabajo de memoria.

1.1. Objetivos de la memoria

1.1.1. Objetivo General

El objetivo general de este trabajo de título es el de diseñar y desarrollar una plataforma *open-source* de evaluaciones en línea llamada “*Easy Written Exam*” para facilitar el proceso de diseño, entrega, repuesta y corrección de las evaluaciones en línea. Además, entregar una plataforma base para poder trabajar sobre las hipótesis mencionadas anteriormente en la introducción¹.

1.1.2. Objetivos Específicos

Para lograr nuestro objetivo general, se definen los siguientes objetivos específicos:

1. Identificar y formalizar los puntos importantes de una plataforma que permita las tareas descritas, explicando en particular cada parte de la plataforma que se construirá.
2. Diseñar una plataforma web que implemente una selección de funcionalidades básicas tales como: poder crear una evaluación, poder responder esta evaluación y, finalmente, poder corregirla, apuntando principalmente a un público universitario científico (MVP).

¹Preguntas retóricas del 5to párrafo de la introducción.

3. Validar de manera breve que el sistema cumple con medidas básicas de usabilidad tanto para los profesores como para los alumnos y evaluadores, tanto en el contexto de la escuela de ingeniería de la FCFM, como también en otras instituciones tales como otras universidades en Chile, o incluso universidades en el extranjero.
4. Perfeccionar las funcionalidades claves como la sugerencia del *feedback* y agrupación de respuestas en base a la similitud.
5. Opcionalmente, integrar la plataforma con U-Cursos.
6. Documentar diseño, implementación y procesos de validación, de forma que el trabajo pueda ser fácilmente extendido por otra persona.

En particular, aunque el proyecto *Easy Written Exam* consideraba validar las siguientes hipótesis: “El agrupamiento (*clustering*) de las respuestas de los estudiantes permite dar retroalimentación (*feedback*) grupal” y “La plataforma ahorra tiempo de trabajo de los ayudantes gracias la reutilización de *feedback* anterior”, tales validaciones no entran en el propósito del trabajo propuesto, sino que simplemente se contempla la producción de un prototipo que permita, teniendo más tiempo, validar dichas hipótesis.

1.2. Estructura general de la solución

La plataforma desarrollada cuenta con 4 vistas principales: La vista *Home*, la cual muestra una tabla con el resumen de las últimas actividades dependiendo del usuario; la vista para el diseño de una evaluación (*TestDesign*), la cual permite a los profesores crear un control; la vista para corregir una evaluación (*TestEvaluation*), a la cual tienen acceso solamente los evaluadores; y finalmente la vista para responder una evaluación (*TestAnswering*), a la cual tienen acceso todos los estudiantes.

Los usuarios de la plataforma no tienen roles estrictamente definidos, sino que sus roles van a depender de la relación que tengan con un curso. Un usuario que ingresa a la plataforma no tiene ningún curso asociado, y por lo tanto no tiene ningún rol. El usuario puede crear un curso, del cual se convertirá en profesor/a; puede unirse a un curso, al cuál entrará como estudiante; puede ser ingresado a un curso como evaluador/a solamente si el profesor o la profesora del curso lo añade como tal.

Con respecto al ordenamiento de las respuestas de los estudiantes en la vista de evaluación, el método utilizado se basa simplemente en la cantidad de palabras que contiene cada respuesta. Este ordenamiento se realiza en el servidor, y las respuestas se muestran en la vista, ordenadas de manera decreciente en una lista lateral.

Por último, la sugerencia de *feedback* funciona solamente para una misma pregunta, es decir, el *feedback* sugerido para una pregunta serán los comentarios anteriormente usados para corregir la misma.

1.3. Estructura del documento

Este documento se divide en 6 capítulos. Se comienza por mencionar la situación actual (Capítulo 2), se explica cómo se lleva a cabo el proceso de las evaluaciones, las tecnologías disponibles y el estado del arte. Además, se explican todos los conceptos necesarios para entender cómo se desarrolla la plataforma.

Luego, se explica en detalle la plataforma propuesta como solución (Capítulo 3), se habla sobre las distintas vistas, la arquitectura y el modelo relacional en detalle la solución. Además, se presenta el producto mínimo viable.

Más adelante, se encuentra la implementación (Capítulo 4). Aquí, se detallan las fases del desarrollo, cómo funciona la lógica y el flujo de la plataforma.

Posteriormente, se habla sobre la evaluación llevada a cabo, algunas encuestas y un pequeño análisis sobre los resultados de las validaciones de la plataforma (Capítulo 5).

Para terminar, se pasa a las conclusiones respectivas al trabajo realizado (Capítulo 6). Esta última sección presenta una pequeña discusión y el trabajo futuro propuesto.

Capítulo 2

Antecedentes

En este capítulo se presenta el contexto en el cual se trabaja, detallando cuales son las metodologías y plataformas que existen hoy en día (Sección 2.1). También, se habla de las tecnologías disponibles para desarrollar una plataforma (Sección 2.2) y cuáles son las oportunidades digitales que se logran visualizar (Sección 2.3).

2.1. Situación Actual

A continuación se presenta una revisión de cómo se realizan las evaluaciones con la metodología tradicional (Sección 2.1.1). Además, se habla del estado del arte en lo que respecta a las diferentes plataformas web de *tests* en línea que existen (Sección 2.1.2).

2.1.1. Metodología Tradicional

Durante mucho tiempo, el papel y lápiz ha sido el método utilizado en las aulas de clases para realizar evaluaciones. Tradicionalmente, el profesor entrega a cada alumno un conjunto de hojas con las preguntas en ellas (enunciado), y el alumno debe responder a estas preguntas en esas mismas hojas, o bien, usar hojas nuevas en blanco.

Si nos situamos en el contexto de la Facultad de Ciencias Físicas y Matemáticas de la Universidad de Chile, la situación es, que en la mayoría de los cursos, los controles son de preguntas de desarrollo (preguntas tipo ensayo), no de alternativas. Normalmente, un control o examen consta de un número acotado de preguntas (inferior a 10 y muy cercano a 3). Estas preguntas se responden en hojas separadas, esto permite más tarde a los ayudantes (evaluadores) dividir el trabajo de corregir las pruebas, dedicándose cada uno a ciertas preguntas. Todo lo anteriormente mencionado no es una regla, sino como funciona en general.

No obstante, como se ha mencionado, la pandemia nos ha obligado a dejar de lado esta manera de evaluar, para hacerlo todo frente una pantalla, un *mouse* y un teclado. Para lograr

describir el cómo se han estado llevando a cabo las evaluaciones durante este tiempo, se han realizado entrevista de las que participó el ayudante Benjamín Vera, el profesor Guillermo Fuentes, ambos de la universidad de Chile y Willy Maikowski, ingeniero civil en computación y parte del equipo de la plataforma U-Cursos de la Universidad de Chile.

A modo de resumen, Willy nos cuenta que ya existe una sub-plataforma dentro de U-Cursos para hacer pruebas, la cual permite hacer preguntas de desarrollo. Dicha plataforma ha sido usada en muy pocos cursos, básicamente porque no se continuó con el proyecto y es algo que está en beta. Benjamín comenta que en los cursos del departamento de ingeniería matemática (DIM) se usa una plataforma del departamento que se llama “Reclamos DIM”, pero tiene pocas funcionalidades. El profesor Guillermo dice que para sus cursos utiliza una plataforma web externa llamada *Testmoz*, plataforma que no es gratuita si se quiere acceder a todas sus funciones.

En síntesis, no hay una plataforma común que se haya estandarizado para llevar a cabo este proceso de evaluaciones en línea, y estas plataformas que ya existen no presentan todas las herramientas que se necesitan o no son gratuitas. Esta es, entonces, una motivación para desarrollar algo que llegue a ser usado en toda la universidad por igual.

2.1.2. Alternativas de plataformas de evaluaciones en línea

En esta sección se describen algunas plataformas web de evaluaciones en línea que se lograron encontrar y se comparan algunas de sus características.

2.1.2.1. Software disponibles en línea

Al día de hoy ya se encuentran disponibles variadas herramientas digitales que facilitan la tarea de tomar exámenes en línea. En muchos países ya se han ido incorporando plataformas de educación *online* tanto en colegios como en universidades [27, 25, 29]. Para poder hacer un pequeño análisis comparativo de estas herramientas, se seleccionaron algunas de ellas y se crearon cuentas en estas para probar algunas funcionalidades, se vio videos demostrativos de gente utilizando estas aplicaciones, y por último, se leyó reseñas [1, 6]. Sin embargo, acá se extraen solo algunas características de cada una, las más relevantes y acordes al interés principal de esta memoria. Habiendo dicho esto, se da paso al listado de plataformas de evaluaciones en línea.

- ***Proprofs Quiz Maker*** [15]: Permite agregar imágenes al momento de crear las preguntas. No tiene una interfaz clara. No permite preguntas de desarrollo en su versión gratuita.
- ***PaperShala*** [13]: Se puede administrar a los alumnos (activar/desactivar). Se puede tener un conjunto de preguntas creadas para utilizar posteriormente. Soporta imágenes en las preguntas.
- ***Quizalize*** [17]: Se puede cambiar idioma entre español, inglés, chino y árabe. Se puede iniciar sesión con Google. Se puede usar material creado por otros usuarios (exámenes

ya hechos). Se puede subir archivo .pdf para crear control. Solo se puede tener 3 clases (cursos) como docente en la versión gratuita. Permite audio en las preguntas (útil para clases de idiomas). Solo permite preguntas de selección múltiple.

- **Edbase** [5]: Cuenta gratuita solo permite añadir 10 alumnos. Los alumnos se unen mediante invitación del profesor. Bloquea la página del *test* con tal de no poder visitar otras páginas mientras se realiza el examen.
- **FlexiQuiz** [8]: Tiene preguntas de texto libre. Se puede aleatorizar el orden de las preguntas. Solo permite 10 preguntas por *test* en la versión gratuita. Se puede dar *feedback* en las respuestas. Un *test* solo puede ser respondido por 20 personas en la versión gratuita. Se puede bloquear la opción de *copy/paste*.
- **Questbase** [16]: Plataforma en inglés e italiano. Tiene preguntas de texto libre. El evaluador tiene la opción de “fácilmente” imprimir la evaluación. Interfaz no es amigable.
- **Testmoz** [21]: Se puede elegir idioma para la interfaz del estudiante. Se puede aleatorizar el orden de las preguntas. Los alumnos pueden ver todas las preguntas en una misma página durante la evaluación.

2.1.2.2. Comparación de software

Principalmente, las características que se toman en cuenta son características no tan comunes. Todas estas plataformas ofrecen las funcionalidades básicas para poder crear evaluaciones (de selección múltiple en su mayoría). A continuación, se muestra un listado con las características que se evaluaron con más énfasis:

- **Múltiples idiomas:** Se apunta a que, a largo plazo, la plataforma pueda ser usada en múltiples países. Se podría pensar en el idioma inglés, ya que es un idioma universal. Sin embargo, no todas las personas hablan inglés en los países de habla hispana, por lo que es fundamental que la plataforma desarrollada cuente al menos con dos idiomas, en este caso, español e inglés.
- **Estadísticas:** Esta característica es bien útil para hacer estudios del curso más adelante, y así poder extraer información rápida y valiosa del rendimiento de los alumnos a modo general.
- **No permitir *copy/paste*:** Si bien, esta característica no es para nada fundamental, e incluso hay técnicas para quitarse esta restricción de encima, parece interesante de considerar. Esto, a modo de dificultar la búsqueda de las preguntas en internet y la ayuda durante un control.
- **Feedback grupal:** Una de las metas importantes de este proyecto es la de dar *feedback* a los alumnos a través de la evaluación. El hecho de dar *feedback* grupal pretende tomar en cuenta el agrupamiento de las respuestas en base a su similitud, y poder mandar un mismo *feedback* a varios alumnos al mismo tiempo.
- **Reutilización de *feedback* anterior:** Para ahorrar tiempo al momento de redactar el *feedback*.
- **Navegar entre preguntas durante evaluación:** Esta característica es una de las más simple, pero igualmente importante, dado que un alumno debería poder tener la

posibilidad de cambiar o corregir sus respuestas durante el control, al igual que en la situación presencial.

- **Respuesta permite subir imagen:** Por último, esta característica es esencial para las respuestas de desarrollo, ya que la idea es que el alumno tenga la facilidad de responder en una hoja si así le acomoda y tener la oportunidad de escanear y subir su respuesta como una imagen.

A continuación, ya habiendo mencionado las características a considerar, se muestra una tabla comparativa para las distintas plataformas y las características.

Característica \ Plataforma	ProProfis	PaperShala	Quizalize	Edbase	FlexiQuiz	QuestBase	Testmoz
Múltiples idiomas	S	N	S	N	N	S	S
Estadísticas	S	N	S	S	N	N	S
No permite <i>copy/paste</i> de enunciados	N	N	N	N	S	N	P
<i>Feedback</i> grupal	N	N	N	N	N	N	N
Reutilización de <i>feedback</i> anterior	N	N	N	N	N	N	N
Navegar entre preguntas durante evaluación	S	S	S	S	S	S	S
Respuesta permite subir imagen	P	N	N	N	N	N	N

Tabla 2.1: Tabla comparativa de plataformas de evaluación en línea. Las letras S, N y P significan “Sí”, “No” y “Pagado” respectivamente.

2.2. Tecnologías para el desarrollo de aplicaciones web

En cuanto a las herramientas, *frameworks* o bibliotecas de desarrollo de aplicaciones web existen varias opciones: *Vuejs*, *Django*, *React*, *Angular*, *Laravel* y *Ruby on Rails* son algunas de ellas. Después de comparar algunas ventajas y desventajas de cada una, se concluyó que cualquiera de estas opciones entrega las herramientas necesarias para desarrollar una plataforma web del estilo de EWE, el punto está entonces en elegir lo que más se ajuste al programador. Para este trabajo, se apostó por que *Javascript* sería un lenguaje que facilitaría el rápido avance debido a que ya se tenía algo de experiencia. Además, se eligió *React* ya que es una tecnología muy popular y ampliamente documentada¹, por lo que encontrar un programador que continúe con este trabajo será mucho más sencillo.

En resumen, se opta por *React* para desarrollar el *front-end* de la aplicación. Para el *back-end*, se decide usar *NodeJs*, siguiendo con la filosofía de que programar en *Javascript* permitiría un avance más rápido. Ahora bien, con lo que respecta a los datos, se ha decide usar una base de datos relacional, como lo es *Postgresql*. Esta decisión se basa nuevamente en la experiencia que ya se poseía con esta tecnología, lo cual agilizaría un poco más el desarrollo. Además, las bases de datos relacionales, si bien son menos escalables, tienen un mejor rendimiento a la hora de hacer consultas [31].

