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

EASY VIDEO COMPETITION: APRENDER ENSEÑANDO

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN COMPUTACIÓN

GUSTAVO ENRIQUE SILVA VELIZ

PROFESOR GUÍA:
JÉRÉMY BARBAY

MIEMBROS DE LA COMISIÓN:
BENJAMÍN BUSTOS CÁRDENAS
EDUARDO GODOY VEGA

SANTIAGO DE CHILE
2023

RESUMEN DE LA MEMORIA
PARA OPTAR AL TÍTULO DE:
Ingeniero Civil en Computación
POR: Gustavo Enrique Silva Veliz
AÑO: 2022
PROFESOR GUÍA: Jérémy Barbay

EASY VIDEO COMPETITION: APRENDER ENSEÑANDO

Es importante que las personas conozcan las normativas que rigen el país, ya que definen un correcto comportamiento cívico. El Ministerio de Transportes tiene un problema de comunicación, las leyes y regulaciones van cambiando, pero la ciudadanía no se "pone al día".

Este problema de comunicación se podría mejorar logrando que los propios niños y jóvenes, mediante la realización y evaluación de vídeos pedagógicos cortos, puedan comunicar y educar al resto de la población y sus propias familias.

Se propuso extender el software existente *Vídeoclases* que permite a alumnos publicar y evaluar vídeos de sus pares en una plataforma que facilite a los ministerios, por ejemplo, el de transportes, de manera que las instituciones solo tengan que enfocarse en elegir un tema para la actividad al inicio del periodo y la publicación del mismo.

Se validó el uso de la plataforma en varias iteraciones del desarrollo. Finalmente, solo se lograron dos validaciones. Una en conjunto con el Ministerio de Transportes, en donde se mostró y reviso completamente la plataforma a funcionarios públicos. Para la otra se realizó un formulario de usabilidad, el cual fue respondido por profesores, auxiliares y ayudantes de la facultad y otros colegios. Este último formulario también ayudó como testing para la plataforma, dado que se usó en un caso más cercano a la realidad y permitió hacer una revisión por parte del alumno del trabajo realizado. Como trabajo futuro se plantea la idea de hacer una validación científica de la calidad de la evaluación colaborativa de los vídeos entregados.

A mi familia que nunca dejo de confiar en mi.
A mis amigos, soporte fundamental en mi vida.

Tabla de Contenido

Capítulo 1

Introducción

Nuestra sociedad humana está en constante evolución y un tema que sigue siendo de suma importancia es la comunicación efectiva por parte de las autoridades hacia la población. En particular, el Ministerio de Transportes del Gobierno de Chile está muy interesado en la búsqueda de nuevas formas para poder informar a la sociedad, aprovechando las nuevas tecnologías y el acceso a internet.

Muchos jóvenes y niños no saben sobre propuestas importantes de distintas áreas, provocando que las autoridades gubernamentales se estén quedando atrás a la hora de educar e informar a la población juvenil. Para dar un ejemplo real, la CONASET (Comisión Nacional de Seguridad de Tránsito) informa que la mayor parte de los accidentes de tránsito (peatones, ciclistas, conductores, etc.) afectan a los adolescentes y adultos jóvenes. De los 64.707 siniestros registrados, en 32.338 participaron jóvenes de entre 15 y 29 años, lo que corresponde a un 49,97 % del total. [8]

Según un estudio del Instituto Nacional de la Juventud (Injuv) [11], un gran porcentaje de los jóvenes y niños del país valoran mucho más ser educados por sus propios pares que por las autoridades, algo que se puede aprovechar para generar mayor efectividad al informar. Los mismos jóvenes serán capaces de enseñarles a sus familias sobre normativas o medidas que ni los mismos adultos tenían en cuenta. Por ejemplo, poca gente sabe qué niños de hasta 5 años pueden andar en bicicleta en las veredas, siempre acompañados de un adulto responsable. Esto puede llegar a abarcar no solo temas gubernamentales, sino también ser una herramienta de apoyo a la enseñanza en colegios y distintas instituciones en donde los mismos alumnos se evalúen y aprendan entre ellos, logrando un mayor impacto en su educación.

El proyecto "Teaching is Learning", iniciado por Jérémy Barbay [5] en 2009, se basa en la hipótesis de que el uso de las nuevas tecnologías mejora la metodología "Aprendizaje a través de la enseñanza" planteada por Frager y Stern [10]. Esta metodología tiene como base en que los mismos alumnos enseñan a otros mientras aprenden, generando un beneficio mutuo para ambas partes. El proyecto considera la utilización de técnicas de evaluación colectivas para aplicar la metodología propuesta por Frager y Stern, usando las evaluaciones que realicen los mismos alumnos. Teaching is Learning es un proyecto que también ha sido utilizado en los cursos dictados por el profesor Barbay en la facultad mediante la creación de vídeos cortos elaborados por los estudiantes, los cuales son evaluados por sus pares.

Así es como surge la idea de extender una plataforma creada para el proyecto Teaching is Learning llamada "Videoclases"[1]. El software, que fue desarrollado en 2015 por Diego Muñoz y luego actualizado en 2017 por Jorge Ampuero, permite aplicar la hipótesis de realización y validación de vídeos pedagógicos cortos. La idea detrás de la remodelación de la herramienta es permitir al Ministerio de Transportes iniciar anualmente concursos de vídeos pedagógicos sobre alguna temática en específico, y que los alumnos, con la ayuda de sus profesores, puedan aprender y evaluar a sus pares con la motivación de un premio por parte de los organizadores.

Objetivos

Objetivo General

El Objetivo General de este trabajo de título es extender y actualizar una plataforma web que permita a alumnos subir vídeos pedagógicos cortos y que puedan ser evaluados por los mismos participantes y profesores, intentando hacer un piloto (concurso más pequeño que los planificados a largo plazo) a fin de año en conjunto al Ministerio de Transportes.

Objetivos Específicos

1. Actualizar las versiones tanto del framework como del motor de la base de datos.
2. Validar el modelo de datos.
3. Realizar una actividad con el Ministerio de Transportes usando la plataforma Videoclases.
4. Tener un tipo de usuario específico para las autoridades de las instituciones.
5. Implementar un sistema de evaluación siguiendo las directrices del Ministerio.
6. Crear plantillas para distintos sistemas de evaluación.
7. Crear una vista de administración para autoridades (tercer tipo de usuario) y modificar la vista ya existente para los profesores.

En el siguiente Capítulo se explicitará el Estado del Arte de la plataforma, explicando la problemática y el interés que mostró el Ministerio de Transportes.

Capítulo 2

Estado del Arte

En la actualidad, existe un proyecto llamado **Videoclases**, cuyo código es abierto y disponible en GitHub [2], en donde se tiene el software completo para lanzar una aplicación web basada en Django y SQLite que permite la subida y posterior revisión de vídeos pedagógicos por los alumnos y profesores. Este software fue desarrollado el año 2015 por el alumno Diego Muñoz y luego actualizado por el alumno de magíster Jorge Ampuero en 2017, junto con su profesor guía Jérémy Barbay. La plataforma es simple y fácil de usar, pero en el apartado visual y temático le falta desarrollo, por lo que es posible trabajar sobre el código, extendiéndolo y agregando los componentes necesarios para lograr el objetivo final. En las siguientes secciones se explicarán detalles específicos de la plataforma Videoclases:

- Una descripción de la aplicación web existente mostrando imágenes y explicando las funcionalidades actuales más importantes. (Sección 2.1).
- Se comentará sobre las problemáticas que tiene la plataforma dado que sigue siendo trabajada y utilizada como un prototipo. (Sección 2.2).
- El interés que planteo el Ministerio de Transportes en el proyecto y la posibilidad de extenderlo a otras instituciones. (Sección 2.3).

2.1. La Plataforma

Durante el año 2015, esta plataforma [1], junto con la metodología **Teaching is Learning**, fueron probadas usando un piloto en alumnos de 12 años del colegio Blest Gana. Estos últimos realizaron 3 vídeos pedagógicos por semestre con la ayuda de sus celulares [3]. En el año 2016, se llevó a cabo un concurso con 3 colegios de Santiago utilizando la plataforma VideoClases en donde se seleccionaron los tres mejores vídeos de cada curso por sus pares y se enviaron al comité organizador del proyecto para luego realizar una invitación masiva a la Facultad de Ciencias Físicas y Matemáticas [4]. El proyecto tuvo éxito y llevo a que la plataforma también fuera usada en algunos cursos del profesor Jérémy Barbay dictados en el departamento de Ciencias de la Computación hasta el día de hoy para hacer exámenes orales asíncronos.

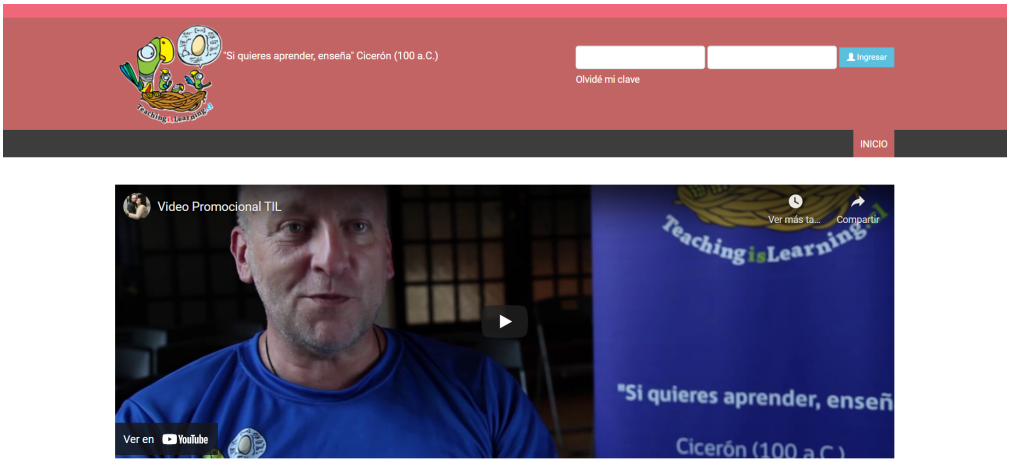

Figura 2.1: Vista inicial de la plataforma a la que llegan todos los usuarios.