¹Se ha comparado la popularidad en Google Trends: <https://trends.google.com/trends/explore?date=all&q=%2Fg%2F11c0vmgx5d,%2Fm%2F01211vxv,%2Fm%2F0j45p7w,%2Fm%2F0505c1>

Conceptos tecnológicos

A continuación se presentan conceptos relacionados con la tecnología que se decidió utilizar:

- **Git** [9]: Es un sistema de control de versiones distribuido de código abierto y gratuito diseñado para manejar todo, desde proyectos pequeños hasta muy grandes, con velocidad y eficiencia.
- **Gitlab** [10]: Es una plataforma *DevOps* basada en *Git* y que abarca todo el ciclo de vida del desarrollo de software. El servicio permite gestionar, administrar, crear y conectar los repositorios con diferentes aplicaciones y hacer todo tipo de integraciones con ellas.
- **Trello** [23]: Es una herramienta de gestión flexible del trabajo, en donde se puede idear planes, colaborar en proyectos, organizar flujos de trabajo y realizar un seguimiento del progreso de una manera visual, productiva y gratificante de las tareas.
- **React** [19]: Es una biblioteca de *JavaScript* para construir interfaces de usuario. Su principal característica es la componetización².
- **Redux** [20]: Es un contenedor de estado para aplicaciones *Javascript*. Ayuda a escribir aplicaciones que se comportan de manera consistente, se ejecutan en diferentes entornos (cliente, servidor y nativo) y son fáciles de probar.
- **JSON Web Token (JWT)** [12]: Es un estándar abierto (RFC 7519³) que define una forma compacta y autónoma de transmitir información de forma segura entre las partes como un objeto *JSON*. Esta información se puede verificar y confiar porque está firmada digitalmente. Los *JWT* se pueden firmar usando una llave secreta (con el algoritmo *HMAC*) o un par de llaves pública / privada usando *RSA* o *ECDSA*
- **Bootstrap** [4]: Es una biblioteca *open-source* de herramientas para *front-end* que permite crear sitios rápidos y responsivos.
- **Internationalization (i18n)** [11]: Es el proceso de adaptar una plataforma o sitio web a diferentes lenguajes y localizaciones.
- **Servidor Apache** [3]: Es un servidor web *HTTP open-source*.
- **Application Programming Interface (API)** [18]: Es un conjunto de procedimientos, definiciones y protocolos que se utiliza para desarrollar e integrar aplicaciones web entre sí.
- **Express** [7]: Es un *framework* para aplicaciones web *back-end* de *NodeJS*. Provee las herramientas para desarrollar *APIs*.
- **Postgresql** [14]: Es un sistema *open-source* de gestión de bases de datos relacional de objetos.
- **Tmux** [22]: Es un multiplexor de terminal para sistemas tipo *Unix*. Permite dividir una consola en múltiples secciones o generar sesiones independientes en una misma terminal.

² *Componetizar* se utiliza como término para crear nuevos componentes más atómicos: <https://worksh ops-js-montevideo.gitbook.io/react/patrones-avanzados/18-componetizacion>

³ RFC 7519 <https://datatracker.ietf.org/doc/html/rfc7519.html>

2.3. Oportunidades Digitales

El hecho de pasar de hacer todo de forma presencial a hacer las cosas a través de un computador, forzó una aceleración brutal en el proceso de digitalización de la docencia, evidenciando problemas de desigualdad social, como lo son las personas que no tienen un computador que se adecúe a los requisitos de los cursos, o incluso personas que no tienen acceso a internet [2]. También, está la falta de conocimiento tanto de parte de profesores como de alumnos para hacer uso de la tecnología. No obstante, digitalización también plantea muchas oportunidades para hacer la docencia, quizás contradictoriamente, más accesible y más eficiente. Un ejemplo de lo anterior serían las grabaciones de clases o videos cápsulas, las cuales se pueden ver al ritmo del alumno, de manera asíncrona y reutilizar de semestre a semestre, de institución a institución, o incluso de país a país.

Las tecnologías educacionales incluyen una gran variedad de dominios de desarrollo. Si se habla de lo material, se considera una gran cantidad de dispositivos, desde proyectores que muestran apuntes en un telón y computadores personales interconectados, hasta tecnologías inteligentes como teléfonos, entornos virtuales, computación en la nube, aparatos *wearable* y *location-aware* [30]. Respecto al software, se considera por una parte el software dirigido a quienes aprenden, como los software de simulación y de visualización, y por otra el software dirigido a la administración del aprendizaje, con los LMS⁴ y su integración vía “Interoperabilidad entre Herramientas de Aprendizaje” (*Learning Tools Interoperability* o LTI)⁵.

⁴Learning Management Systems: https://en.wikipedia.org/wiki/Learning_management_system

⁵Learning Tools Interoperability: https://en.wikipedia.org/wiki/Learning_Tools_Interoperability

Capítulo 3

Solución

Este capítulo explica en detalle la plataforma propuesta como solución, cómo funciona esta (Sección 3.1), cuál es la arquitectura del sistema (Sección 3.2) y el modelo relacional (Sección 3.3). Además, se presenta el producto mínimo viable (Sección 3.4).

3.1. Funcionamiento de la plataforma

Básicamente, la plataforma desarrollada cuenta con 4 vistas principales: La vista principal (*Home*), la cual muestra una tabla con el resumen de las últimas actividades dependiendo del usuario; la vista para el diseño de una evaluación (*TestDesign*), en la cual un profesor puede crear un control para uno de sus cursos; la vista para corregir una evaluación (*TestEvaluation*), en la cual los evaluadores pueden poner nota a las respuestas de los estudiantes para un control; y finalmente la vista para responder una evaluación (*TestAnswering*), a la cual tienen acceso los estudiantes para responder una evaluación.

3.1.1. Usuarios

Existen 3 tipos de usuarios: Profesores, Evaluadores y Estudiantes. Sin embargo, el rol de un usuario no está fijamente definido en la plataforma, sino que será más bien el rol del usuario en un curso. Por ende, un usuario puede ser profesor en un curso, evaluador en otro curso y estudiante en otro curso distinto.

- **Profesor/a:** Un usuario se convierte automáticamente en profesor/a cuando crea un curso. El profesor del curso tiene la facultad de crear controles para el curso y agregar otros miembros al curso, tanto profesores, como evaluadores y alumnos.
- **Evaluador/a:** Un usuario solamente puede convertirse en evaluador/a de un curso si este es añadido al curso como tal por el profesor. El evaluador solamente puede evaluar controles ya respondidos.

- **Estudiante:** Un usuario puede formar parte de un curso como estudiante simplemente yendo a la sección **Cursos > Unirse a un curso**. Allí, podrá navegar por los distintos cursos y hacerse parte de uno haciendo *click* en **Unirse**¹.

3.1.2. Controles

Los controles son las evaluaciones que se pueden diseñar en la plataforma. Un control solamente puede ser creado por un profesor.

Un control puede poseer, sin límite de cantidad, los siguientes tipos de preguntas:

- **Pregunta de desarrollo:** Pregunta que consta de un enunciado y, opcionalmente, imágenes. El estudiante debe responder en un campo de texto abierto.
- **Pregunta de selección múltiple:** Pregunta que consta de un enunciado, una lista de alternativas y, opcionalmente, imágenes. El estudiante debe seleccionar la alternativa que crea correcta. Solo puede seleccionar una alternativa
- **Pregunta de verdadero o falso:** Pregunta que consta de un enunciado, dos opciones (verdadero o falso) y un campo de texto. El estudiante debe elegir la opción que crea correcta pudiendo justificar su decisión en el campo de texto.

Además, un control tiene un título, un curso, fecha y hora de inicio, fecha y hora de término, instrucciones y duración.

3.1.3. Creación de un control

Como se mencionó anteriormente, solo uno de los profesores de un curso tiene la facultad para crear controles para dicho curso. Para llevar a cabo el diseño de un control, el usuario deberá ir a la sección de controles en la barra lateral izquierda y luego hacer *click* en **Nuevo control**. Esto lo lleva a una vista especializada para la creación de controles (*TestDesign*).

TestDesign es una vista que cuenta con un formulario que permite al usuario ingresar los datos que un control requiere para su creación. Un usuario puede añadir preguntas de los distintos tipos y remover preguntas añadidas anteriormente.

Posteriormente, el usuario podrá previsualizar el control en una vista que simula el cómo se vería este al ser respondido por el estudiante.

En la primera versión de la plataforma, solo se pueden crear controles, no se pueden editar ni borrar tras su creación. Eso es algo que está pendiente.

¹Se es consciente de que esto puede ser una forma insegura de unirse a un curso. Por ende, se pretende agregar una capa de seguridad, haciendo que **Unirse** simplemente mande una solicitud de ingreso al curso y luego el profesor tendrá la opción de aceptar o rechazar dicha solicitud

3.1.4. Evaluación de un control

Para poder tener acceso a la vista de evaluación de un control (*TestEvaluation*), el usuario debe pertenecer al curso de dicho control con rol de evaluador. Una vez dentro, puede ver a un costado izquierdo la lista de preguntas. Al seleccionar una pregunta, al costado derecho le aparecerá la lista de respuestas de todos los estudiantes que hayan respondido hasta la fecha. Finalmente, al hacer *click* en una respuesta se le mostrará en el centro de la pantalla el enunciado de la pregunta junto a la respuesta del alumno.

El ordenamiento de preguntas, en primera instancia, se ha hecho en base a la cantidad de palabras que posee la respuesta. Para esto, no se considera ningún tipo de *stop word*². Este ordenamiento permite que el evaluador vea las respuestas que contienen más texto al principio de la lista.

El ordenamiento de las preguntas tiene como objetivo dejar juntas las respuestas que se parezcan más, por lo que más adelante se debe cambiar el método de agrupamiento. Se debe considerar el contenido del texto por sobre la cantidad de palabras, ya que esto último no nos dice nada a nivel semántico.

Por último, la sugerencia de *feedback*, en primera instancia, se ha desarrollado en base a la pregunta, pero más adelante se desea implementar en base a etiquetas. Actualmente, cuando un evaluador comenta (deja *feedback*) una respuesta, este *feedback* se sugiere en una lista para las siguientes respuestas de la misma pregunta. Esto debería aminorar el trabajo de escribir *feedback* para tal pregunta, ya que se puede reutilizar el que se sugiere.

El punto de hacer funcionar el *feedback* mediante las etiquetas de las preguntas, es el de poder reutilizar el *feedback* no solo dentro de una misma pregunta, sino que también en distintos controles con preguntas que sean similares y que por lo tanto tengan etiquetas similares.

3.1.5. Responder un control

Un usuario que se encuentra dentro de un curso con el rol de estudiante puede responder los controles de dicho curso (acceso a la vista *TestAnswering*).

La vista *TestAnswering* muestra las instrucciones del control en primera instancia, junto con un botón que le permite al usuario comenzar al control. Cuando un usuario hace *click* en **Comenzar**, comienza a correr el tiempo (duración del control). En el costado izquierdo, se muestra un botón que abre un modal con las instrucciones. En el costado derecho se muestra el tiempo restante, la lista de preguntas con los números coloreados según corresponda, y un botón para enviar el control. En el centro de la vista se muestra la pregunta seleccionada con los respectivos campos para responder.

El usuario hace efectiva la respuesta a una pregunta cuando hace *click* en **Responder**

²Stop word según Wikipedia: https://en.wikipedia.org/wiki/Stop_word

pregunta justo debajo de la pregunta. Cuando el usuario hace *click* en el botón para enviar el control, se le abre un modal que le avisa en caso de que no haya contestado todas las preguntas.

3.2. Arquitectura de la aplicación

La arquitectura de EWE es la de una plataforma *server-side rendering* basada en *Apache*, quien sirve en el cliente los archivos estáticos generados con *React* en el *front-end*. Por el lado del *back-end* se provee una *API REST* montada en *Node.js*, la cual se conecta a la base de datos relacional en *Postgresql*.

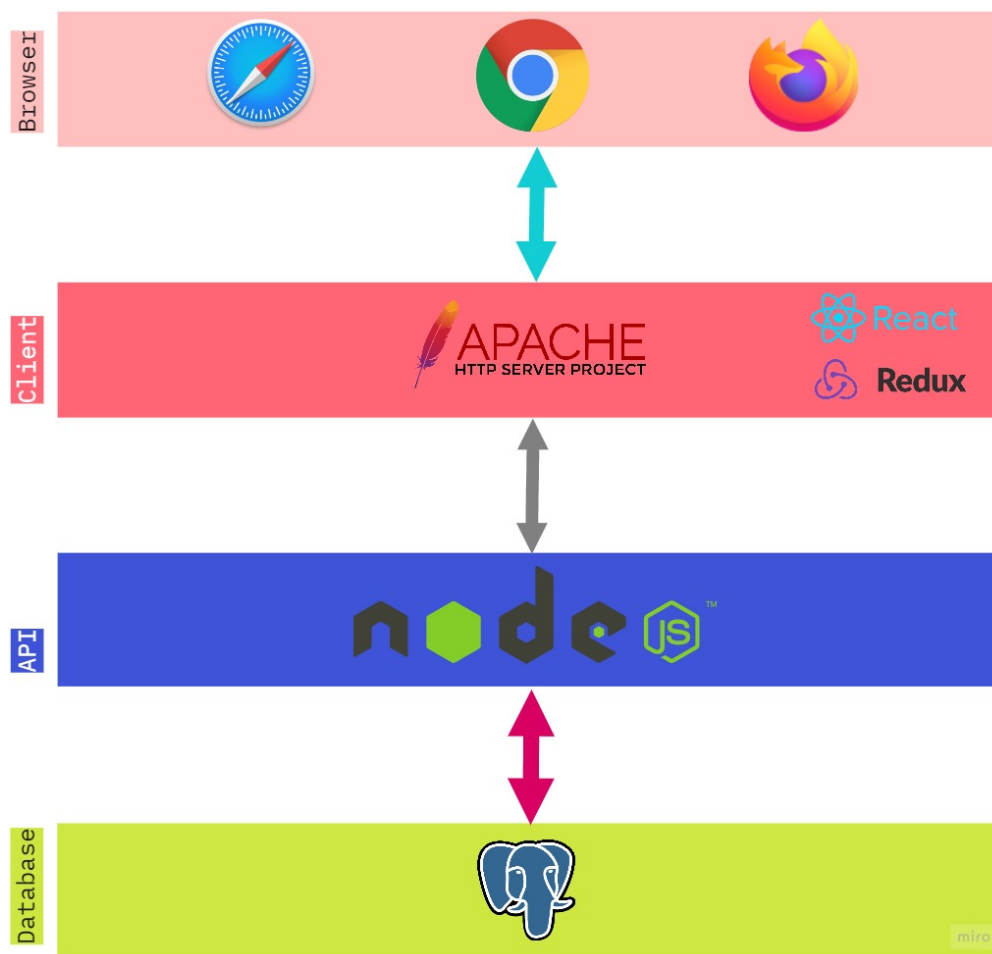


Figura 3.1: Arquitectura de *Easy Written Exam* basada en *Apache*, quien sirve en el cliente los archivos estáticos generados con *React* en el *front-end*. Por el lado del *back-end* se provee una *API REST* montada en *Node.js*, la cual se conecta a la base de datos relacional en *Postgresql*.