El software actual presenta una vista inicial en donde se permite el ingreso a la plataforma tanto para alumnos como profesores. Luego de hacer ingreso, como se ve en la Figura 2.2, si el usuario es tipo alumno, se muestra una página con la información propia del estudiante, el curso correspondiente y una tabla con las tareas pendientes por hacer. El alumno, si es que es parte de algún curso y existe una tarea pendiente, puede subir el link de su vídeo a la plataforma junto con unas preguntas escritas en un formulario para que sean respondidas por sus demás compañeros basándose en la cápsula pedagógica entregada.

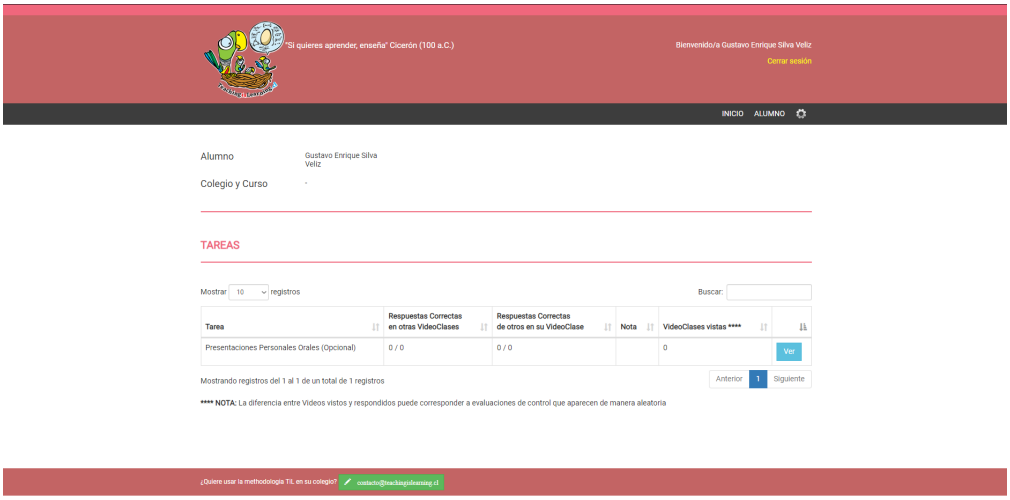

Figura 2.2: Vista de usuario tipo Alumno. Aparecen las tareas realizadas y pendientes en formato de tabla.

Si se ingresa a la plataforma como usuario tipo profesor, la vista presenta información sobre el usuario, colegio (Facultad, Escuela, etc.) y una tabla con los cursos actuales asociados a la cuenta, además del profesor encargado, el número de alumnos y la tarea en curso. Esto se puede ver en la Figura 2.3. Aparte de las vistas mencionadas, se tienen las necesarias para crear cursos y tareas. Dado que el software está basado en Django, también se puede utilizar la interfaz de administración del framework para labores más específicas.

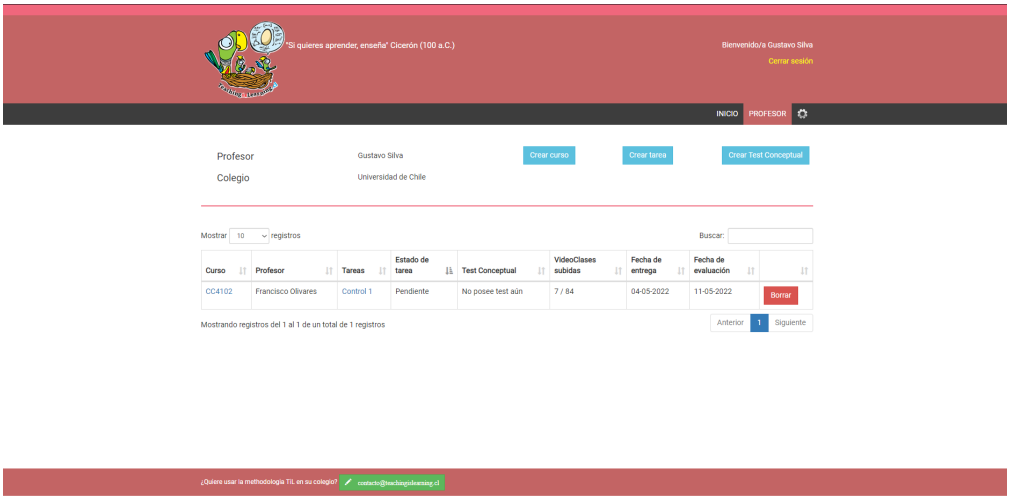

Figura 2.3: Vista de usuario tipo Profesor. Se puede apreciar una tabla con la información correspondiente a todas las actividades asociadas al usuario profesor.

Si el usuario tipo profesor presiona el botón para crear un curso, se le mostrará una vista de creación como se ve en le Figura 2.4. Aquí el usuario debe descargar una planilla a modo de template para posteriormente llenarla con la información requerida por el sistema y subirla a la plataforma. Con esto hecho, el curso generado aparecerá en la tabla de la página anterior.

Para crear una tarea, el profesor debe llenar los parámetros obligatorios:

1. Título de la tarea
2. Descripción de la actividad
3. El curso asignado
4. La cantidad de revisiones mínimas por alumno. Recordemos que la plataforma está basada en que los alumnos deben revisar las entregas de sus compañeros.
5. Fecha límite para subir la tarea y fecha límite de evaluación de estas últimas.

Luego se presenta una sección para agregar los Criterios de Evaluación, en donde se debe escoger tanto el tipo de escala a utilizar como los criterios específicos a evaluar, además de una descripción opcional si se estima necesario. Se puede agregar la cantidad de criterios que uno desee. Los tipos de escala presentes actualmente son:

- 3 Criterios: Cumple, Cumple parcialmente y No cumple
- 5 Criterios: Cumple, Cumple en su mayoría, Cumple parcialmente, Cumple muy parcialmente y No cumple
- Escala de 0 - 7
- Escala de 1 - 7

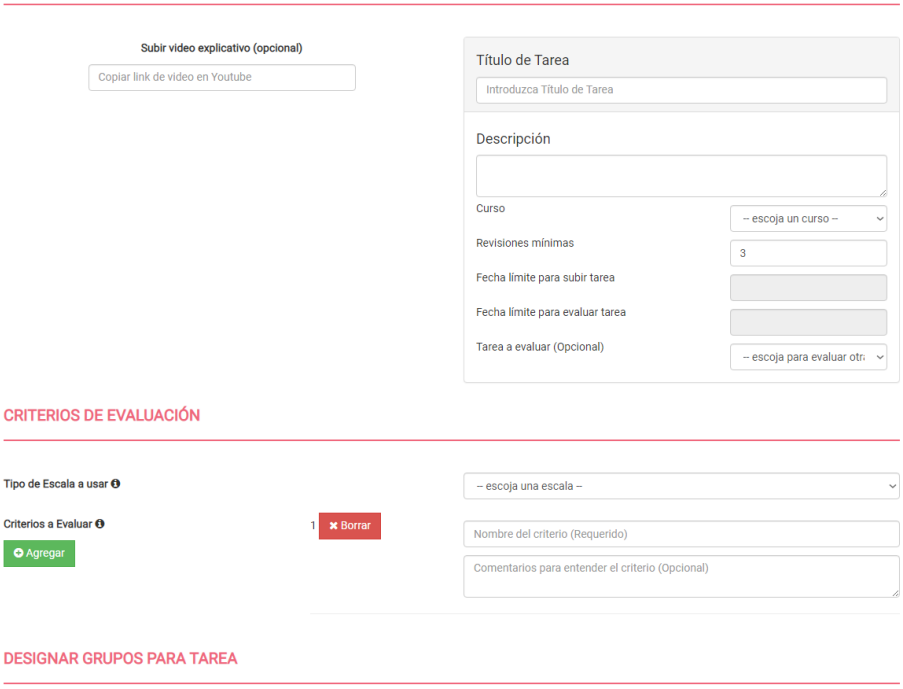

Figura 2.4: Vista de creación de tareas. Presenta un formulario para los detalles y los criterios de evaluación.

Finalmente, se pueden designar grupos a los alumnos para que las tareas a realizar sean hechas en equipos de una cierta cantidad de estudiantes en vez de que sean individuales.

La siguiente Sección dará cuenta de que Videoclases es un prototipo aún en desarrollo, comentando los problemas existentes y la antigüedad de la plataforma.

2.2. Un Prototipo

Al momento de ser desarrollada, la plataforma Videoclases se planteó como un prototipo por los alumnos y profesor guía, algo que se sigue manteniendo en la actualidad. Debido a esto, existen ciertas componentes y funciones deficientes, llegando a faltar ciertas vistas esperables para una aplicación web de este tipo. Algunas características incompletas son las siguientes:

1. No se puede modificar una lista de alumnos ya creada para un curso.
2. El profesor es el que asigna las claves a los alumnos. Si un alumno, luego de modificar su clave, se le olvida, debe contactar al profesor del curso correspondiente para recuperarla.
3. No se tiene una vista de administración general para el usuario de tipo profesor, además las opciones que tiene este último son muy limitadas. Para realizar labores más específicas es necesario entrar a la vista de administrador de Django.

4. Al generar un curso con alumnos, se tiene que agregar desde la interfaz de administración de Django la primera fecha de ingreso, si no, el sistema no reconoce al usuario como existente. (Este error ocurrió solamente una vez durante el trabajo adelantado en la aplicación.)

5. La aplicación web fue diseñada teniendo en mente cursos de colegios o universitarios, no para una institución como un Ministerio, por lo que no existe un usuario específico para este caso.

6. Hay que hacer una validación del modelo de datos actual de la plataforma dada la necesidad de agregar un nuevo tipo de usuario.

7. Hay que actualizar las tecnologías usadas porque son versiones extremadamente antiguas y con problemas de compatibilidad.

Al haber sido creada hace ya algunos años, las versiones tanto del framework y del motor de base de datos están desactualizadas, por lo que se tuvo que dedicar parte del tiempo inicial del plan de trabajo entregado en la propuesta en el análisis y revisión de las tecnologías que permiten el funcionamiento de la plataforma.

En la siguiente Sección se comentará sobre el interés del Ministerio de Transportes en este proyecto y en la aplicación web.

2.3. Interés del Ministerio

El ministerio de transportes está muy interesado en probar y usar una extensión de tal plataforma, ya que podría ser un enfoque de comunicación hacia la ciudadanía que no han utilizado antes, según comentaron en reuniones con el profesor Barbay y posteriormente el alumno.

Para informar sobre nuevas normativas o informes, se han usado afiches con distintas temáticas para atraer tanto al público infantil como joven, pero no tienen nada similar a esta plataforma, por lo que les interesa la adaptación de la aplicación web para lograr sus objetivos. Lo ven como algo novedoso porque se trabajaría directamente con alumnos jóvenes y las ideas que estos creen al hacer los vídeos pedagógicos. Además, se lograría validar la utilidad de estos últimos dada la evaluación que se tiene que realizar entre los mismos creadores.

Dado que la plataforma Videoclases permite la creación de distintos grupos de alumnos (cursos) solamente subiendo una planilla, la aplicación web puede ser extendida de forma general a otras instituciones. Sin ir más lejos, distintos Ministerios del país podrían verse interesados en el proyecto dada la facilidad de uso que presenta.

En el siguiente capítulo se comenzará la revisión completa de la aplicación web, pasando primero por el Diseño, luego por la Implementación y finalmente por los procesos de Validación.

Capítulo 3

Diseño

En este capítulo, se explicará el diseño presente en la plataforma Videoclases junto con detalles y comentarios sobre el trabajo realizado.
El capítulo se divide en las siguientes secciones:

- Una descripción y análisis del diseño ya existente. (Sección 3.1)
- Explicación de los cambios hechos en el diseño para este trabajo de título. (Sección 3.2)

3.1. Diseño de Videoclases

La plataforma en la que se trabajó presenta una estructura simple basándose en el framework Django para la construcción de aplicaciones web. El diseño se establece desde los siguientes puntos principales que nos entrega el framework:

- Views (Vistas): Son el esqueleto primordial del software, ya que en ellas se especifica que información, modelos, requerimientos, formularios, plantillas y todas las componentes que Django debe utilizar o entregar como resultado.
- Models (Modelos): Arman la estructura interna de la base de datos relacional que usa Django. Aquí se crean las relaciones comunes entre cada modelo (entidades) como llaves foráneas, relaciones uno a varios, etc.
- Templates (Plantillas): Archivos de extensión **.html** que, en conjunto con los archivos estáticos (JavaScript, CSS, imágenes), son el frontend de la aplicación.
- Archivos estáticos: Contemplan todos los archivos con variadas extensiones (.js, .css, .jpg, etc.) que son la parte visual y estética de la plataforma, además de darle el funcionamiento responsivo a ciertas vistas.
- Forms (Formularios): Son lo que facultan a la página web de recibir inputs para el ingreso y creación de usuarios, por ejemplo. Para la plataforma, permiten la creación de tareas, cursos y posterior modificación de estos.

- urls.py: Archivo de Python en el cual se definen todas las rutas (urls) de la aplicación junto con las interacciones y envío de información.

El modelo de datos tiene las siguientes entidades principales para su correcto funcionamiento:

- School: Representa a la institución educativa que está utilizando la plataforma. Solo tiene como campo el nombre.
- Course: Entidad que hace referencia al curso o asignatura. Tiene como llave foránea el modelo School además de los campos, nombre y año de creación. Este último parámetro es importante para la creación de tareas, ya que, si el curso no es del año actual, no es posible usarlo.
- Teacher: Uno de los 2 tipos de usuarios originales. Representa a los profesores. Tiene como llave foránea School, asociándolo así a un solo establecimiento educacional. Además, puede tener varios cursos asociados, a pesar de que estos últimos sean de otro colegio o institución.
- Student: El segundo tipo de usuario original. Representa a los alumnos. Un alumno puede estar asociado a más de un curso en la plataforma.
- Homework: Entidad que representa a las tareas creadas en la plataforma. Tiene los campos Teacher, Course y Homework_to_evaluate (opcional, se refiere a un video explicativo de la tarea para los alumnos) como llaves foráneas y title, description, video, date_upload (fecha límite para subir la tarea), date_evaluation (fecha límite para evaluar las tareas de sus compañeros), revisión (cantidad de videos a evaluar), criteria y scale (criterios y escala de evaluación) como campos importantes.
- Videoclase: Son las tareas subidas por los alumnos. Su llave foránea es Homework, la tarea generada por el profesor y que los alumnos están respondiendo. Además, tiene los campos group (el número del grupo del alumno; pueden existir tareas grupales), video, question que es la pregunta que el alumno deja planteada para los evaluadores, 3 alternativas, siendo una la correcta a la pregunta anterior y upload_students, la fecha de subida de la tarea.

El esquema completo se puede apreciar en la Figura 3.1 presente en la siguiente página del documento, en donde se detallan todos los modelos, además de los mencionados anteriormente.

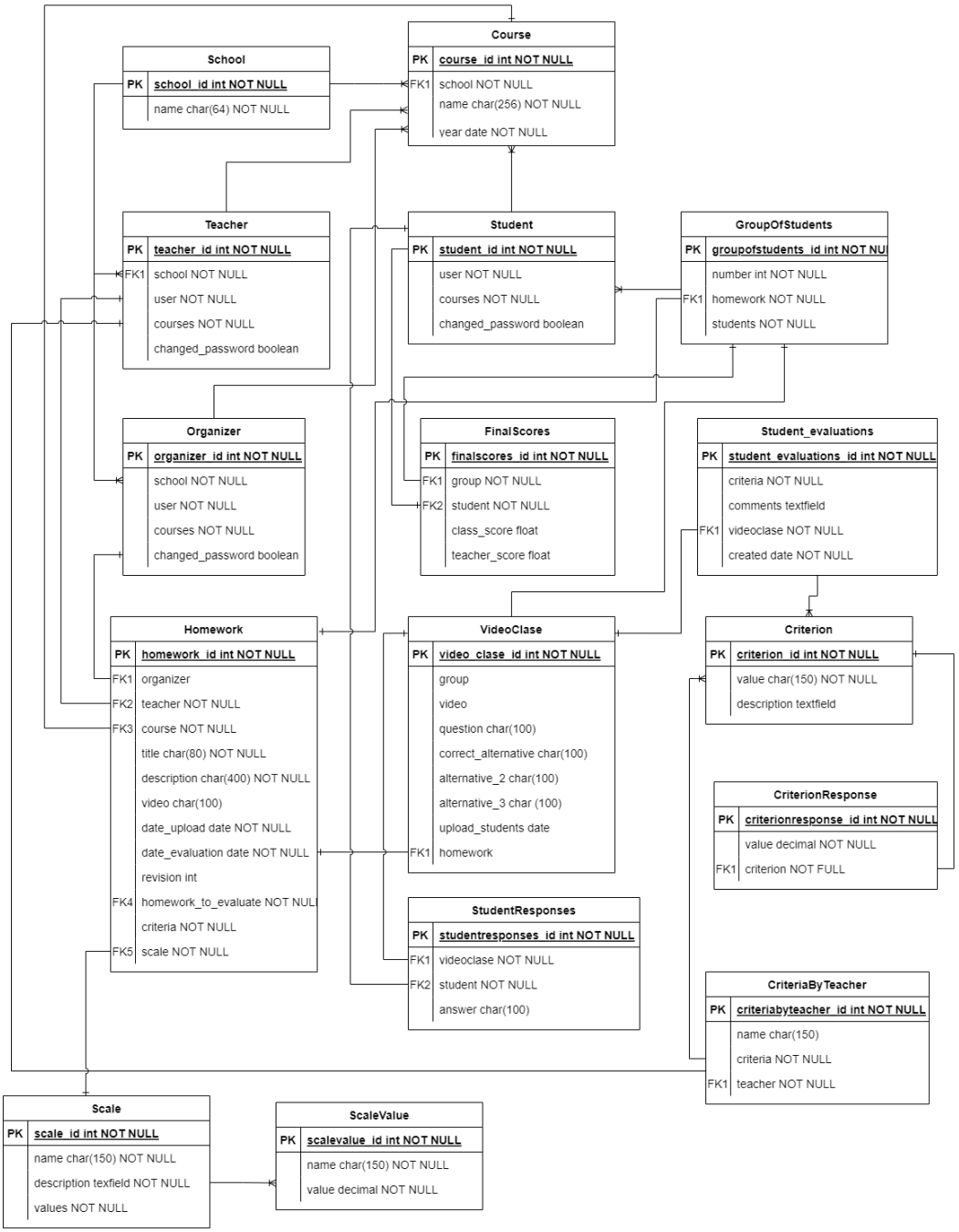

Figura 3.1: Diagrama de tipo entidad-relación de la base de datos de la aplicación.

Videoclases está diseñado de la siguiente manera:
Primero toma la información de las Views, extrayendo desde la base de datos los modelos correspondientes, si es necesario, para luego enviarla a la plantilla previamente definida en el método de Python y así poder renderizar la página con ayuda de los archivos estáticos. Si es que la página necesita de inputs del usuario para crear una tarea, la vista envía lo necesario para poder recibirlo, validarlo si es el formato correcto y posteriormente enviar la petición de guardado a la base de datos. Más detalles de la implementación y el código se explicarán en el Capítulo 4: Implementación.

La estructura general de la aplicación lleva a los usuarios por los siguientes flujos de acciones, representados en el siguiente diagrama 3.2, dependiendo de los casos de uso definidos por los tipos de usuarios.