3.3. Modelo de la base de datos

El esquema de datos cuenta con los siguientes modelos o entidades:

1. `ewe_role`: Rol de un usuario en la plataforma (`User` o `Admin`). Solo se usa el tipo `User` por ahora, pero se espera que a futuro existan usuarios de tipo `Admin` con más permisos
2. `role`: Rol de un usuario dentro de un curso (`Professor`, `Evaluator` o `Student`)
3. `user`: Usuario
4. `course`: Curso
5. `inscription`: Instancia que representa la inscripción de un usuario en un curso
6. `question_type`: Tipos de pregunta (*Essay*, *Multiple Choice* o *True or False*)
7. `question`: Pregunta
8. `essay_question`: Pregunta de desarrollo
9. `mc_question`: Pregunta de alternativas o selección múltiple
10. `tof_question`: Pregunta de verdadero o falso
11. `feedback`: *Feedback* o comentario
12. `test`: Control
13. `papersheet`: Representa la hoja de un control. Es la instancia que asocia a un estudiante (`user`) con un `test`
14. `answer`: Respuesta
15. `essay_answer`: Respuesta de desarrollo
16. `mc_answer`: Respuesta de alternativas
17. `tof_answer`: Respuesta de verdadero o falso. Contiene la justificación.
18. `evaluation`: Consiste en la nota y el *feedback* con los que se evalúa un control

A continuación se presenta un esquema relacional, en donde cada componente tiene sus atributos y muestra con qué otro modelo se relaciona. La imagen en formato *svg* se encuentra disponible en *Drive de Google* para ser mejor visualizada³.

³<https://drive.google.com/file/d/1cG-b6dwHz1p1LWMhjtOZMUrhKKyqbEin/view?usp=sharing>

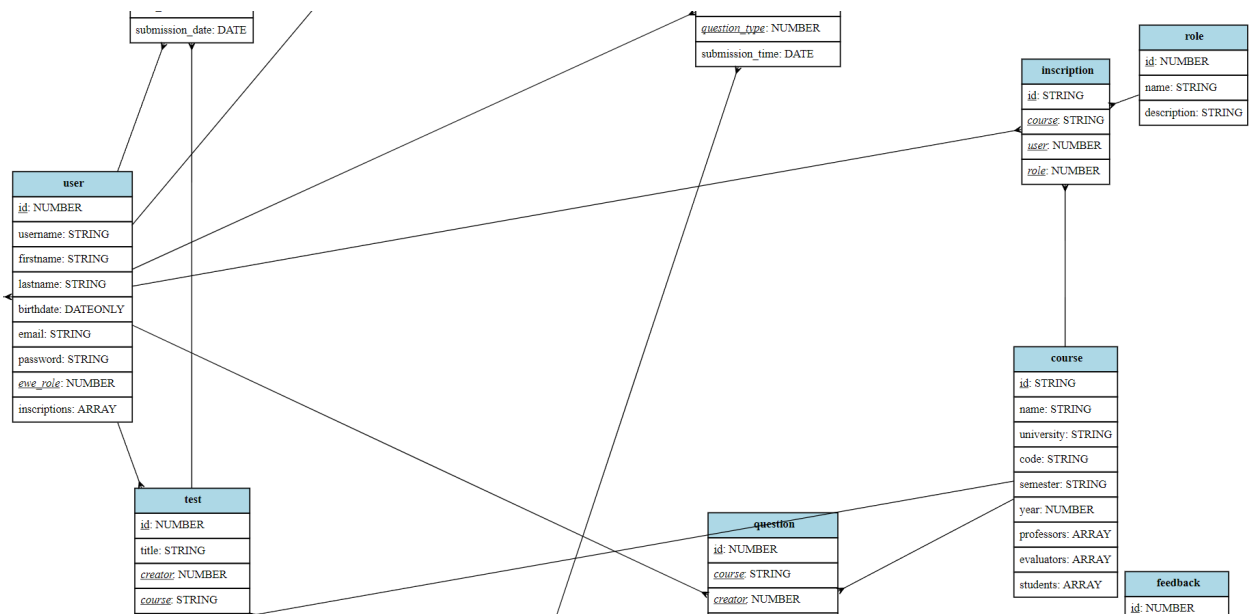


Figura 3.2: Segmento del modelo relacional *Easy Written Exam*. El modelo de *Inscription* es uno de los principales, pues aquí es donde se almacena el rol que tiene un usuario en un determinado curso.

3.4. Producto Mínimo Viable

En el área de computación se ha visto que un desarrollo progresivo e iterativo es mejor, en el sentido de que permite darnos cuenta de errores y validar cuanto antes la aplicación a través de un MVP [24]. Por esta razón, es importante tener el MVP de *Easy Written Exam* lo antes posible. Para nuestro MVP se tuvo en cuenta solamente una selección de *features* “básicos” con la finalidad de entregar una plataforma simple y que permita lo esencial, es decir, crear una evaluación, responderla, y luego poder revisarla y evaluarla con nota.

A continuación, se define un producto mínimo viable con las siguientes características según las vistas que tiene:

3.4.1. Test Design

1. Se puede agregar título al control
2. Se puede elegir un curso al cual va a pertenecer el control
3. Se puede agregar fecha y hora de inicio para el control
4. Se puede agregar fecha y hora de término para el control
5. Se puede definir la duración del control
6. Se puede agregar instrucciones
7. Se puede agregar pregunta de desarrollo

8. Se puede agregar pregunta de alternativas
9. Se puede agregar pregunta de verdadero o falso
10. Cada pregunta tiene nombre y enunciado
11. Las preguntas de alternativas tienen 5 alternativas
12. Las preguntas de desarrollo y de alternativas pueden tener imágenes
13. Se pueden eliminar preguntas ya agregadas
14. Se puede previsualizar el control antes de crear

3.4.2. Test Answering

1. Se muestran las instrucciones al inicio del control
2. Se muestra una pregunta por página
3. Se puede navegar entre las preguntas con botones de paginamiento
4. Se pueden volver a ver las instrucciones en un modal
5. Se muestra el tiempo restante
6. Se muestran las preguntas respondidas
7. Se indica en color verde las preguntas respondidas, en gris las que faltan por responder y en amarillo la pregunta seleccionada.
8. Las preguntas de desarrollo tienen un *input* de tipo *text* para responder
9. Las preguntas de alternativas tienen *input* de tipo *radio* para responder
10. Las preguntas de verdadero o falso tienen un *select* para V o F y un *input* de tipo *text* para la justificación
11. Las preguntas son respondidas al apretar un botón para responder cada pregunta
12. Al responder una pregunta, se marca como respondida en la lista de preguntas
13. Se muestra un botón que permite responder el control
14. Al responder el control se responden solo las preguntas en las cuales se apretó el botón de Responder pregunta
15. Al apretar Enviar control se muestra un modal para confirmar
16. El modal de Enviar control muestra advertencia en caso de que queden preguntas por responder

3.4.3. Test Evaluation

1. Se muestra una lista con las preguntas del control en donde el usuario puede elegir una pregunta para evaluar
2. Al seleccionar una pregunta, se muestra una lista con las respuestas de los estudiantes, en donde el usuario puede elegir una respuesta a evaluar
3. Al seleccionar una respuesta en particular correspondiente a un estudiante, se muestra la respuesta seleccionada con la pregunta relacionada en pantalla

4. Se muestra un *input* de tipo *number* para colocar nota
5. Se muestra un *input* de tipo *text* para añadir feedback opcional
6. Se muestra botón de evaluar pregunta para enviar la evaluación y generar la entrada en la base de datos.
7. Las respuestas para las cuales se presionó el botón **Evaluar pregunta** se marcan como revisadas en la lista de respuestas

3.4.4. Home

1. Se muestra barra de navegación izquierda
2. Barra de navegación muestra opción de **Home**, **Cursos**, **Controles**
3. En la opción de **Home** se muestra tabla con historial de evaluaciones
4. En la opción de **Cursos** se muestra lista de cursos, se puede crear un curso o unirse a alguno
5. En la opción de **Controles** se puede crear un control
6. La tabla principal muestra en cada fila: nombre del curso y del control, fecha y hora de inicio y término, estado del control, rol del usuario en el curso de dicho control y una acción a ejecutar.
7. Opción de **Responder** se muestra para estudiante y permite ir a responder un control
8. Opción de **Evaluar** se muestra para evaluadores y permite ir a evaluar un control
9. Opción de **Ver** se muestra para estudiantes y permite revisar las respuestas una vez respondido el control

Capítulo 4

Implementación

En este capítulo se da a conocer el funcionamiento general de la plataforma (Sección 4.1), se muestran y explican las distintas partes de las vistas principales (Sección 4.2), y se presenta un flujo completo dentro de EWE (Sección 4.3).

4.1. Funcionamiento general

4.1.1. LogIn

El login de la plataforma se ha desarrollado utilizando una guía de *bezcoder*¹. Las tecnologías que se han utilizado son *Redux*, *React* y *Jwt* para manejar la autenticación. Un usuario, al presionar el botón **ingresar**, manda una *request* con su email y su contraseña. El controlador verifica si el usuario se encuentra en la base de datos. Si es así, el controlador devuelve un *token* que el usuario puede usar para navegar por la aplicación.

4.1.2. Idiomas

Para lograr traducir la plataforma a distintos idiomas y de manera escalable, se ha usado *i18n*. La carpeta *translation*, dentro de *client*, contiene todos los archivos *.json* con las traducciones. Esto significa que dentro de la aplicación en sí no hay ningún texto en bruto o forzado (*hardcoded*). De esta forma, si se desea agregar un nuevo idioma, solo se debe crear una nueva carpeta para el nuevo idioma con los archivos *.json* correspondientes dentro.

¹<https://www.bezkoder.com/react-hooks-redux-login-registration-example/>

4.1.3. Apache

El servidor *Apache* que se usa sirve los archivos estáticos que se encuentran en una carpeta llamada *public_html*. En esta carpeta deben ir todos los archivos que genera *React* con `npm run build`.

4.1.4. React

El *front-end* se encuentra en *client/*. Aquí, se ha creado una carpeta *pages* para los archivos con las vistas (14 vistas en total). Estas vistas complementan a las 4 principales para que pueda existir un flujo completo dentro de la aplicación. Debido a que se usa *React*, se tiene la carpeta *components* en donde están todos los elementos que se han *componetizado*. Las carpetas de *actions*, *reducers* y *services* corresponden a la parte de *Redux* que se utiliza para el *login* y el *logout*, a través del manejo de estado de la aplicación.

4.1.5. Express

El *back-end* maneja toda la lógica de las rutas y controladores. En él se encuentran todos los modelos también. Cuando llega una *request*, esta es manejada por `app.use(<api/ruta>, <Ruta>)`. Cada `<Ruta>` se exporta desde archivos particular alojado en *app/routes*, estos archivo contienen los *endpoints* que se ejecutan para una llamada (`router.get`, `router.post`, etc), utilizando `express.Router`.

Desde la ruta se llama a los controladores, quienes se encargan de la lógica necesaria y la conexión con la base de datos a través de los modelos. Los modelos están definidos con *sequelize* en forma de objetos. *Sequelize* ayuda a gestionar la base de datos gracias a distintos métodos para hacer las consultas pertinentes (`findAll`, `findOne`, `findByPk`, `create`, `update`, `delete`, etc)

4.1.6. Postgresql

Se ha creado una instancia de base de datos, con la cual se establece la conexión usando *pg* y *pghstore*. Se tiene un archivo *db.sql* con todos los modelos a crear. El contenido de este archivo se copia y se pega manualmente en una consola *psql* para crear los modelos por primera vez, es decir, esto solo se hace la primera vez. También, se tiene un archivo *db.commands.sql* con las primeras inserciones para inicializar la base de datos, esto se refiere a los tipos de usuarios, los tipos de preguntas, etc.

4.1.7. Deployment

En la máquina de buho² se tiene una carpeta para este proyecto de memoria. Dentro de dicha carpeta, existe una carpeta donde corre el *back-end*, y otra llamada `public_html`, que utiliza apache para servir los archivos estáticos (*front-end*).

Una vez terminados los cambios en local, se prueban usando también apache en local y luego se hace un *merge* a la rama de *develop* en *Git*. Se cambian algunas constantes como `API_URL` para que las *requests* apunten a *buho* y se ejecuta `npm run build`. La carpeta *build* creada se envía al servidor, a la carpeta `public_html` usando *scp*.

Una vez los archivos están arriba, en el servidor se va a la carpeta del proyecto, se corre `git pull` para traer los cambios del *back-end*. Ahora se abre una terminal con *tmux* y se deja corriendo la *API* con `npm start`. Luego, se sale de *tmux* con `ctrl + B ->D` y el *back-end* queda funcionando.