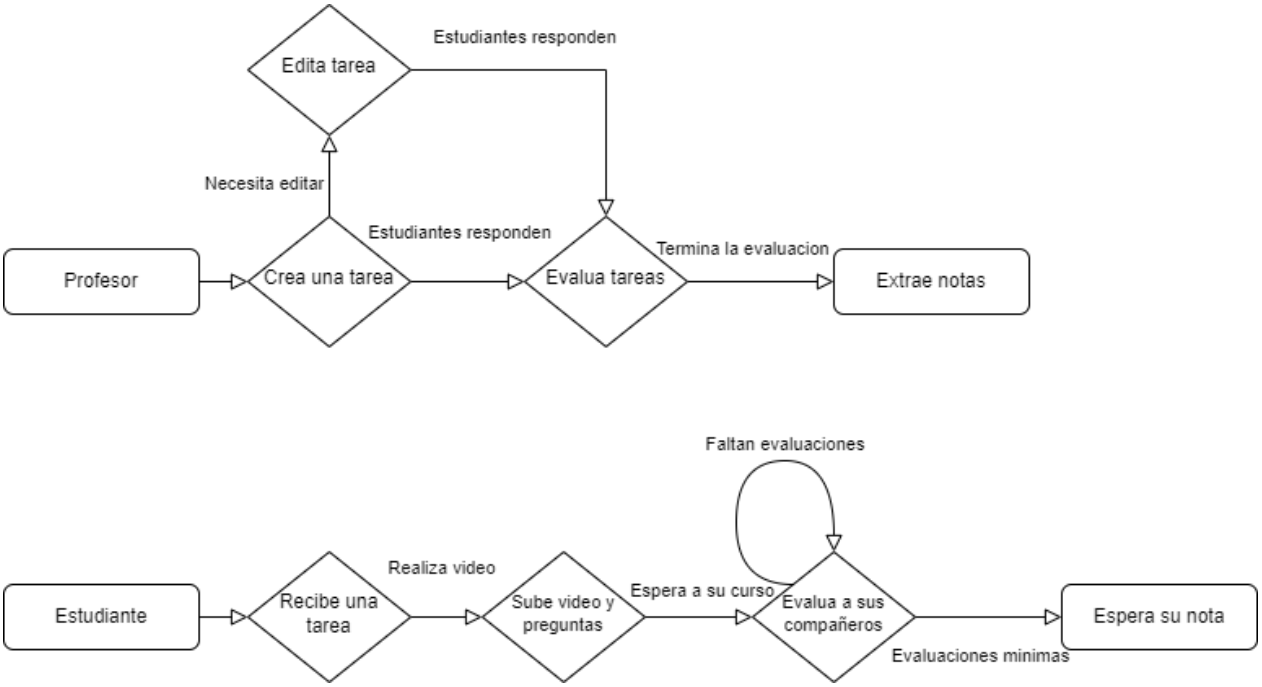

Figura 3.2: Diagrama que representa las acciones generales hechas por los usuarios Profesores y Estudiantes.

En la figura anterior no se incluyeron las acciones de crear un curso por parte de un profesor, además, las tareas no tienen test conceptuales, opción presente en la plataforma, pero que no fue trabajada durante el semestre. Ambas componentes no fueron modificadas en el desarrollo de esta memoria por tiempo y dado que ya son funcionales.

En la siguiente Sección 3.2 se comentarán primero los problemas y correcciones que se tuvieron que hacer al momento de actualizar las tecnologías utilizadas en Videoclases y después, los cambios realizados en el diseño para agregar las nuevas componentes.

3.2. Actualización del diseño

Antes de trabajar agregando nuevas componentes a la plataforma, se debió estudiar a fondo el diseño y su comportamiento para poder seguir un flujo similar al existente.
Durante esta primera labor, al haber actualizado las tecnologías usadas al principio del semestre, existieron varios modelos que de alguna forma dejaron de funcionar como originalmente debían. Por ejemplo, los criterios de las tareas y las fechas tanto de subida como de evaluaciones se dejaron de guardar en la base de datos de la aplicación. Debido a esto, el diseño fue revisado minuciosamente para poder volver a tener una plataforma usable, similar a la versión lanzada en `https://videoclases.teachingislearning.cl`[1] alojada en los servidores Búho de la facultad.

Videoclases, al presentar un diseño simple y funcional, permitió agregar un nuevo tipo de usuario que está por encima de los profesores y alumnos, los organizadores.
Un organizador corresponde a entidades como el Ministerio de Transportes, dado que estos pueden manejar varias actividades con varias instituciones en simultáneo, comprobando los detalles de cada una, los profesores encargados y hasta pudiendo editarlas. La idea detrás es que desde una sola vista pueden recoger toda la información necesaria sobre lo que necesiten de las tareas creadas, sin necesidad de ir por cada profesor. Dentro de la información que pueden verificar se encuentran los detalles de cada tarea y las evaluaciones realizadas a estas últimas, ya sea por parte de profesores o alumnos.

Para agregar estos comportamientos se agregó una nueva entidad al modelo de la base de datos de la plataforma llamado Organizer el cual tiene los siguientes campos:

- School: Un organizador puede estar asociado a uno o más establecimientos educacionales al mismo tiempo. Es debido a esto que puede llevar el registro de todas las actividades a las que este asociado, independiente de la institución educativa.
- User: Al ser un tipo de usuario, debe tener el campo que corresponda a este.
- Courses: Un organizador puede tener más de un curso asociado, siguiendo la misma idea de que puede estar relaciona a más de un colegio o institución.
- Changed_password: Campo de tipo boolean que informa, luego de la creación del usuario por parte del administrador de la aplicación, si es que la persona que lo utilice cambió la contraseña genérica.

El flujo de acciones de un usuario tipo organizador está mostrado en la Figura 3.3. De las acciones posibles se destacan el poder editar cualquier tarea a la cual este asociado y poder revisar y extraer las evaluaciones hechas tanto por los alumnos como por los profesores de la actividad que se desee.

Para lograr mantener el funcionamiento de las demás componentes de la aplicación, el diseño general solamente fue actualizado, replicando el comportamiento del usuario tipo profesor para los organizadores, pero entregándole más información y distribuyéndola de tal forma manteniendo la cohesión de la plataforma.

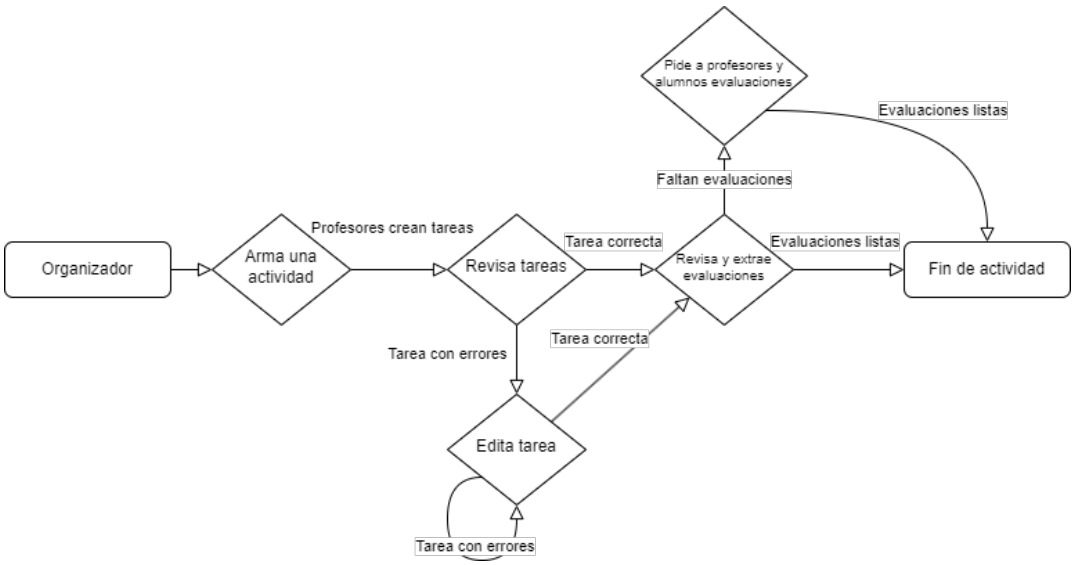

Figura 3.3: Diagrama que representa las acciones generales hechas por los usuarios Organizadores

Dentro de las actualizaciones, se añadió como nueva llave foránea el campo **organizer** a la entidad **Homework** (tareas y actividades). Así, cada tarea puede tener o no un solo Organizador asociado a ella, centralizando varias actividades en una persona. Además de modificar el modelo, se debió agregar a los formularios de tareas la posibilidad de un Organizador para así guardar la información correcta en la base de datos.

La plataforma ahora permite a cualquier usuario poder recuperar su contraseña de forma normal, agregando una nueva acción posible para todos los usuarios. La recuperación funciona enviando un mensaje personal al correo asociado al usuario si es que este último está registrado. Antes, al presionar el enlace para recuperar contraseña, la plataforma te pedía enviar un correo a la cuenta de administración para hacer el cambio de forma interna con el administrador de Django. Esta nueva funcionalidad agrega una nueva acción posible al flujo de todos los usuarios, pero no modifica diseño general de Videoclases, ya que no se cambiaron o agregaron campos a los modelos.

En el siguiente capítulo se explicará toda la implementación de Videoclases, explicitando los cambios realizados, los problemas al actualizar los frameworks, las dificultades en el deployment tanto local como en Búho y porque se decidió utilizar las tecnologías de la nube para la validación.

Capítulo 4

Implementación

En el presente capítulo se detallará la implementación realizada de la aplicación web Videoclases y todo el trabajo que corresponde al lanzamiento apropiado de la plataforma para lograr realizar pruebas para el proceso de validación explicado en el siguiente capítulo. Se debe destacar que durante cada Deployment se realizaban pruebas para verificar el correcto funcionamiento de Videoclases.
Se utilizó un nuevo repositorio de GitHub [14] al comenzar el trabajo de implementación. El capítulo se divide en las siguientes secciones:

- Deployment local y remoto en servidores búho del Departamento de Ciencias de la Computación (DCC) de la aplicación para un entorno de pruebas y desarrollo. (Sección 4.1).
- Trabajo llevado a cabo en la aplicación web Videoclases. (Sección 4.2).
- Breve explicación de las tecnologías de la nube y segundo deployment. (Sección 4.3).

4.1. Deployment de Videoclases y Búho

Videoclases, como se mencionó anteriormente, está disponible como código abierto en un repositorio de GitHub [2]. Luego de tener acceso durante el desarrollo de la propuesta de esta memoria el semestre pasado, se procedió a trabajar sobre el software, labores que quedaron plasmadas como trabajo adelantado en el documento mencionado. Destacando lo más importante se tiene que se logró un deployment de forma local de toda la aplicación y se apoyó al profesor guía durante el semestre con el uso de Videoclases en un curso dictado en la facultad, creando usuarios tipo profesor, resolviendo dudas de los alumnos en el uso de la plataforma y corrigiendo ciertos errores que fueron ocurriendo.

Al comenzar el presente semestre, se debió realizar un nuevo deployment del software debido a complicaciones con el computador en el cual se estaba trabajando. Al hacer este segundo proceso, se notó que varias librerías requeridas para el correcto funcionamiento estaban con problemas. Esto fue debido al proceso de actualización que tuvo la plataforma mencionada

reiteradas veces en este documento. Ante esto, se modificó el archivo de requerimientos de Python, eliminando así los errores y problemas para un siguiente despliegue.

Durante este nuevo proceso se comenzó a trabajar con los servidores Búho del DCC con unas credenciales entregadas por el profesor Jérémy para ingresar. Búho funciona de forma similar al servidor Anakena, también del departamento, ya que se generó una carpeta con el nombre del alumno (gsilva) para poder llevar a cabo los trabajos necesarios desde ahí. Luego de generar un certificado SSH y enviarlo a Sistemas del DCC, se logró el acceso a los servidores y a la carpeta necesaria. Ya dentro, mediante el programa FileZilla [9] se enviaron los archivos necesarios para el deployment. Este programa fue usado durante toda la fase de implementación para una rápida transferencia de los archivos. Con todo listo se intentó lanzar la aplicación, pero ocurrieron dos problemas:

1. Los archivos estáticos (visuales y responsivos) no eran cargados por el servidor, provocando que la pagina se viera sin formato y con un diseño plano como se ve en la Figura 4.1. En el Anexo B, Figura B.1 se ve el código de error mostrado por la consola del navegador.

2. La única página funcional era el inicio, al presionar cualquier otro enlace, botón o incluso intentar ingresar como usuario llevaba a un error y una vista en blanco.

La solución de ambos errores consistió en modificar el archivo **settings.py** que, como dice su nombre, configura los parámetros necesarios de Django para lanzar la aplicación web. Para el primero, se debió añadir el prefijo $\sim gsilva/$ a la variable que recibe la ruta de la carpeta con los archivos estativos *STATIC_URL*. Con esto, Django busca directamente el material necesario para el deployment. Con respecto al segundo error, se decidió dejarlo de lado momentáneamente para enfocarse directamente en el trabajo a realizar para la memoria, ya que en el despliegue local si funcionaba correctamente la aplicación. Cuando se retomó dicho error para el despliegue del ambiente de pruebas para el proceso de validación, luego de mucha investigación y ayuda de Sistemas, se encontró como solución agregar la variable *FORCE_SCRIPT_NAME* con el valor $/\sim gsilva$. Como dice su nombre, esta variable fuerza a Django a agregar el prefijo $\sim gsilva$ a todos los enlaces de Videoclases, ajustando a la vez las referencias entre ellos. Con esto, el despliegue en Búho fue logrado y utilizable. El extracto de código del archivo de configuraciones de Django se muestra en la Sección A.2.1, código A.1.

A pesar de haber solucionado ambos errores, la versión desplegada en Búho no fue utilizada en el proceso de validación debido a que algunas componentes que utilizaban **AJAX** (Asynchronous JavaScript And XML) no lograban entregar la información responsiva necesaria para algunas vistas, logrando que el comportamiento básico necesario para poder validar la plataforma no fuera posible. Se cree que esto fue debido a problemas con los permisos del servidor, ya que en la consola del navegador web aparecieron las solicitudes bloqueadas con el código 403 Forbidden.

En la siguiente sección se comentará el trabajo hecho en la aplicación durante este semestre, el cual pudo hacerse gracias al despliegue de forma local antes mencionado, incluyendo los problemas ocurridos y explicaciones del funcionamiento de Videoclases.

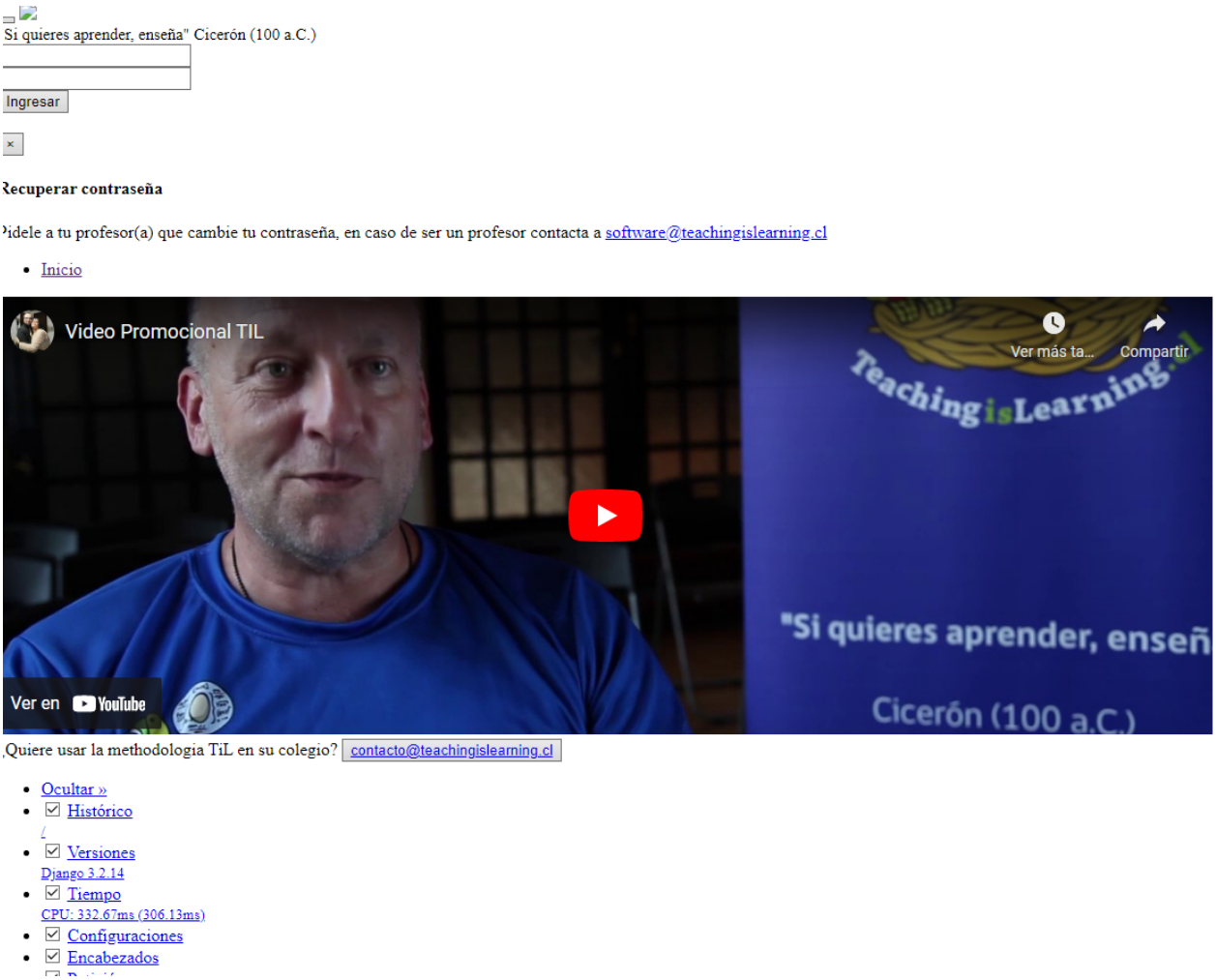

Figura 4.1: Página principal sin estilo y con un formato plano.

4.2. Desarrollo del prototipo

Previamente a trabajar en Videoclases, se verificó que la plataforma funcionara completamente luego de las actualizaciones de las librerías y los frameworks asociados. Para esto, se crearon nuevos usuarios: un profesor y 4 alumnos. Estos ayudaron a poder usar las funcionalidades originales de la plataforma antes de comenzar a trabajar de lleno en los cambios propuestos. Las pruebas consistieron en:

- Crear un curso nuevo y asociarlo a la Universidad de Chile.
- Generar 2 actividades, las cuales tenían nombres diferentes y número de criterios distintos.
- Se revisó que las actividades se crearan de forma correcta.
- Se modificaron y nuevamente se revisaron.
- Con una cuenta de alumno se subió un video como respuesta a la tarea y como profesor se revisó y se asignó una nota.

Durante este proceso de prueba, se confirmó que Videoclases funcionaba perfectamente como su versión existente y usada en los cursos del profesor guía.

Antes de mencionar el trabajo realizado se comentarán ciertas componentes clave de la plataforma web para definir correctamente su funcionamiento. En el Capítulo 3 se habló sobre como la aplicación está estructurada con un formato básico de Django mediante vistas, plantillas, modelos, etc. En esta Sección, comenzaremos mostrando y explicando un ejemplo de vista y plantilla para luego interiorizarnos en el trabajo efectuado.

Para empezar, el código A.2 en el Anexo A.2.1 corresponde a la vista para los usuarios tipo profesores.

Esta es una estructura típica de una vista de usuario de Django. Se define la plantilla *'teacher.html'* a la cual debe entregar la información guardada en la variable *context*. Luego, comienza a llenar *context* solicitando la información a los modelos correspondientes, en este caso *Homework*. Finalmente, retorna la variable y se define el método para despachar al usuario a la página correcta. El siguiente paso es definir la plantilla a la cual debe llegar la información. En el código A.3 presente en el Anexo A.2.1 se ve a *'teacher.html'* el cual recibe lo entregado por la vista y lo renderiza en una tabla.

Pero, ¿cómo Django sabe que debe dirigir a esa plantilla con esa información en específico?. El paso que responde a esta pregunta está en la definición de todas las rutas de la aplicación en el archivo **urls.py**. En este archivo están todas las rutas posibles de Videoclases relacionando la vista a un enlace. Para terminar este ejemplo, la ruta que corresponde a la vista de un profesor es la siguiente:

```
...
re_path(r'^teacher/',
        vv.TeacherView.as_view(),
        name='teacher'),
```

Código 4.1: Definición de la ruta para la vista de profesor.

Durante el presente semestre se trabajó en Videoclases intentando seguir el Cronograma planteado en la Propuesta de este trabajo de título (Ver Anexo B, Figura B.2). Como se mencionó en la sección anterior, se hizo un despliegue de la aplicación para luego realizar una validación inicial de los modelos y del código en general dada la actualización de las tecnologías utilizadas. Al comenzar dicha validación, se notó que varias componentes del programa dejaron de funcionar, tales como comportamientos responsivos y de guardado de datos. Ante esto, se hizo una revisión completa del código, buscando y corrigiendo estos problemas. Los más importantes que vale la pena destacar fueron dos dado que afectaban a la creación y edición de tareas:

- El selector para las fechas, creado con **JavaScript**, dejo de funcionar repentinamente. El error mencionaba que la librería usada no existía. Para solucionar el problema del selector se decidió simplificar el HTML para la página, cambiando el tipo de entrada de este campo del formulario a *input='date'*. Con esto, cada navegador desplegará su selector de fecha propio, además, se tuvo que retocar las validaciones internas del formulario tanto para la creación como edición de tareas. Un extracto de **JavaScript** para cumplir dichas funciones está en la Sección A.2.1, codigo A.4.