4.2. Vistas Principales

A continuación se presentan las 4 vistas principales:

4.2.1. Home

En la Figura 4.1 se muestra la vista principal de un usuario que es profesor del curso *Deep Learning* y del curso *Desarrollo Web*, ambos con un control. En la vista se tienen los siguientes elementos:

1. **Logo de EWE:** Lleva al inicio desde cualquier vista al ser *clickado*
2. **Sidebar:** Permite navegar entre Inicio, Cursos y Controles
3. **Nombre del usuario:** *Dropdown* menu con algunas opciones
4. **Tabla de controles:** Muestra los últimos controles de todos los cursos del usuario
5. **Acción:** Puede ser *Responder* o *Ver* para un estudiante, y *Evaluar* para un evaluador.
6. **Botones de navegación:** Paginamiento

²<https://buho.dcc.uchile.cl>

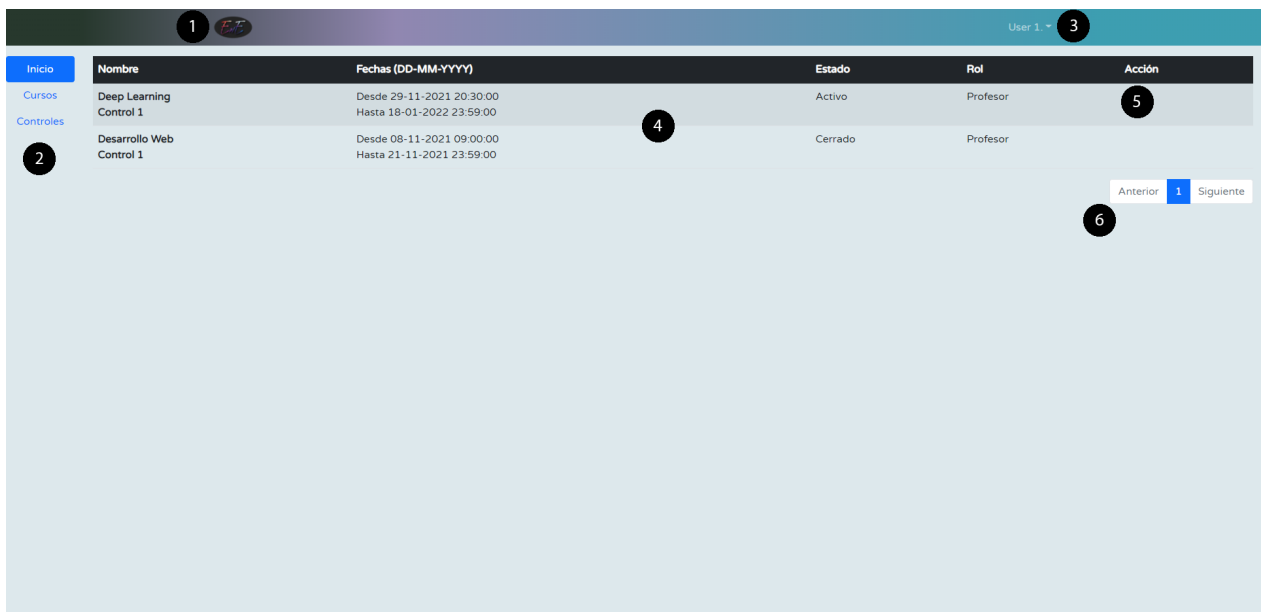


Figura 4.1: Vista Home. Se indexan las distintas secciones de la vista.

4.2.2. TestDesign

La Figura 4.2 muestra la parte superior de la vista para la creación de un control. En la vista se tienen los siguientes elementos:

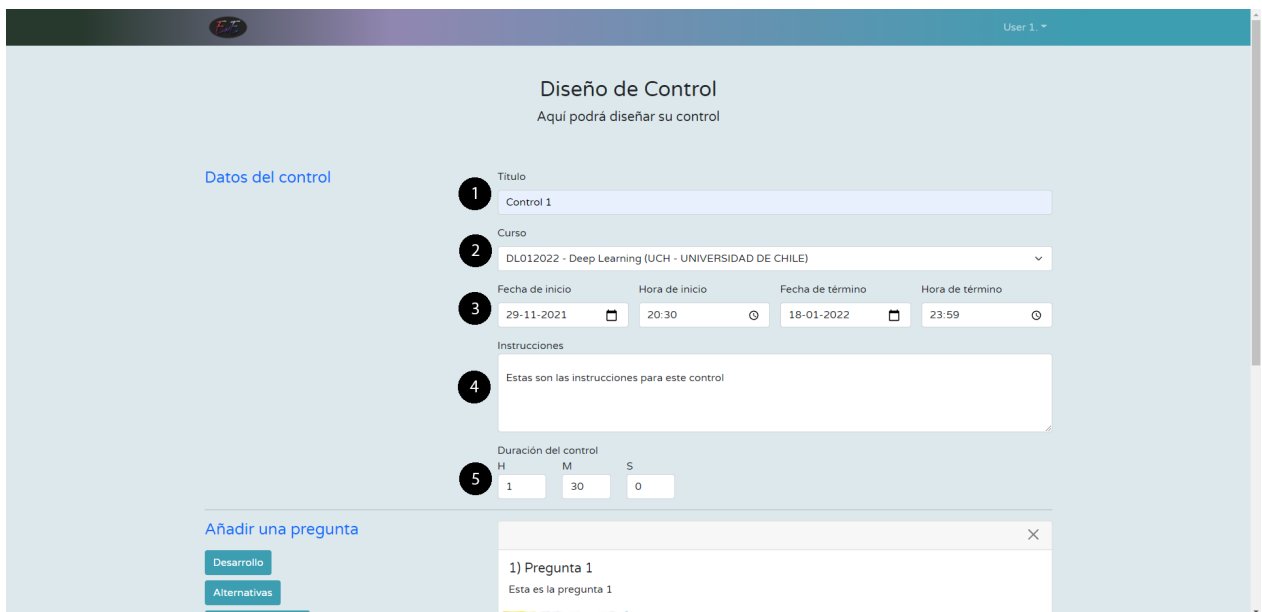


Figura 4.2: Vista TestDesign. Muestra la parte superior de la vista. Se indexan las distintas secciones de la vista.

1. **Campo Título:** Campo para ingresar el título del control
2. **Campo Curso:** Entrega como opciones todos los cursos de los cuales el usuario es profesor
3. **Camops de fechas y horas:** Permite establecer las fechas y horas de inicio y término del control
4. **Campo Instrucciones:** Campo opcional para agregar instrucciones
5. **Campos Duración:** Para agregar la duración del control. H son las horas, M son los minutos y S son los segundos

La Figura 4.3 muestra la parte inferior de la vista para la creación de un control. En la vista se tienen los siguientes elementos:

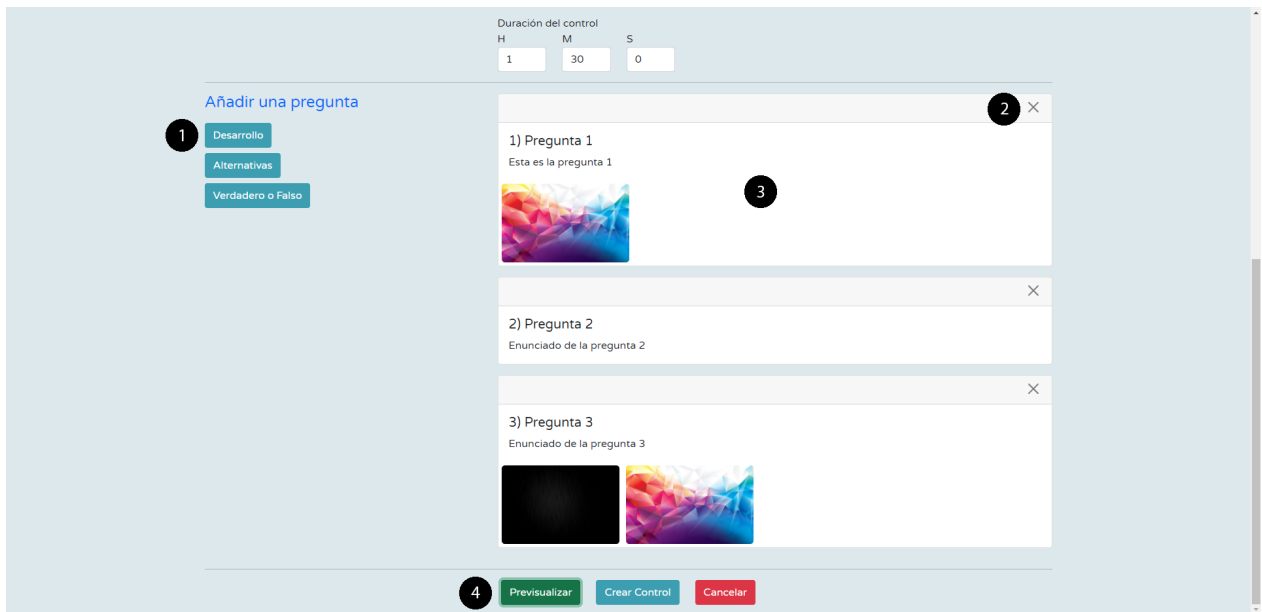


Figura 4.3: Vista TestDesign. Muestra la parte inferior de la vista. Se indexan las distintas secciones de la vista.

1. **Botones para añadir preguntas:** Tres botones que pueden añadir preguntas de desarrollo, de alternativas y de verdadero o falso
2. **Botón para eliminar preguntas:** Elimina la pregunta
3. **Carta de pregunta:** Carta que muestra la pregunta con su título, su enunciado, sus imágenes (si aplica), y alternativas (también si aplica)
4. **Botones inferiores:** Botón para previsualizar el control, botón para crear el control y botón para cancelar

4.2.3. TestAnswering

La Figura 4.4 muestra la vista para responder un control. En ella se tienen los siguientes elementos:

1. **Botón de instrucciones:** Muestra un modal con las instrucciones
2. **Información del la pregunta:** Titulo y Enunciado de la pregunta
3. **Imágenes:** Imágenes de la pregunta en caso de haber
4. **Campo de respuesta:** Campo de texto en el caso de preguntas de desarrollo, *radio-buttons* en caso de pregunta de alternativas, y *select* más campo de texto para preguntas de verdadero o falso
5. **Panel de tiempo:** Muestra el tiempo que queda
6. **Botón de respuesta:** Responde la pregunta actual. La pregunta se colorea de verde en el panel de preguntas
7. **Panel de preguntas:** Muestra los números de las preguntas coloreados. Amarillo para la pregunta en la que se encuentra el usuario, verde para las preguntas respondidas, y gris para las preguntas no respondidas
8. **Botón de envío:** Envía el control
9. **Botones de paginamiento:** Para navegar entre las preguntas del control

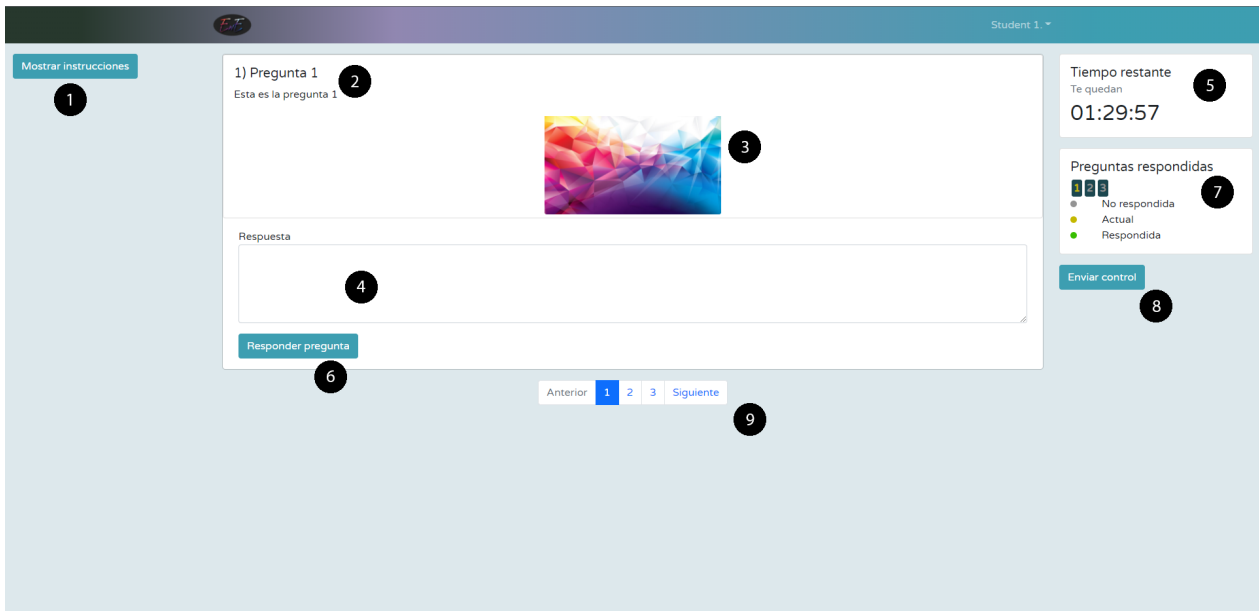


Figura 4.4: Vista TestAnswering. Se indexan las distintas secciones de la vista.

4.2.4. TestEvaluation

La Figura 4.5 muestra la vista para evaluar un control. En ella se tienen los siguientes elementos:

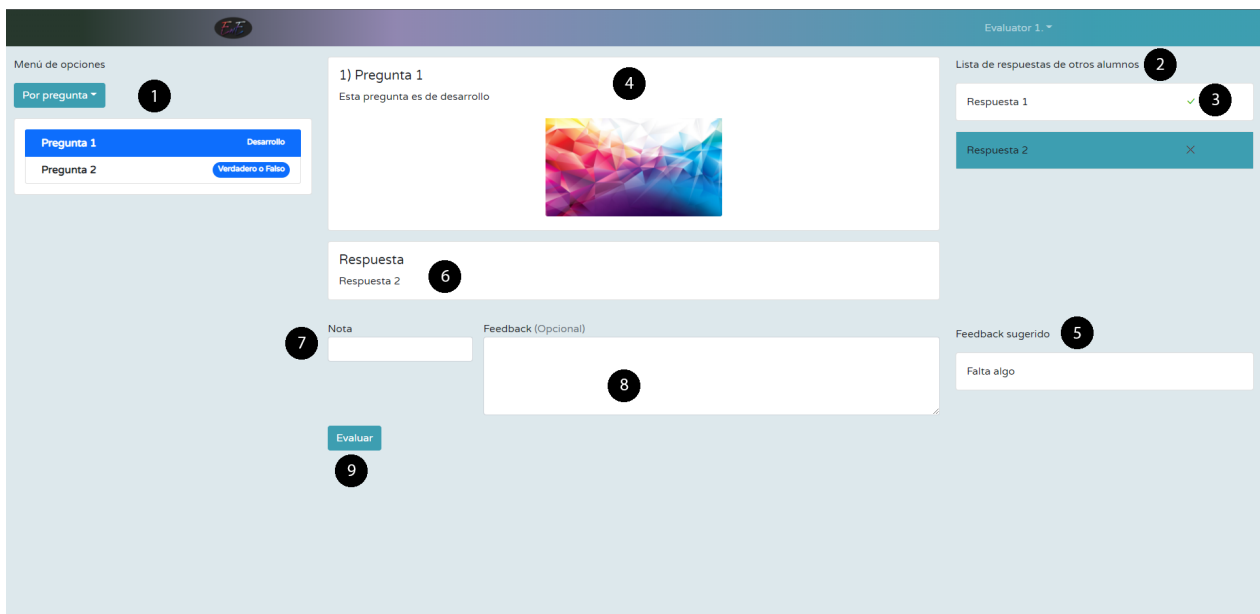


Figura 4.5: Vista TestEvaluation. Se indexan las distintas secciones de la vista.