- Los criterios no eran guardados al generar una nueva tarea. Se revisaron como estaban estructurados en el modelo y su guardado en los formularios. Extrañamente, algunas variables cambiaron de nombre y ciertas funciones fueron modificadas por las actualizaciones. Ante esto, se corrigieron dichas variables y se volvió al comportamiento original de las funciones para un correcto guardado de los criterios. Un extracto de **JavaScript** para verificar lo mencionado está en la Sección A.2.1, código A.5 y A.6.

Al notar errores de este estilo, se decidió tomar el tiempo necesario para revisar y corregir la plataforma a fondo y así evitar encontrarse con "sorpresas" durante el trabajo. Durante este proceso se arreglaron varios detalles, tanto visuales como funcionales, por ejemplo, librerías que tenían errores de renderizado, tablas que perdieron el comportamiento responsivo, entre otros, que se prefirieron no mencionar tan a fondo debido a la simplicidad de estos.

Ya con la aplicación web revisada completamente se comenzó con la siguiente funcionalidad: Añadir una página para la recuperación de contraseñas. Antes, al apretar el enlace correspondiente a esta labor, aparecía un mensaje pidiendo que se contactara a los administradores para el cambio de contraseña. Este comportamiento se ve en la Figura 4.2.

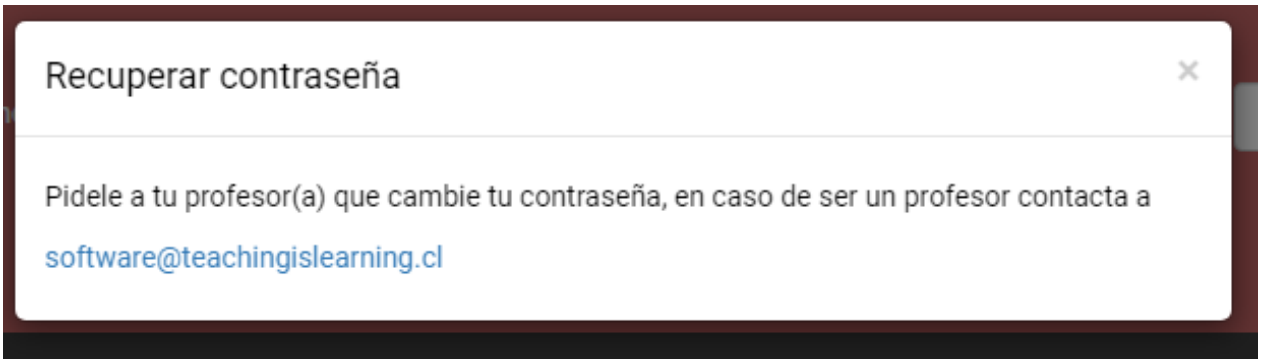

Figura 4.2: Mensaje mostrado al presionar enlace para recuperación de contraseña.

Para solucionar esto se debieron agregar 4 plantillas HTML. En el Anexo B se encuentran algunas imágenes (Figuras B.3, B.4) con el resultado de las plantillas. Los 4 archivos fueron los siguientes:

1. password_reset: Recibe el correo asociado a un usuario y envía la solicitud.
2. password_reset_done: Página que confirma el envío de la información al correo.
3. password_reset_confirm: Accesible solo desde un enlace personal. Recibe la nueva contraseña.
4. password_reset_complete: Muestra que el cambio fue correcto y permite volver al inicio.

Con las plantillas listas, el siguiente paso consistió en crear la vista correspondiente en **views.py** para el proceso. En el código A.7 se muestra la implementación completa. El proceso que ocurre en la vista es: Se recibe la solicitud con un correo válido, se envía un correo utilizando la plantilla definida en un archivo **.txt** y finalmente se recibe el resultado y

se actualiza la contraseña. Además de esto, se agregaron las rutas correspondientes a **urls.py** (Ver código A.8) para cada plantilla mencionada anteriormente. Un detalle importante es que solo la primera plantilla necesitaba la creación de una vista específica debido a que para las demás Django provee una solución interna ya implementada. Finalmente, para el envío de correos se debió modificar nuevamente el archivo **settings.py**, añadiendo una variable *EMAIL_BACKEND* que recibe los parámetros de un servicio de correos para el envío de un email personalizado. Para la validación se empleó el propio correo del alumno, pero se plantea como trabajo futuro utilizar algún servicio de mensajería, como el que ofrece Amazon Web Services (AWS) de forma gratuita.

La siguiente componente que se agregó fue un nuevo tipo de usuario con sus vistas y funciones propias: Organizador. Como se mencionó en el Capítulo 3, los organizadores están por encima de los profesores con respecto a la información que tienen a su disposición. Además, este usuario puede también editar tareas y revisar evaluaciones que estén asociadas a él. Lo primero que se hizo fue agregar un nuevo modelo para la base de datos que corresponda a un organizador. Ya con esto resuelto, se realizó un proceso muy similar al de añadir la recuperación de contraseñas. Se agregaron las vistas necesarias, las plantillas y finalmente las rutas.

Para las vistas y plantillas se desarrolló lo siguiente:

- OrganizerView: Vista muy similar a la de profesores. La diferencia importante es que se utiliza otra plantilla y se solicita la información a través del propio organizador al modelo de datos. Su plantilla es *'header.html'* que muestra todas las tareas en una tabla responsiva, además de las instituciones a las que esta realacionado (Figura B.5, código A.9).

- VideoclasesHomeworkOrganizerView: Vista que permite al organizador revisar una tarea específica. Su plantilla es *videoclases_homework_organizer.html* y muestra cada tarea subida junto con las respuestas de los alumnos. (Figura B.6).

- CourseOrganizerView: Vista que entrega la información completa de un curso presente en la aplicación. Para cada alumno muestra el nombre completo, usuario, tareas entregadas, pendientes y corregidas. La plantilla correspondiente es *'course_organizer.html'* (Figura B.7).

- EditHomeworkOrganizerView: Vista que permite a los organizadores ver los detalles y editar tareas ya creadas por otros profesores. Su funcionamiento es similar a la edición del usuario profesor, pero tomando la información proveniente directamente desde la tarea, no desde el usuario que está editando. El código A.10 muestra la vista completa en donde existen los métodos para enviar la información a la plantilla *'edit-homework-organizer.html'* (Figura B.8) y validar los cambios hechos por el usuario para luego guardarlos en la base de datos. También se debió añadir un nuevo archivo llamado **EditHomeworkOrganizer.js** (código A.11) que también actúa como validador del formulario y organiza la información para ser guardada. Este último **JavaScript** es casi una copia del usado por los profesores porque tienen los mismos campos que validar y guardar.

Con respecto a las rutas, se agregaron todas las correspondientes para los organizadores. El extracto de código A.12 en el Anexo A.2.1 muestra las líneas correspondientes para cada ruta definida para las vistas de este nuevo tipo de usuario.

Además de las nuevas vistas, se debió modificar levemente la que corresponde a la creación de tareas junto con su plantilla para permitir a un profesor agregar a un organizador como encargado de una tarea. En la Figura 4.3 se ve el nuevo campo desplegable que puede mostrar a los organizadores disponibles.

Figura 4.3: Formulario de creación de tareas en donde se aprecia el nuevo campo desplegable para agregar organizador.

Un detalle sumamente importante es que los organizadores, al igual que los profesores, solo pueden ser creados desde la administración de Django. Además, se deben asociar las instituciones educativas que uno requiera de forma manual para cada organizador.

Finalmente, la vista de concursos mencionada en el Cronograma B.2 fue reemplazada por la de Organizadores, ya que cumplen la misma función de contener y mostrar la información de todas las tareas asociadas al usuario. Lo que no se logró hacer es que un organizador pudiera generar tareas. Siguiendo la misma idea, durante la revisión completa de la plataforma se notó que los Criterios pueden ser fácilmente modificados y añadidos desde el administrador de Django, por lo que no se vio necesario (y también en honor al tiempo en el semestre) el agregar una página solamente para esta función.

En la siguiente Sección se comenzará explicando sobre las tecnologías de la nube, específicamente Huawei Cloud, y luego se comentara el nuevo proceso de deployment utilizado para la validación.

4.3. Cloud y nuevo deployment

El desarrollo de nuevas tecnologías ha llevado a preferir servicios en vez de tener infraestructura física (servidores, computadores, etc.) en varias empresas tanto de tecnología como de cualquier rubro debido a los costos de mantención y el tiempo de vida útil determinado por cada máquina [6]. Ante esto, Google, Amazon, Microsoft y Huawei (presentes en Latinoamérica) han impulsado el desarrollo en la nube pública con los conceptos *Platform as a Service (PaaS), Software as a Service (SaaS) e Infraestructure as a Service (IaaS)*. Durante el presente año, se ha estado trabajando en formato híbrido part-time en Huawei Cloud [7] en el cargo de Arquitecto de Soluciones (Solution Architect, SA) Interno. Esto significa ser una especie de practicante, pero con tiempo de estadía extendido a 1 año completo. Durante

este periodo se aprendió sobre las tecnologías Cloud Native y cómo funciona el desarrollo en la nube para distintos tipos de situaciones, llegando a trabajar con varios Ministerios del país ayudando en la migración de su infraestructura.

Como interno, le entregaron al alumno una consola de pruebas en donde se pueden crear máquinas virtuales en la nube junto con otros servicios como almacenamiento, direccionamiento IP, etc. Dado que era necesario un nuevo ambiente para poder realizar el proceso de validación de la plataforma, se produjo una nueva rama del proyecto en GitHub llamada *DeployHuawei* que tiene como objetivo el realizar un despliegue utilizando los servicios Huawei Cloud. Para poder hacer esto se explicarán algunos conceptos importantes de la nube de Huawei:

- ECS: Corresponden a las máquinas virtuales de la nube. Tienen como parámetros la cantidad de núcleos virtuales (vCPU) y el tamaño de la memoria RAM. Existen una multitud de combinaciones posibles junto con distintos tipos de ECS dependiendo de las necesidades. Por ejemplo, para este proyecto se usó una de tipo "Computo general" con 2 vCPUs y 4 GB de RAM, una máquina más que suficiente para una página web.
- EVS: Son los discos virtuales para cada ECS. El tamaño puede llegar a ser de Terabytes y se tienen distintos tipos en donde varía la velocidad de lectura y escritura. Para Videoclases basto con un disco de 50 GB de tipo general dada la simplicidad de la aplicación.
- Elastic IP: Este servicio entrega una dirección IP pública a una máquina ECS para que esta pueda ser accedida desde internet. Para este proyecto, se empleó solamente una y se especificó un ancho de banda de 1 Mbit/s.