1. **Lista de preguntas:** El evaluador puede elegir qué pregunta evaluar a partir de una lista
2. **Lista de respuesta:** Se muestran una lista con las repuestas de los estudiantes para la pregunta seleccionada. La respuesta seleccionada se marca con un color verde agua oscuro
3. **Estado de la respuesta:** Un visto verde si la pregunta ya fue evaluada. Una cruz roja si no
4. **Pregunta:** Enunciado de la pregunta
5. **Feedback sugerido:** Lista con sugerencias de *feedback*
6. **Respuesta:** Respuesta del alumno seleccionado
7. **Campo Nota:** Nota del 1 al 7. Permite *float*
8. **Campo Feedback:** Campo para escribir comentario (Feedback)
9. **Botón Evaluar:** Evaluar pregunta

4.3. Flujo de la plataforma

A continuación se presenta un caso de uso, junto a su flujo en la plataforma.



Figura 4.6: Vista SignUp. Los campos del formulario se han llenado con la información del usuario.

El flujo comienza con un usuario que no tiene acceso a la plataforma, UP de ahora en adelante. UP se encuentra en la página de login. En este caso, UP pincha en “registrarse”, lo cual lo redirecciona a la vista de *SignUp* (Figura 4.6). Ahora el usuario ingresa sus datos, elige una contraseña y pincha en “registrarse”, lo que le mostrará un mensaje de éxito y le dará la opción de volver al login.

UP ingresa a plataforma con sus credenciales. En este momento, no se le muestra nada en la vista de inicio, ya que no tiene historial. Se muestra un mensaje de “No hay evaluaciones aún”.

UP se dirige a **Cursos > Crear un nuevo curso**. Elige “Universidad de Chile” (la única opción por el momento) y completa el resto de datos. En el campo “semestre” se espera algo como “1-Otoño” o “2-Primavera” (Figura 4.7).

Una vez se crea el curso, UP se convierte en profesor del curso. Ahora el usuario se dirige a **Controles > Nuevo control**. Completa los datos principales del control, elige el curso que creó y comienza a agregar preguntas. Agrega una pregunta de desarrollo y una de alternativas. Luego, decide eliminar la pregunta de alternativas y agrega una de verdadero o falso (Figura 4.8). Antes de crear el control, UP visualiza el control. UP crea el control, vuelve a la página de inicio y ve que hay un control abierto.

Un segundo usuario se registra en la aplicación (UE). UP, mientras tanto, se dirige a la vista de su curso pinchando en **Cursos** y luego en la fila que contiene su curso. Hace *click* en *más* (+) y agrega a UE como evaluador del curso, poniendo el email de UE en el campo de email, y eligiendo “evaluador” en rol.

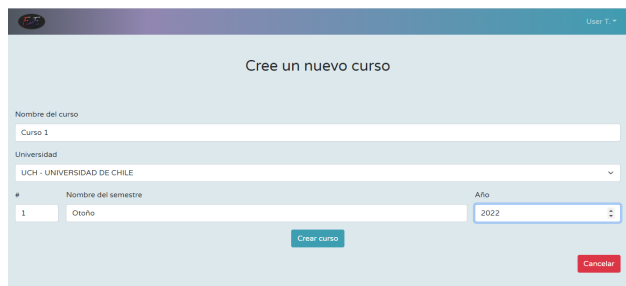


Figura 4.7: Vista CourseCreation. Los campos del formulario se han llenado con la información del curso.

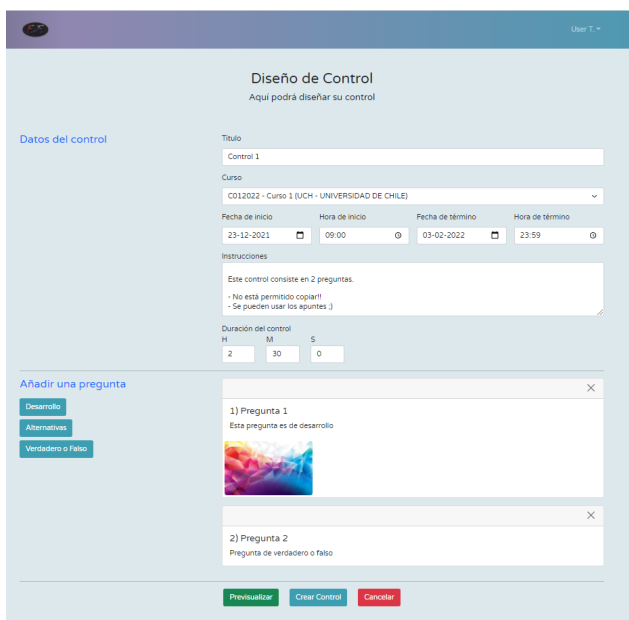


Figura 4.8: Vista TestDesing. Los campos del formulario se han llenado con la información del test. En este caso se ha agregado un pregunta de desarrollo con una imagen, y una pregunta de verdadero o falso.

desarrollo, presionando **Responder pregunta**. US presiona enviar control, se le muestra un modal con una advertencia de que aún le quedan preguntas por responder, por lo que decide volver atrás. US va a la pregunta de verdadero o falso y responde la pregunta. Finalmente, envía el control.

Ahora US decide ver las respuestas de su control, estas aún no han sido revisadas. UE, por otro lado, tiene la opción de evaluar el control creado por UP e ingresa a la vista de evaluación. UE selecciona la primera pregunta y se le muestra la respuesta de US a mano derecha. UE selecciona la única respuesta y se le muestra el enunciado y la respuesta en medio de la pantalla (Figura 4.10). UE coloca una nota y deja un comentario, y envía la evaluación presionando **Evaluar**.

US vuelve a entrar a la plataforma y ve sus respuestas. Ahora se le muestra la evaluación para la primera pregunta por parte de UE (4.11). La pregunta 2 aún no ha sido

Un tercer usuario se registra en la aplicación (US). US ingresa a EWE, y se encuentra en la vista de inicio. Ahora, US va a **Cursos > Unirse a un curso** y busca seleccionando Universidad de Chile en la opción de universidad. Se le muestra el curso que existe, en donde UP es profesor y UE es evaluador. US presiona **Unirse**, se le muestra un modal para preguntarle si está seguro/a y, luego de aceptar, US pasa a ser estudiante del curso.

A US se le muestra el control creado por UP en la tabla del inicio de la plataforma. US puede dar el control presionando en **Responder**. US lee las instrucciones del control y le da a **comenzar**, lo que lo manda a la vista de *TestAnswering* (Figura 4.9). Comienza a correr el tiempo de la prueba, US no recuerda bien las instrucciones, así que presiona el botón de instrucciones en el lado izquierdo de la pantalla, lo que le muestra un modal con las instrucciones.

US puede navegar por las preguntas con los botones de paginamiento en la parte inferior de la vista. US responde la pregunta de desarrollo, presionando **Responder pregunta**. US presiona enviar control, se le muestra un modal con una advertencia de que aún le quedan preguntas por responder, por lo que decide volver atrás. US va a la pregunta de verdadero o falso y responde la pregunta. Finalmente, envía el control.

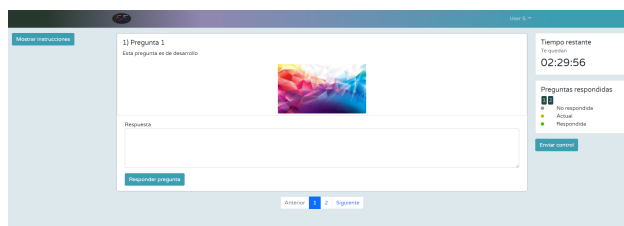


Figura 4.9: Vista TestAnswering. Al usuario se le muestra una pregunta de desarrollo. Hay dos preguntas en este control. El tiempo comienza a correr desde las dos horas y medias en cuenta regresiva.

evaluada.



Figura 4.10: Vista TestEvaluation. Al costado izquierdo se muestra la pregunta del control seleccionada, mientras que a mano derecha se muestra la respuesta del alumno seleccionada.

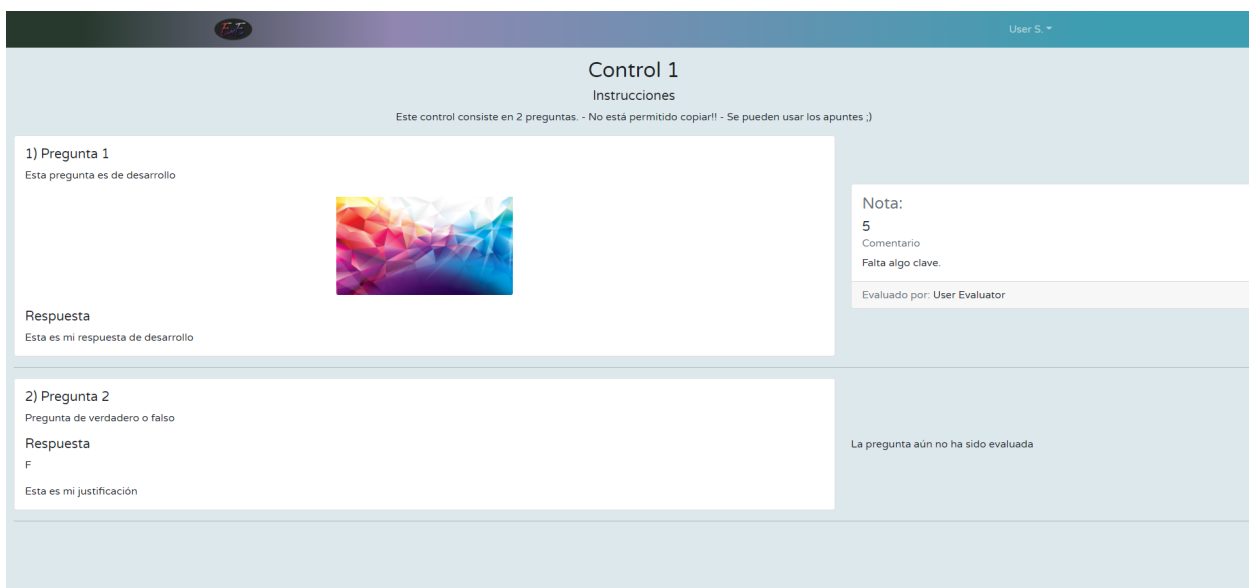


Figura 4.11: Vista TestReview. La pregunta 1 fue corregida por UE. La pregunta 2 aún no ha sido evaluada

Capítulo 5

Evaluación de la solución

El presente capítulo profundiza sobre la validación llevada a cabo (Sección 5.1), algunas encuestas realizadas (Sección 5.2) y un pequeño análisis sobre los resultados de las validaciones de la plataforma (Sección 5.3).

5.1. Validación

La validación de EWE se llevó a cabo en dos fases. La primera, habiendo desarrollado solo interfaz de usuario (*User Interface* o UI), ha tenido como propósito evaluar la distribución de los elementos en las vistas. La segunda, habiendo ya desarrollado las funcionalidades básicas, ha tenido como propósito evaluar la simpleza y eficacia de las funcionalidades.

En un comienzo, fueron cuatro las personas que se ofrecieron a probar la plataforma para poder probarla y validar su diseño y funcionamiento. Estas personas fueron Guillermo Fuentes (Facultad de Ciencias Astronómicas de la Universidad de Chile), Jorge San Martín (Departamento de Ingeniería Matemática de la Facultad de Ciencias Físicas y Matemáticas de la Universidad de Chile (FCFM)), Tonka Simunovic (Departamento de Ingeniería Química y Biotecnología de la FCFM) y Luca Castelli (Laboratoire d'Informatique École Polytechnique) y Jérémy Barbay (Departamento de Ciencias de la Computación de la FCFM). Sin embargo, una vez enviados los emails con las solicitudes de validación, solo dos profesores contestaron, afirmando que se encontraban disponibles. No obstante, a la hora de darle las instrucciones de cómo probar la plataforma, fue solamente una persona quién respondió los correos con sugerencias y comentarios sobre la plataforma.

En la primera fase de validación, el usuario envió comentarios sobre qué cosas estaban bien y qué cosas se podrían agregar/mejorar:

“Lo que me gusta de la plataforma es que resulta bien intuitiva, como que no se necesita una guía de uso ni nada, me gusta también que esté diseñada para responder ahí mismo y lo de añadir un archivo para complementar la respuesta”. A lo que añade: “Algo que le agregaría es, que en el caso de la tercera evaluación que dice Cerrado, se puedan revisar las respuestas

ingresadas (puede servir para corroborar que si se subió un archivo, este sea el correcto y se haya subido bien) y en la evaluación donde se tiene el rol del evaluador agregaría lo mismo de seleccionar un archivo, esto para poder añadir una posible pauta de la pregunta”.

Para la segunda fase de validación, el mismo usuario de la primera fase, quien nuevamente se dio el tiempo de enviar comentarios, dice: “Creé una cuenta para la página y me uní al curso de desarrollo web, al crear la cuenta me gustó que se pueda cambiar el idioma automáticamente y no tener que recargar la página. Además, al responder la evaluación esta súper lo de responder las preguntas individualmente y que se marquen las respondidas así es mucho más fácil llevar la cuenta”.

A lo que añade: “Cree también un curso, y no entendí muy bien la opción al lado del nombre del semestre, la que está marcada con # y que solo deja poner hasta el 2, pero al crearlo salio todo bien. Lo único sí, es que cuando quise ver cómo quedó en la pestaña de cursos entendí que solo es posible ver los cursos en la pestaña de mis cursos. Está bien la opción, pero quedaría mucho más cómodo si mis cursos y cursos llevaran a los cursos creados, y estuvieran incluidos los botones de crear y unirse a un curso”.

Finaliza con: “Al crear una evaluación añadí imágenes a una pregunta de desarrollo, y si se añade más de una se distorsionan un poco, al añadir sólo una por pregunta se ve bien. Encontré cómoda la manera de crear una evaluación, no tuve ningún problema. Está bien intuitivo, y lo de previsualizar ayuda montón a tener una visión de cómo va a quedar la evaluación, solo que sería bueno añadir la opción de editar una vez lanzada. Y algo que también añadiría es que, igual que los correos, se pueda guardar un borrador del control por si es que se realiza por partes o si es que se reinicia el *compu* y no se puede recuperar lo hecho, además serviría para, si es que hay dos secciones del curso, poder coordinar los controles sin tener que hacer dos veces el proceso de crear. Además, pondría algún botón para eliminar una evaluación creada por si hay recalendarización se acumulan temas y hay que modificarlos”.

Además, a último minuto se logró obtener comentarios de dos profesores más, distintos a los 4 anteriormente nombrados. Sin embargo, esta vez, los comentario fueron en su mayoría negativos. Ambos profesores encontraron que la interfaz de la plataforma es enredada y hay elementos que no cumplen una función clave, elementos que no se entienden y/o están sobrando. Algunos comentarios que se recibieron fueron: “Lamento no haber pasado de la primera etapa, pero la interfaz presenta demasiadas limitantes”, “La hora de inicio o de término presenta fallas. No puedo asignar una hora posterior a las 12:00”, “La preguntas de alternativas no permite agrupar respuesta”, “¿Qué significa el número donde dice # al inicio (cree nuevo curso)?”, “ Al crear control en fijar horario es incómodo, no corren las horas ni los minutos con el *mouse*, debo hacerlo con el cursor del teclado (flecha arriba/abajo)”, entre otros. Esto demuestra que no tuvieron una buena experiencia con la plataforma y no pudieron completar un flujo.

Teniendo en cuenta que un total de 5 usuarios es una cantidad muy baja para validar una plataforma, no se puede hacer un análisis de calidad. El motivo de esto será un punto a tratar en la conclusión (Capítulo 6).

5.2. Encuesta

Por otra parte, se realizó una encuesta¹ para medir la facilidad con la que se podían llevar a cabo las distintas acciones. Esta encuesta ha considerado las siguientes preguntas, las cuyas respuestas son notas del 1 al 7, en donde 1 tiene una connotación negativa, y 7 tiene una connotación positiva:

1. ¿Qué tan fácil le resultó crear un curso en la plataforma?
2. ¿Qué tan fácil le resultó crear un control en la plataforma?
3. ¿Qué tan fácil le resultó responder un control en la plataforma?
4. ¿Qué tan fácil le resultó evaluar un control en la plataforma?
5. ¿Cree que el uso de la plataforma es intuitivo?
6. En general, ¿Qué nota le pondría a la plataforma?

Esta encuesta fue publicada en U-Cursos. Sin embargo, solo fue respondida por 2 personas, y no es para nada representativa. Por esta razón, los resultados de esta encuesta se dejan en la sección de anexos (D).

5.3. Análisis de resultados

Teniendo en cuenta que la versión final que se logra de la plataforma tiene solo algunas funcionalidades más que el MVP que se validó, se puede suponer que la validación es efectiva para la plataforma que se tiene a la fecha, en el sentido de que lo que los usuario probaron no es muy distinto a lo que se entrega. No obstante, como se vio, la validación solo se pudo hacer con cinco personas, lo cual es bastante poco y no entrega la fiabilidad necesaria a la evaluación realizada.

De esta validación, se puede ver que existen opiniones encontradas, es decir, que se oponen. Algunos usuarios piensan que la plataforma es sencilla e intuitiva, mientras que otros no entendían nada de la aplicación, y no pudieron completar el flujo esperado.

Esto nos hace ver que la solución presenta problemas, ya que si hay un grupo de usuarios que no logran usar la plataforma, esta no está siendo útil y no cumple con el propósito.

¹<https://forms.gle/SbbVVGi6zDTiFMct7>

Capítulo 6

Conclusiones y trabajo futuro

En este capítulo se revisan los objetivos logrados (Sección 6.1), haciendo una autocrítica y dando a conocer ciertas reflexiones tras lo vivido en este proyecto (Sección 6.2). Además, se presentan perspectivas para la continuación de este proyecto (Sección 6.3).

6.1. Trabajo realizado

Tras un período de 16 semanas aproximadamente, se ha logrado desarrollar una plataforma incompleta, que si bien permite crear, responder y evaluar un control, deja bastante que desear. Si se mira hacia atrás, se puede decir que solamente lograron ciertos objetivos, es decir, se logró implementar las funcionalidades básicas para que se pudiera diseñar un control, responderlo y evaluar las respuestas. Es por esto que aún siguen habiendo muchos detalles a mejorar y cosas que no pudieron ser validadas tanto por falta de usuarios disponibles para hacer dichas validaciones como por negligencia en este desarrollo, por no buscar alternativas a tiempo.

Un objetivo importante era el de implementar una forma de agrupar las respuestas de los estudiantes en base a la similitud, usando algún algoritmo de *clustering* para texto, lo cual claramente no se llevó a cabo. También, dar *feedback* de forma grupal, para evitar que los evaluadores gasten demasiado tiempo en el proceso de evaluación.

Durante el trabajo, se invirtió mucho tiempo en ciertas funcionalidades que se subestimaron. Además, se gastó mucho tiempo en intentar hacer el *deployment* de la aplicación, ya que las configuraciones del servidor *Apache* dependían del equipo de sistemas del DCC y la comunicación era lenta, de hecho, hubo respuestas de parte de ellos que tardaron más de una semana, lo cuál complicó el proceso. En la sección de proceso del anexo (Sección B.2), se puede encontrar una pequeña bitácora en dónde se explica cómo se distribuyó el tiempo de desarrollo.

A pesar de lo anterior, se extrae todo como una experiencia única, y de aprendizaje. Es decir, de los errores se aprende, y todo, absolutamente todo, entrega alguna enseñanza, lo

que claramente ayudará en futuros trabajos.

6.2. Discusión y aprendizaje

Parto esta sección por un mea culpa en primera persona, por no pedir la ayuda suficiente para modelar la plataforma. Faltó seguramente trabajar con alguien experto en experiencia de usuario (user experience o UX). Las validaciones debieron hacerse con mayor antelación, para no tener el problema de saber que los usuarios tienen dificultades con la plataforma justo antes de la fecha de entrega de este trabajo. Además, no se probó la plataforma en distintos navegadores, ni sistemas operativos, algo que es clave hoy en día, y no se tuvo en consideración.

Todo esto se resume a que hubo dos grandes errores. El primero fue subestimar las tareas y por tanto dar menos plazo que el que de verdad iba a tomar, y el segundo fue no pedir ayuda temprana cuando estuve estancado sin poder avanzar. Además, no se definió el MVP al tiempo y luego hubo features que se quitaron para poder salir con un MVP a tiempo (antes de acabar el semestre). Esto, porque se quería hacer un MVP un poco ambicioso, y al final de cuentas, soportar imágenes no era necesario para el MVP, sino que podía ser perfectamente un *feature* que se agregara más adelante.

A pesar de no haber cumplido con todos los objetivos, creo que el avance ha sido notorio y significativo, y el aprendizaje ha sido enorme. Por esta razón me siento feliz. No hay que lamentarse por no haber podido desarrollar todo, simplemente hay que mirar hacia adelante y ver qué cosas quedan por hacer. El proceso de esta memoria en sí mismo es lo que importa, el trabajo de todo un semestre, es lo bonito de esta experiencia, y solo queda seguir mejorando.

6.3. Trabajo futuro

Durante el desarrollo de esta plataforma, fueron surgiendo muchas ideas de tanto pequeñas como grandes características que podrían aportar valor. Se sabe que aún queda mucho por hacer para que la plataforma sea realmente útil y buena. Además, quedan pendiente las validaciones de hipótesis que se mencionaron en un comienzo, y para eso, es indispensable que incorporen las características de poder ordenar las respuestas según similitud.

Teniendo en cuenta lo anterior, y con el fin de seguir mejorando *EasyWrittenExam*, se proponen a continuación las siguientes metas y tareas para un futuro trabajo:

1. Hacer refinamiento de los requisitos, en base a los comentarios que se obtuvieron en la última validación
2. Añadir más funcionalidades claves para un profesor, como editar y borrar un control ya creado
3. Incorporar un modelo entrado para hacer *clustering* sobre texto, basado en la similitud semántica de las respuestas

4. Permitir responder a las preguntas de desarrollo cargando un archivo, imagen o documento en formato *pdf*.
5. Hacer pruebas de la plataforma con un curso real.

Para hacer el refinamiento, se debe crear un documento con todos los comentarios que se obtuvieron de la plataforma. Así, se pueden levantar nuevas tareas para resolver estas falencias. Un refinamiento como este demoraría alrededor de una semana de trabajo.

Añadir más funcionalidades para un profesor es una tarea un poco ambigua, se debe hacer un *sprint planning* (planear) sobre cómo se va a avanzar, las tareas que conlleva la implementación de estas funciones. Editar un control puede volver a la vista de TestDesign, o bien llevar a una nueva vista, levantar un modal, etc. Son decisiones que se pueden tomar y llevar a cabo en implementación en unas dos o tres semanas.

Incorporar un modelo entrenado para hacer *clustering* es una sugerencia como trabajo futuro. En este punto, se debe estudiar mejor la propuesta. Incluso, se puede tratar como otro trabajo de memoria a parte. Un trabajo de memoria de este estilo contemplaría la investigación sobre los distintos modelos, implementación y validaciones con evaluadores reales. Esto sería trabajo para algunos meses.

Permitir responder preguntas subiendo archivos es una funcionalidad que se puede implementar sin mucho esfuerzo. Sin embargo, es trabajo un poco extenso, hay que verificar si los modelos requieren alguna modificación. En este caso, se podría lograr la implementación con un mes de trabajo, lo cual significa que este trabajo se podría incluir como parte de la continuación y mejora de EasyWrittenExam, para otro memorista. Este memorista tendría también la labor de finalizar las pruebas con usuarios reales.

Bibliografía

- [1] *6 Exam Platforms - Himanshu Singh*. <https://medium.com/@MarktrHimanshu/6-best-open-source-exam-software-and-assessment-platforms-37a667675edb>. Visitado el 3 de mayo de 2021.
- [2] *Acceso a internet en América Latina 2020*. <https://www.iadb.org/es/noticias/al-menos-77-millones-de-personas-sin-acceso-internet-de-calidad-en-areas-rurales#:~:text=En%20total%2C%20un%2032%25%20de,diferencia%20de%2040%20puntos%20porcentuales>. Visitado el 4 de mayo de 2021.
- [3] *Apache*. <https://www.apache.org/>. Visitado el 5 de diciembre de 2021.
- [4] *Bootstrap*. <https://getbootstrap.com/>. Visitado el 4 de diciembre de 2021.
- [5] *Edbase*. <https://www.edbase.net/>. Visitado el 3 de mayo de 2021.
- [6] *Exam Platforms - SoftwareSuggest*. <https://www.softwaresuggest.com/blog/best-open-source-exam-software-assessment-platforms/>. Visitado el 3 de mayo de 2021.
- [7] *Express*. <https://expressjs.com/>. Visitado el 5 de diciembre de 2021.
- [8] *FlexiQuiz*. <https://www.flexiquiz.com/>. Visitado el 3 de mayo de 2021.
- [9] *Git - Scm*. <https://git-scm.com/>. Visitado el 3 de diciembre de 2021.
- [10] *Gitlab - About*. <https://about.gitlab.com/what-is-gitlab/>. Visitado el 3 de diciembre de 2021.
- [11] *Internationalization - i18n*. <https://developer.mozilla.org/es/docs/Mozilla/Add-ons/WebExtensions/API/i18n>. Visitado el 5 de diciembre de 2021.
- [12] *Jwt - Introducción*. <https://jwt.io/introduction>. Visitado el 4 de diciembre de 2021.
- [13] *PaperShala*. <https://papershala.com/>. Visitado el 3 de mayo de 2021.
- [14] *Postgresql*. <https://www.postgresql.org/>. Visitado el 5 de diciembre de 2021.

- [15] *Proprofs Quiz Maker*. <https://www.proprofs.com/quiz-school/>. Visitado el 3 de mayo de 2021.
- [16] *Questbase*. <https://questbase.com/>. Visitado el 3 de mayo de 2021.
- [17] *Quizalize*. <https://www.quizalize.com/>. Visitado el 3 de mayo de 2021.
- [18] *Qué es una API - RedHat*. <https://www.redhat.com/es/topics/api/what-are-application-programming-interfaces>. Visitado el 5 de diciembre de 2021.
- [19] *React*. <https://reactjs.org/>. Visitado el 4 de diciembre de 2021.
- [20] *Redux*. <https://redux.js.org/>. Visitado el 4 de diciembre de 2021.
- [21] *Testmoz*. <https://testmoz.com/>. Visitado el 3 de mayo de 2021.
- [22] *Tmux - Wikipedia*. <https://es.wikipedia.org/wiki/Tmux>. Visitado el 5 de diciembre de 2021.
- [23] *Trello - About*. <https://trello.com/about>. Visitado el 3 de diciembre de 2021.
- [24] Basil, Victor R y Albert J Turner: *Iterative enhancement: A practical technique for software development*. IEEE Transactions on Software Engineering, (4):390–396, 1975.
- [25] Basilaia, Giorgi y David Kvavadze: *Transition to online education in schools during a SARS-CoV-2 coronavirus (COVID-19) pandemic in Georgia*. Pedagogical Research, 5(4), 2020.
- [26] Chen, Lijia, Pingping Chen y Zhijian Lin: *Artificial intelligence in education: a review*. Ieee Access, 8:75264–75278, 2020.
- [27] Chen, Tinggui, Lijuan Peng, Xiaohua Yin, Jingtao Rong, Jianjun Yang y Guodong Cong: *Analysis of user satisfaction with online education platforms in China during the COVID-19 pandemic*. En *Healthcare*, volumen 8, página 200. Multidisciplinary Digital Publishing Institute, 2020.
- [28] Christensen, Rhonda: *Effects of Technology Integration Education on the Attitudes of Teachers and Students*. Journal of Research on Technology in Education, 34(4):411–433, 2002. <https://doi.org/10.1080/15391523.2002.10782359>.
- [29] García-Alberti, Marcos, Fernando Suárez, Isabel Chiyón y Juan Carlos Mosquera Feijoo: *Challenges and Experiences of Online Evaluation in Courses of Civil Engineering during the Lockdown Learning Due to the COVID-19 Pandemic*. Education Sciences, 11(2):59, 2021.
- [30] Hazas, Mike, James Scott y John Krumm: *Location-aware computing comes of age*. Computer, 37(2):95–97, 2004.
- [31] Jatana, Nishtha, Sahil Puri, Mehak Ahuja, Ishita Kathuria y Dishant Gosain: *A survey and comparison of relational and non-relational database*. International Journal of

Engineering Research & Technology, 1(6):1–5, 2012.