En la Figura 4.4 se ve la máquina creada y su estado. Se usó el sistema operativo Ubuntu Server 20.04 dado que es gratuito y de fácil uso a pesar de no tener interfaz de usuario. También se ven las dos IP asociadas al ECS, la publica, que corresponde a la Elastic IP y la privada, propia de la máquina.

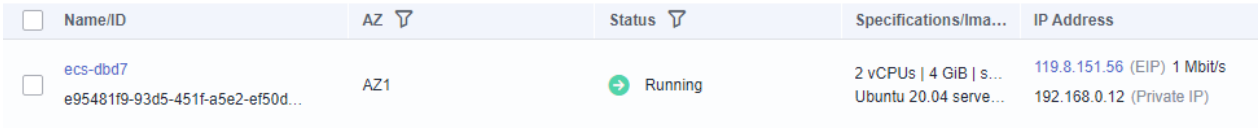

Figura 4.4: Maquina ECS desplegada en Huawei Cloud. Imagen extraída de la consola en donde también se ve el sistema operativo y la Elastic IP.

Ya con la máquina virtual creada se accedió a ella para descargar la rama del repositorio correspondiente. Para conectarse a la máquina se usó la consola de comandos de Visual Studio Code [12], IDE utilizado para todo el desarrollo de este trabajo de título, y el protocolo SSH. Con la conexión lista, se procedió a descargar la rama *DeployHuawei* en la ECS. Dado que las versiones de Python de la máquina personal del alumno y de la máquina en la nube eran distintas, se actualizó el archivo **requirements.txt** con las librerías necesarias para todo el funcionamiento de Videoclases. Finalmente, se logró un deployment en Huawei Cloud de la plataforma, permitiendo así el proceso de validación final. La página puede ser accedida desde

el siguiente enlace `http://119.8.151.56` (enlace deprecado por ahora) y es completamente funcional a diferencia de la que está presente en los servidores de Búho de la facultad.

Luego del despliegue en la nube, se hicieron las mismas pruebas para verificar el funcionamiento original de la aplicación junto con los cambios implementados en este capítulo. A los test se le agregaron:

- Se creó un usuario organizador y se asignó a las tareas creadas durante el proceso.
- Dicho usuario modificó las tareas y se verificó nuevamente que los cambios surtieron efecto.

En el siguiente capítulo se explicará todo el proceso de validación para Videoclases, incluyendo el contacto con el Ministerio de Transportes y el uso de un formulario para usuarios finales.

Capítulo 5

Validación

En este capítulo se explicará todo el proceso de validación de la plataforma web Videoclases. Es importante destacar que las personas que actuaron como usuarios finales para las pruebas son todas del ámbito educacional (profesores, ayudantes o auxiliares). Además, se comentará sobre la reunión final que se tuvo con el Ministerio de Transportes sobre la aplicación. El Capítulo se divide en las siguientes secciones:

- Reunión final con el Ministerio de Transportes y las ideas para el futuro.
- Creación de un formulario de usuario final junto con una lista de instrucciones y los resultados del proceso.

5.1. Reuniones con el Ministerio de Transportes

Durante todo el desarrollo de la memoria, considerando también el primer semestre del año en donde se entregó la propuesta de trabajo de título, se mantuvo contacto con funcionarios del Ministerio de Transportes a modo de clientes para la plataforma Videoclases. Al final del semestre de otoño del presente año se tuvo una reunión con un funcionario público de la Comisión Nacional de Seguridad de Tránsito (CONASET) la cual estaba encargada del área de educación. Durante esa reunión la funcionaria nos comentó que le encantaba la idea presentada por la plataforma, pero que veía muy difícil ayudarnos a probar la Videoclases en colegios manejados por la entidad debido a que se tiene que pedir autorización a más autoridades y toda la burocracia implicada lleva demasiado tiempo, llegando incluso a demorarse un año. Dado este problema, se descartó el uso de la aplicación en un colegio, ya que no se tiene el tiempo suficiente para esto.

Ya en el semestre de primavera, trabajando de lleno en la plataforma, se tuvo una reunión nuevamente con los funcionarios con los que se empezó el proyecto el día 24 de noviembre. El propósito de esta reunión fue hacer una demostración en vivo (de forma remota, ya que fue una reunión online) de la nueva versión de la aplicación web. La demo consistió en:

- Mostrar las interfaces para un usuario profesor. Para esto se creó un usuario de prueba

y se asignaron tareas ya existentes en el sistema para demostrar la edición y creación de estas últimas. Mientras se navegaba por la página se iba explicando detalladamente los pasos ejecutados.

- Mostrar las interfaces para un usuario organizador. Esto tenía una gran importancia dado que el Ministerio podría actuar como dicho tipo. Para la demostración se generó un usuario de prueba con tareas asignadas previamente en la plataforma y así poder replicar el flujo de acciones mostrado en la Figura 3.3 del Capitulo 3.

Al término de la demostración se pidieron las opiniones de los funcionarios participantes. El feedback fue completamente positivo. Estaban muy interesados en la plataforma dada su facilidad de uso, aunque aún ven la herramienta como un piloto o prototipo. A pesar de esto, les surgió la idea de utilizarla de forma interna en el futuro para talleres desarrollados para los propios trabajadores del Ministerio. Así, la persona encargada podría gestar una actividad y los demás funcionarios deberán subir los videos respectivos sobre el taller que se les está enseñando, agilizando el proceso de aprendizaje. También vieron con muy buenos ojos usar Videoclases en otros Ministerios para labores similares y actividades con la ciudadanía.

A pesar de que el feedback fue mayormente positivo, como validación no es suficiente, dado que los funcionarios solo vieron una demostración y no interactuaron directamente con la plataforma. Es debido a esto que de esta reunión de "validación" fueron tomadas más en cuenta las posibles ideas como trabajo a futuro más que una validación formal propiamente tal, aunque el feedback recibido sirve como guía para saber si el trabajo va en el camino correcto. Aún se mantiene el contacto con los funcionarios, por lo que, más adelante, se podría seguir explorando las ideas plasmadas.

En la siguiente sección se explicará la segunda validación hecha mediante un cuestionario de usuario final hacia específicamente profesores, mostrando los resultados relevantes y opiniones de los encuestados.

5.2. Validación con formulario

Para poder realizar una correcta validación, se recordaron los conocimientos adquiridos en el curso CC6502-1 Taller de Usabilidad de Interfaces de Software tomado en primavera del año 2021, en donde se aprendió el cómo validar la usabilidad y otras capacidades de distintos tipos de software, entre ellos, aplicaciones web. Gracias a esto, se utilizó y modifico un formulario armado por el profesor del curso en donde se tenían las preguntas necesarias para un correcto proceso de validación.

El propósito de este formulario, además de analizar la usabilidad del prototipo, fue también el de "testear" y reportar posibles errores que podrían surgir durante un uso normal de Videoclases dado que más personas utilizarían la plataforma durante este periodo de validación, logrando una mayor retroalimentación general en un uso normalizado.

El formulario [13] usado para Videoclases estaba estructurado de la siguiente manera:

- Breve introducción de la plataforma y el objetivo de la encuesta.
- Una lista de 11 instrucciones a seguir en el contexto de un usuario tipo profesor.
- Una lista de 5 instrucciones a seguir en el contexto de un usuario tipo organizador antes de responder las preguntas.
- 19 preguntas a responder por parte de los usuarios finales.

Se adjunta en la Sección A.3.1 del Anexo A.3 la breve introducción y las instrucciones explícitas para los usuarios. Todas estas instrucciones también fueron realizadas por el alumno, pero con una visión más enfocada en probar la implementación realizada en busca de errores en vez de la usabilidad. Por lo tanto, este proceso de validación sirvió para "testear" el trabajo realizado de forma directa por parte del alumno.

Las instrucciones para el primer tipo usuario consistían en simular el comportamiento de un profesor que ya tiene un curso cargado en la plataforma y quiere crear una nueva tarea o actividad. La lista empieza con instrucciones para el ingreso a la plataforma desplegada en la nube pública siguiendo el enlace `http://119.8.151.56/` utilizando una cuenta generada específicamente para este proceso de validación. Luego, como instrucción opcional, se sugería ver el video de presentación del proyecto "Teaching is Learning" en la página de inicio. La lista de órdenes posteriores es la siguiente:

1. Revisar una tarea de prueba ya originada en el sistema viendo sus detalles.
2. Crear una tarea llenando todos los campos obligatorios correspondientes, incluyendo los criterios, escalas de evaluación y asignar el organizador.
3. En la tabla de tareas ingresar a la actividad creada para revisar que todo este correcto y además editar algunos campos.
4. Cerrar sesión.

Con esta primera parte completa, se les pidió a los encuestados ingresar nuevamente a Videoclases pero utilizando una nueva cuenta de tipo organizador. Las instrucciones resumidas fueron las siguientes:

1. Revisar la tarea creada como profesor en los pasos anteriores o cualquier otra del sistema.
2. Ver los detalles y editar dicha tarea agregando un nuevo criterio de evaluación.
3. Guardar y revisar que los cambios se hayan realizado.
4. Cerrar sesion.

Finalmente, los usuarios finales debieron responder las 19 preguntas teniendo en cuenta su trabajo en la aplicación web, las funcionalidades observadas y posibles errores encontrados

o que les sucedieron. Las preguntas estaban enfocadas en la usabilidad y apreciación general de la página web en su totalidad, con preguntas tales como si el sitio web es fácil de navegar o si las componentes con las que se puede interactuar (botones, enlaces, tablas) eran fácilmente identificables y funcionaban de forma correcta.

El formulario de validación fue respondido por 8 personas, de las cuales uno es profesor de música, otro es profesor de educación física y la otra, profesora de la FCFM del DCC, enfocada en cursos de Formación Integral de Especialidad y Trabajo en equipo. Los demás fueron y son ayudantes y/o auxiliares de muchos cursos, mayoritariamente del Departamento de Ingeniería Matemática (DIM) y del DCC, mientras que uno está terminando la carrera Ingeniería Comercial en la Universidad Adolfo Ibáñez. Las respuestas tenían una escala de 5 apreciaciones, partiendo de 1: Muy en Desacuerdo y subiendo progresivamente hasta 5: Muy en Acuerdo.