Anexos

Anexo A

Entrevistas preliminares

Para llevar a cabo este trabajo se propuso partir con una fase de identificación de las maneras de facilitar el proceso de las evaluaciones en línea. Para esto se creó una encuesta a través de *Google Forms* con el fin de ver el punto de vista de los potenciales usuarios que son quienes se ven enfrentados a los sistemas de evaluaciones. Esta encuesta fue publicada en la plataforma U-Cursos y se usó para complementar los requisitos de la aplicación. Además, se logró entrevistar a un par de personas, también con la misma finalidad.

A.1. Análisis encuesta

Se creó una encuesta a través de *Google Forms*, la cual se encuentra disponible en <https://forms.gle/scGRFgbXUgv3MKuU6> y se publicó a través de U-Cursos.

Tras unas semanas, la encuesta fue respondida por 88 personas. Algunas de las características más solicitadas fueron la de que una evaluación se pueda responder en cualquier momento dentro de un período de tiempo (92% de las respuestas). Esto daría más flexibilidad a los estudiantes y ya que les permite organizar sus demás tareas.

Otra característica deseada es la de poder subir imágenes como respuestas (78.4% de las respuestas). Es decir, subir un *scanner* de una hoja con la respuesta. Esto, debido a que en ciertas materias es más rápido responder con papel y lápiz que con el teclado en un computador.

Ahora, en lo que respecta al diseño de las evaluaciones, algunas respuestas hacían alusión a la comodidad que entrega el papel y lápiz convencional. Por esto, el input para las respuestas fue con un `textarea` que soporta `LaTeX`, aunque también se da la posibilidad de simplemente subir una imagen.

A.2. Entrevistas

Las personas entrevistadas fueron las siguientes:

- Guillermo Fuentes: Profesor de la Facultad de Ciencias Agronómicas de la Universidad de Chile. Guillermo aseguraba usar la plataforma Testmoz para sus evaluaciones. Se ofreció como voluntario para probar EWE.
- Benjamín Vera: Estudiante de ingeniería civil matemática en la Facultad de Ciencias Físicas y Matemáticas de la Universidad de Chile. Ayudante en cursos del DIM. Mencionó que en los cursos en que corregía se usaba la plataforma Reclamos DIM para las evaluaciones.
- Willy Maikowski: Ingeniero Civil en Computación y parte del equipo de U-Cursos. Demostró disposición a ayudar con la integración de EWE a U-Curso por medio del desarrollo de alguna API y/o implementación de LTI.

Anexo B

Proceso

A continuación se explica cómo se llevó a cabo este proyecto, se muestra el calendario ideal de avance (Apéndice B.1) y una pequeña “bitácora” (Apéndice B.2). Este plan de trabajo contempló tanto parte del período de la propuesta de memoria, como el período actual de trabajo.

B.1. Calendario de Trabajo

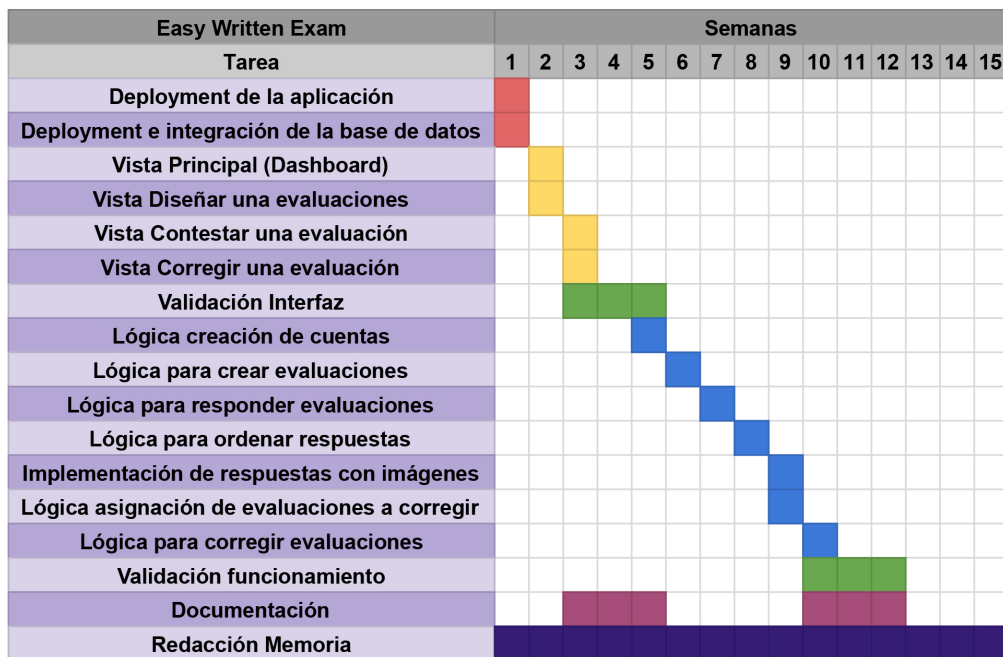


Figura B.1: Calendario de Trabajo - Carta Gantt

B.2. Desarrollo

Antes de comenzar el semestre, se trabajó en el diseño del modelo de la base de datos. Para esto se usó un modelo de clases, pensando en cuáles serían las clases que interactuarían en nuestra aplicación. Cabe destacar que no se hizo ningún tipo de estudio, simplemente se tomó como decisión la mejor alternativa que se pensó, creyendo que sería un preliminar a su vez. Luego, se creó un *script* en formato *.sql* encargado de crear nuestra base de datos.

Para unir la base de datos con el *back-end*, se siguió una guía de *bezcoder*¹. Con esto, se logró, además, implementar el sistema de autenticación con *Jwt*. Todo esto correspondió a un trabajo previo.

Al comenzar la semana 1, se habló con la gente de sistemas del DCC, para coordinar bien el *deployment*, ya que depende de ellos el manejar el servidor de *buhó*. De esta tarea surgieron algunos inconvenientes, que junto con el tiempo que toma la comunicación vía correo electrónico, dieron como resultados un retraso en esta parte.

Los inconvenientes mencionados se refieren, principalmente, a la inexperiencia de trabajar con *Apache* y *React* al mismo tiempo. Se probaron muchas cosas para poder levantar bien la aplicación, pero siempre se encontraban pequeñas trabas, sobre todo con el tema de las rutas ya que *React* trabaja con *client-side rendering* mientras que *Apache* sirve archivos estáticos. Por lo que las rutas que *React* sirve con *JavaScript* no se encontraban disponibles al poner la *url* directamente en el navegador.

Para no perder mucho tiempo, y dado que las respuestas por parte de sistemas podían tardar hasta días, se decidió continuar con las tareas pendientes de las siguientes semanas.

Estando ya casi en semana 3, solo se había logrado avanzar en la vista de *Home* y parte de *TestDesign*. La nula experiencia con *React* y los *frameworks* de *front-end* complicaba mucho poder lograr lo que se tenía en mente. Para simplificar un poco el desarrollo, se decidió dejar de trabajar con *material-ui* y *tailwindcss* para empezar con *bootstrap*, dado que aparentemente es más sencillo y ya se había trabajado antes con este *framework* en cursos de la universidad.

Ya finalizando la semana 4, solo se había logrado hacer las vistas de *Home*, *TestDesign* en un 90%, *TestEvaluation* en un 50% y *TestAnswering* en un 50% también. Si bien había un retraso en el desarrollo con respecto a lo programado en el calendario, se logró arreglar el tema de las rutas en *buhó*.

En la semana de receso se avanzó un poco más sobre la vista *TestDesign*, logrando que las *previews* para las preguntas de alternativas mostraran las alternativas debajo del enunciado. Se cambió un poco el estilo de la *Navbar* (nuevos colores). También, se trabajó en la incorporación de nuevos idiomas con *i18n*, se actualizaron los idiomas para las vistas de *Home*, *TestDesign*, *TestAnswering* y *TestEvaluation*. Por último, se agregó la vista de *SignIn* sin que esta funcione realmente.

En paralelo, se comenzó a trabajar en un *refactoring* para la vista de *TestDesign*, ya que

¹Tutorial de Bezcoder: <https://www.bezkoder.com/node-js-jwt-authentication-postgresql/>

este componente había crecido mucho y no se estaba sacando el máximo provecho a *React*, cuyo principio es el de dividir nuestro código en componentes reutilizables. Este *refactoring* trajo consigo un problema y era el de almacenar el estado para todos los componentes. Para esto, se comenzó a trabajar con *Redux* y otras alternativas.

Semana 5 se trabajó en la autenticación, se desarrolló un sistema basado en *Redux* siguiendo una guía. Por supuesto, se adaptaron un poco ciertas cosas, como los mensajes para que pudieran ser traducidos. Lo malo fue que al momento de *deployar*, hubo un problema con el tema de conectar el *front* con el *back*.

Para solucionar el problema anterior, se mandó un *mail* a sistemas, y se estuvo a la espera de la respuesta por algún tiempo. Mientras tanto, se avanza en la creación de exámenes. Al final, con Hernán (persona a cargo de sistemas del DCC) solucionamos el problema y las solicitudes comenzaron a funcionar correctamente a la *API*. El único problema fue que las *requests* llegaban independientemente de la dirección IP, por lo que había que colocar una capa de seguridad en el *back* para que las rutas te exigieran un *token* y lo validaran.

Dado que esto estaba funcionando de alguna forma, se dio prioridad al desarrollo de la aplicación. De hecho, todo esto de *Apache* me tomó mucho tiempo poder dejarlo funcionando, ya que habían muchos problemas con la configuración de *Apache* y *React* y todo. Entonces, una vez me funcionó, ya no se quiso hacer más cambios para no echar a perder nuevamente el funcionamiento. Por lo que no se ha pulido este proceso de *deployment* ni tampoco se está 100 % seguro de que la configuración es la correcta para un *deployment*. La configuración que se tiene es la que pude averiguar en internet para hacer *deployment* y la que me funcionó a mí.

Por otro lado, durante esta misma semana, se mandó correos a los profesores que se comprometieron para ayudar en el *feedback*. Sin embargo, solo se recibió la respuesta de una profesora, la profesora Tonka, quién dio su opinión sobre las vistas. Esto fue bastante útil para saber si las vistas cumplían o no su opinión.

Durante las siguientes semanas se trabajó en lograr que se crearan los *tests*, aunque para esto, se debió colocar una capa extra en donde un usuario puede crear un curso y también puede unirse a un curso. La lógica de estas capas aún son preliminares, ya que no se pensaron demasiado tiempo y fueron, más bien, una medida rápida para solventar las necesidades de la lógica de creación de *tests*.

Es semana 9 y recién se está trabajando en la lógica de la vista para responder una evaluación. Ha habido un notorio retraso. Además, aún no se recibe respuesta de los profesores a los cuáles se les envió el *email* pidiendo *feedback*. Esta es una mala noticia, ya que hasta ahora solamente se tiene la opinión de una sola profesora, lo cual es muy poco.

Semana 10 y mitad de la semana 11 se ha continuado la vista para responder una evaluación. Lo más complicado ha sido tener que hacer que todo sea dinámico, y se pueda navegar entre las preguntas sin perder información, además de que al enviar una respuesta, esta tiene que estar asociada a la pregunta correcta. Para esto, se utilizó un objeto en *localStorage*, el cual guarda la información en caso de que se recargue la página.

Manejar el tiempo restante igual ha sido un poco complicado. Se ha vuelto a reflexionar sobre cómo definir el MVP, ya que hay cosas que realmente no son necesarias para un MVP y se les estaba dando mucha importancia, lo que causó un retraso. Por esta razón, se decidió simplificar algunas opciones y detalles de la vista que estaba pensada en primera instancia. Así, se pasó más rápido al desarrollo de la vista de evaluaciones.

En el resto de la semana 11 y de la semana 12 se comenzó a utilizar *Trello* para ir definiendo las tareas e ir dividiéndolas en tareas más simples. Lo cual ha impactado positivamente en la productividad. En esta semana y media se logró casi por completo el desarrollo de la vista de evaluaciones. Se había decidido pasar directamente al desarrollo de esta vista debido al retraso, con el objetivo de terminar el MVP.

Las tareas de ordenar respuestas, implementar respuestas con imágenes y asignación de controles a corregir se tuvieron que posponer para poder avanzar con otras cosas que son más importantes para un MVP, como el de tener una vista en donde el usuario pueda ver su nota y la corrección de su prueba. Se ha decidido que desarrollar estas vistas básicas, que abarquen todas las facetas de la plataforma, es más importante para que la plataforma se encuentre operando que desarrollar características más elaboradas.

Durante la semana 13 y 14 se trabajó en la vista de evaluaciones. Además, para hacer el MVP un poco más completo, se creó una vista para que los alumnos ingresen a un control respondido y revisen sus respuestas y la nota que se les puso en cada pregunta, más el comentario en caso de haber. Además, se creó una vista simple para el curso, en donde el profesor puede agregar nuevos integrantes al curso con distintos roles, para así tener una forma de agregar evaluadores a los cursos.

Además, se les envió *emails* a los profesores que se habían ofrecido para probar la plataforma, pero de 4 de ellos, sólo respondieron 2 y una sola profesora mandó comentarios.

Durante la semana 15 se avanzó en el informe, asumiendo que ya no se había logrado todo lo que se propuso en un comienzo, pues se había subestimado todo el trabajo que era desarrollar la plataforma.

Anexo C

Otras Vistas

A continuación se muestran otras vistas principales que no se vieron en el reporte.

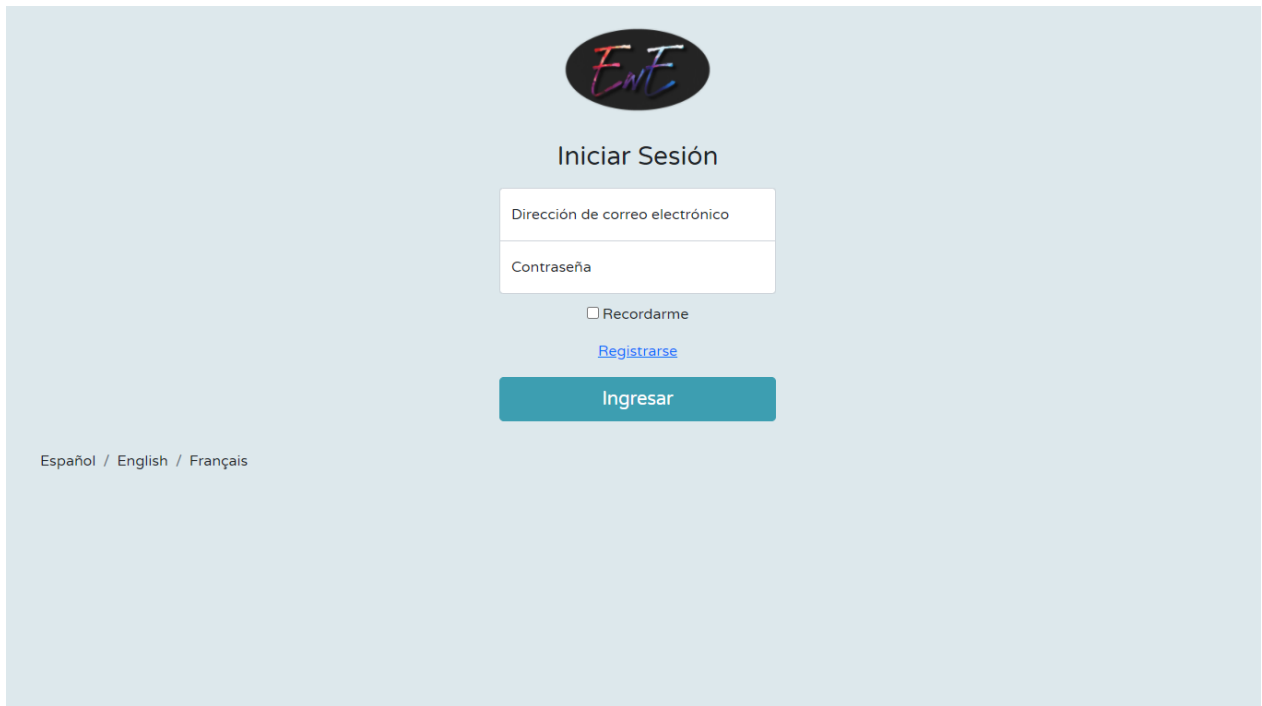


Figura C.1: Login de la plataforma. Se puede apreciar que las opciones por los idiomas se encuentran por debajo a un costado izquierdo.

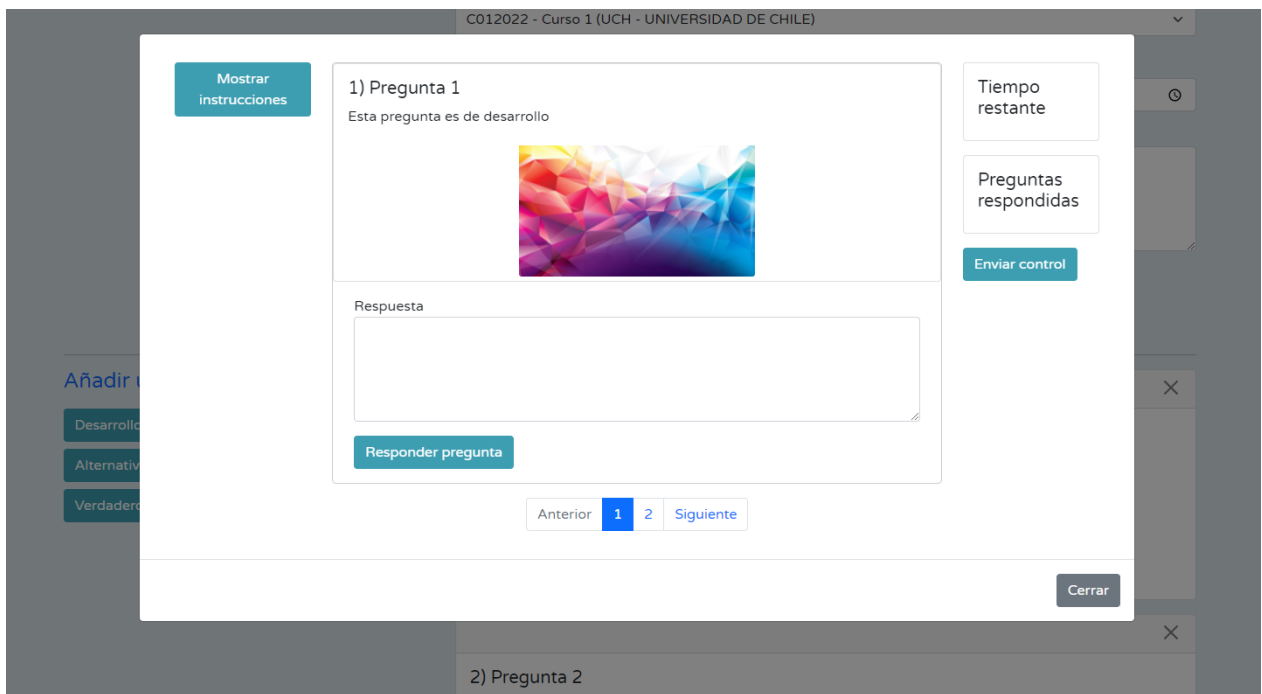


Figura C.2: Previsualización de un control. Cuando se presiona **Previsualizar** en la vista *TestDesign* se levanta un modal que tiene la vista que se vería al momento de responder el control. En esta vista no se ve el tiempo ni las preguntas en el costado derecho, pero se muestra la ubicación que tendrían dentro de la vista.

Anexo D

Resultados de la encuesta de usabilidad EWE

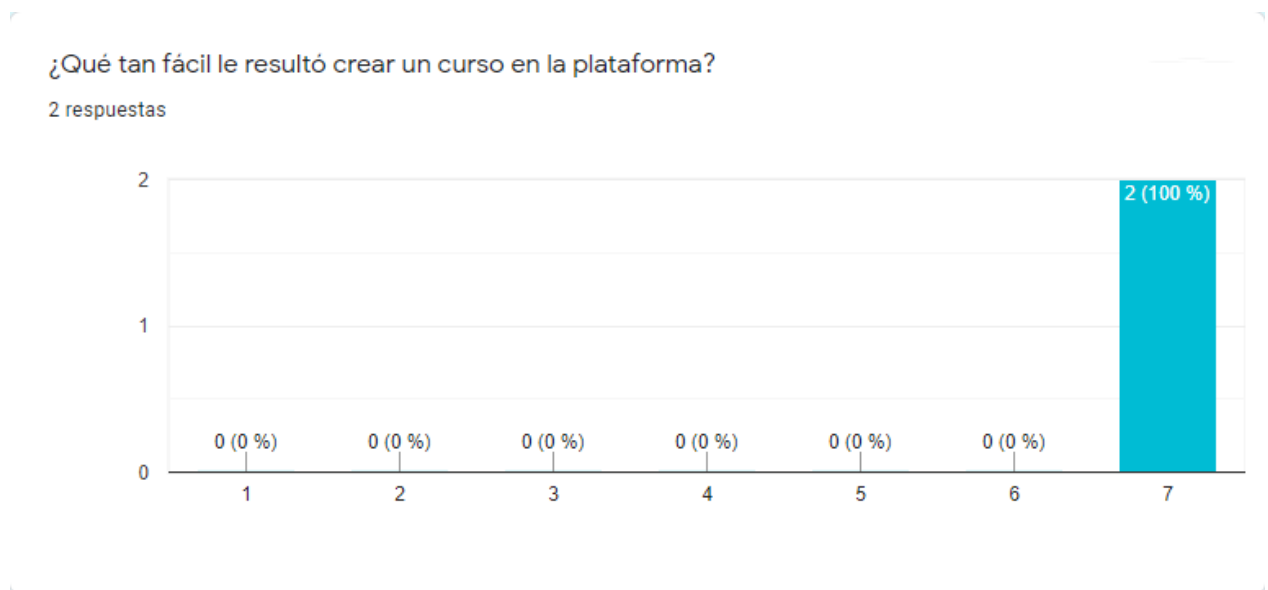


Figura D.1: Resultados pregunta 1 de encuesta

¿Qué tan fácil le resultó crear un control en la plataforma?

2 respuestas

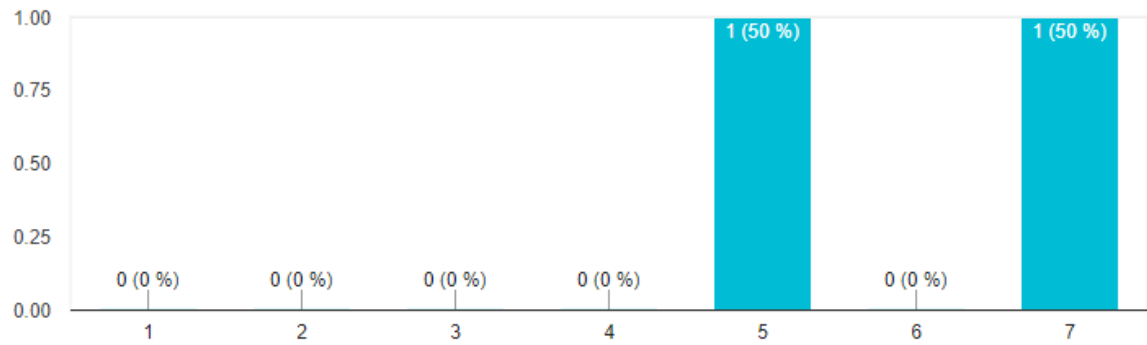


Figura D.2: Resultados pregunta 2 de encuesta

¿Qué tan fácil le resultó responder un control en la plataforma?

2 respuestas

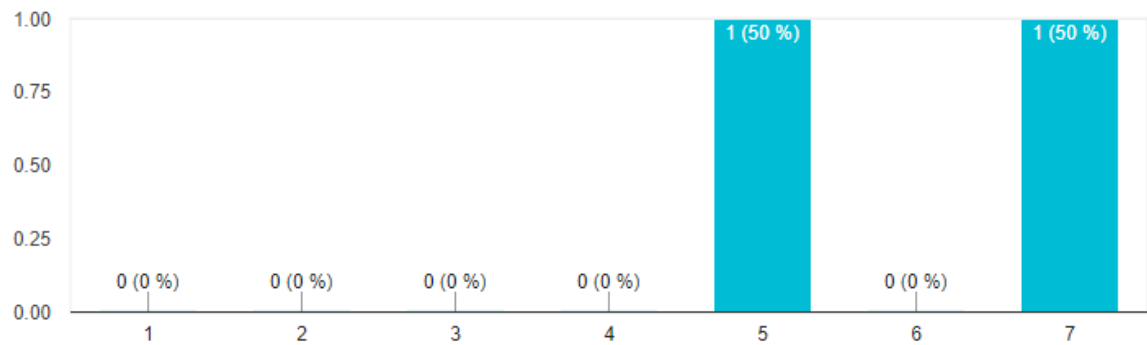


Figura D.3: Resultados pregunta 3 de encuesta

¿Qué tan fácil le resultó evaluar un control en la plataforma?

2 respuestas

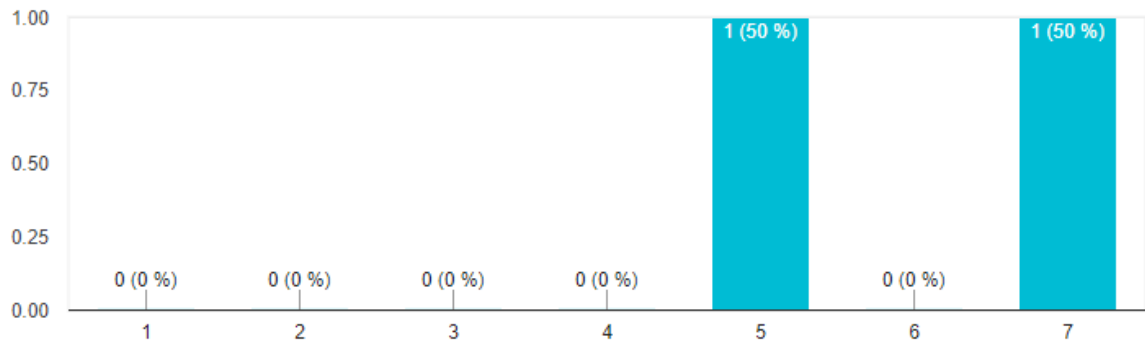


Figura D.4: Resultados pregunta 4 de encuesta

¿Cree que el uso de la plataforma es intuitivo?

2 respuestas

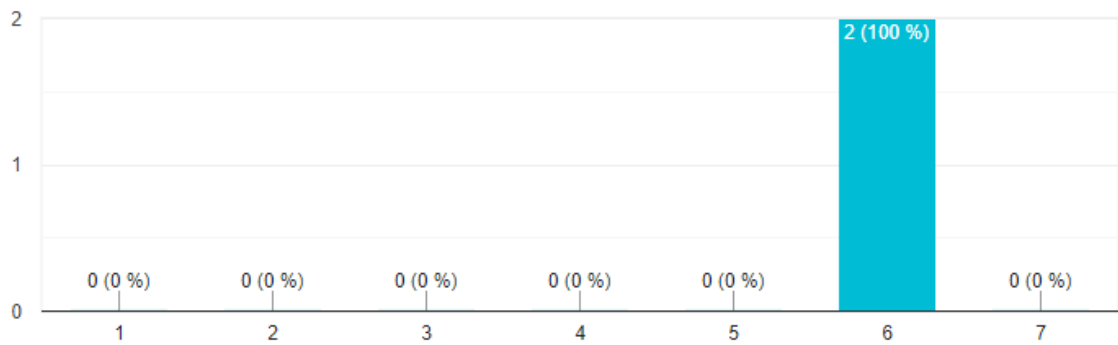


Figura D.5: Resultados pregunta 5 de encuesta

En general, ¿Qué nota le pondría a la plataforma?

2 respuestas

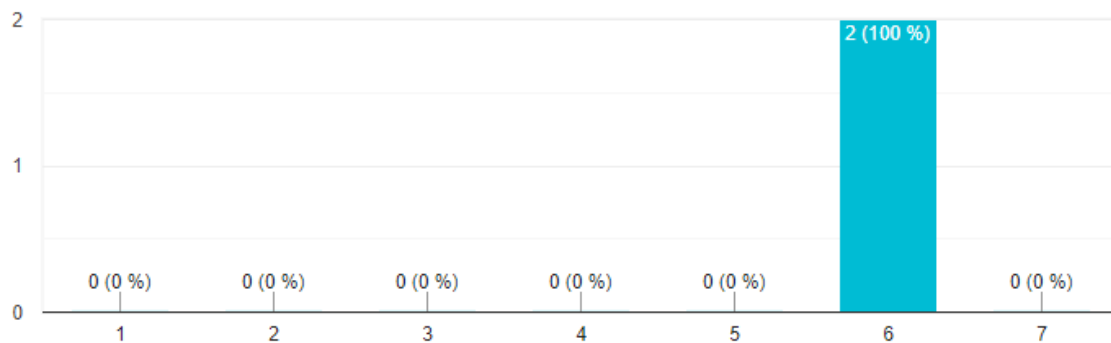


Figura D.6: Resultados pregunta 6 de encuesta