Los resultados finales arrojaron que la totalidad de los encuestados estaban de acuerdo con que el sitio web fácil y cómodo de navegar, agregando que las páginas cargaban rápidamente (menos de 30 de segundos de demora) y que el diseño general de la aplicación web, dada su objetivo, es apropiado. Además, estaban muy de acuerdo en que todo el contenido mostrado es muy relevante para su propósito. A pesar de esto, 2 personas consideraron que la interfaz visual del sitio no es placentera. Todas las preguntas, a excepción de la mencionada anteriormente, presentaban respuestas partiendo desde el número 3 (De Acuerdo) hacia arriba de la escala de apreciación. Se ve que los usuarios estaban conformes con este nuevo prototipo para Videoclases dado que la última pregunta (antes de las que necesitaban un texto breve como respuesta) que corresponde a la calificación global del sitio web analizado tiene seis respuestas con un 4/5 y dos con un 5/5 según la escala definida.

En la parte final del formulario se hicieron 3 preguntas que necesitaban una respuesta breve escrita. Estas preguntas eran: Justifique su calificación global, errores encontrados en la página web y sugerencias para una nueva versión en el futuro. Para cada pregunta, las conclusiones son las siguientes:

1. Las respuestas fueron todas positivas pero con ciertos reparos. Por ejemplo, el diseño y todo el apartado visual es algo que se criticó y que puede ser mejorado con colores y detalles más llamativos. Una persona comentó que: "La página es fácil de ocupar, pero tal vez es muy enredada el cómo poder subir la información/tarea. Una vez ya ingresado a la realización de la tarea, está todo muy claro." Esto lleva a pensar en mejorar la interfaz general de Videoclases para lograr que los usuarios no se pierdan durante su uso.

2. Con respecto a los errores, no se encontraron problemas que interrumpieran el flujo general de la aplicación, pero si algunos detalles como por ejemplo que los colores llevan a hacer que la plataforma sea muy monótona, llegando a confundirse en ciertas vistas. También se comentó que al editar una tarea al parecer se borró cierta información como las notas ya ingresadas a la base de datos.

3. Dentro de las sugerencias para un trabajo futuro se destacan agregar más detalles para los estudiantes, como el porcentaje de ponderación de las tareas. Se comentó la idea de agrandar la letra del nombre de los alumnos al revisar una tarea, junto con agregar

un buscador por estudiante para agilizar la revisión de los profesores y organizadores y también agregar la opción de mezclar las escalas de evaluación. Un detalle pequeño mencionado es que las fechas había que seleccionarlas de nuevo al momento de editar una tarea. Nuevamente, se mencionó el tema visual y los colores de la aplicación.

Todas las respuestas, sin la información personal de los usuarios, se pueden verificar en la Sección A.3.2 (tabla A.1) del Anexo A.3.

A modo de conclusión de este segundo proceso de validación, se consideró que fue más útil que el mencionado en la Sección 5.1, ya que se logró trabajar con usuarios finales reales mediante un formulario con instrucciones para que ellos mismos pudieran interactuar directamente con el software a diferencia de la primera validación en donde se hizo solo una demostración. A pesar de esto, hubiera sido aún mejor haber logrado una actividad con un curso de la facultad o colegio en donde Videoclases pudiera ser usado en su totalidad y no solo las partes que fueron modificadas por el alumno para este trabajo de título. Así, se podría validar correctamente la plataforma y las interacciones entre las componentes de forma completa y no parcial. Finalmente, se considera este proceso como algo muy útil como primera, aunque única, instancia de validación más cercana a la realidad, porque arrojo opiniones desde un contexto más cercano a lo esperable junto con ideas para un trabajo a futuro.

En el último capítulo de este informe se concluirá sobre el trabajo realizado en Videoclases durante el semestre, junto con las problemáticas ocurridas y las ideas para un posible trabajo futuro.

Capítulo 6

Conclusión

Concluimos este informe con un resumen de las contribuciones realizadas a la aplicación web Videoclases (sección 6.1), una sección de discusión sobre los objetivos planteados, potenciales problemas y detalles con el trabajo realizado (sección 6.2) y, finalmente, una visión sobre el trabajo futuro a realizar en el software (sección 6.3).

6.1. Contribución

Luego de analizar y validar la usabilidad del modelo de datos de Videoclases, se llevó a cabo una actualización de las tecnologías utilizadas por la aplicación, destacando Django y Python como las más importantes, dado que son las bases del software en cuestión. Este proceso se hizo con la ayuda de Jorge Ampuero, dado que él, como se mencionó anteriormente, fue la última persona en trabajar en la página web. Esto llevo a algunos problemas de compatibilidad con algunas librerías de JavaScript que daban todo el funcionamiento a ciertas componentes de Videoclases. Gracias a este proceso, se aprendió más a fondo sobre las tecnologías más específicas usadas en la página. Luego, se validó que la plataforma mantuviera todas las funcionalidades básicas y previas a este trabajo de título. Este proceso de revisión producido por la actualización de las tecnologías fue un trabajo que llevo más tiempo de lo esperado debido al salto entre las versiones antiguas y empleadas actualmente. Cabe notar que, además de las librerías de **JavaScript** mencionadas anteriormente, varias vistas y componentes internas comenzaron a fallar, provocando que la plataforma no fuera funcional.

El diseño de Videoclases (descrito en el capítulo 3) es de carácter general y similar a una aplicación web creada basándose solamente en Django y Python, sin ningún otro framework importante empleado. Dado que este prototipo se ha utilizado varias veces con éxito tanto en la propia facultad como en concursos con colegios, se decidió mantener la estructura y diseño original, cambiando solamente los componentes necesarios para arreglar los problemas existentes y entregar más funcionalidades a la aplicación, tales como un nuevo usuario, manejo de contraseñas y nuevas vistas, etc.

La implementación (explicada en el capítulo 4) de los nuevos componentes fue hecha de

tal forma que no se rompiera el flujo normal del software Videoclases, logrando que se pueda seguir usando como previamente se ha hecho tanto en cursos del departamento de computación como en actividades con colegios si se llegasen a hacer en algún futuro. La creación de un nuevo tipo de usuario Organizador ayuda a instituciones como el Ministerio u otro tipo de establecimientos a tener toda la información correspondiente de las actividades o concursos que se estén realizando en una sola página, pudiendo ingresar a ver los detalles de cada uno y hasta modificarlos si es necesario. Además, que los alumnos puedan recuperar su contraseña sin necesidad de contactar a algún usuario con poder de administrador, agiliza mucho este proceso. Con respecto a los criterios de evaluación de las actividades, al analizar la plataforma actualizada se dio cuenta de que este componente funciona correctamente y es extensible por naturaleza, por lo tanto, no fue necesario agregar alguna funcionalidad que permitiera agregar más de ellos dado que la aplicación web ya de por sí acepta el número de criterios y escala de evaluación que uno necesite. También, la misma escala a usar es modificable desde la administración de Django sin necesidad de programar algo extra. Finalmente, la vista del usuario organizador permite tener una visión general de todas las tareas y actividades de las que está a cargo, pudiendo incluso editarlas si es necesario, logrando así que este por encima de un profesor en temas de información.

Para terminar, con respecto a la validación, (descrita en el capítulo 5) a pesar de que se planeaba realizar una actividad de prueba en conjunto con el Ministerio de Transportes y un colegio (lo cual no se pudo por temas de tiempo y burocracia ministerial), se logró validar Videoclases con profesores de distintas áreas utilizando una encuesta de usuario final y también con funcionarios del Ministerio durante una demostración realizada en una reunión en conjunto con el profesor guía. La validación con los funcionarios fue de carácter simple, usando un método de observación y algunas preguntas mientras el alumno les mostraba la página web actualizada, logrando que entregaran sus apreciaciones e ideas que se podrían implementar a futuro. La validación mediante el formulario también permitió al alumno realizar pruebas cercanas a la realidad para así buscar posibles errores o ideas que podrían ser útiles para un trabajo a futuro.

En la siguiente sección se discutirá sobre los potenciales problemas con el trabajo hecho y la plataforma.

6.2. Discusión

Para empezar, el objetivo general (principal) planteado en la sección 1 fue cumplido, pero no de forma completa por algunas razones. La plataforma Videoclases si fue actualizada y se extendió su funcionamiento agregando nuevas componentes y funcionalidades a ella, pero en el proceso de validación no se logró lo estipulado. No se logró realizar un piloto en conjunto con el Ministerio de Transportes (en esta misma sección se explicará con más detalles). Con respecto a los objetivos específicos:

1. Se lograron actualizar las versiones de las tecnologías utilizadas en el prototipo, específicamente el framework (Django) y la versión del motor de base de datos.

2. Se validó el modelo de datos tanto antes como después de la actualización de la versión

del motor y de las entidades creadas para este trabajo.

3. Se logró una actividad con el Ministerio de Transportes en donde analizaron y revisaron Videoclases, pero no fue como se esperaba al principio del semestre. Más que actividad, podríamos decir que fue una reunión extendida.

4. Se creó un nuevo tipo de usuario (Organizador) pensando en instituciones o autoridades distintas a profesores.

5. No se implementó un nuevo sistema de evaluación debido a que las componentes actuales ya son lo suficientemente capaces para ser editadas según las necesidades de quien lo utilice.

6. No se crearon nuevas plantillas para distintas evaluaciones debido a que el trabajo de título se enfocó en la creación del nuevo usuario y la actualización de las tecnologías ya utilizadas, además de algunos cambios para mejor la calidad de uso de la aplicación. Además, como se mencionó en el punto anterior, las componentes en sí mismas son editables dependiendo del caso.

7. Se implementó una vista de administración para autoridades, la vista para los organizadores, la cual permite una visión general de las actividades que está llevando a cabo. También se modificó la vista de profesores, arreglando ciertos errores que existían en el prototipo original y otros que surgieron al actualizar las tecnologías.

La plataforma sigue teniendo un carácter de prototipo debido a que no se están utilizando las últimas versiones de las tecnologías presentes en Videoclases. Esto provoca que aún existan fallos en la optimización de ciertas vistas y que algunas cosas básicas como la creación de usuarios aún se tengan que hacer desde la administración de Django y no de forma personal. Si se hiciera una actualización de los frameworks usados, se podrían llegar a romper ciertas funciones base de la aplicación. Para dar un ejemplo, durante este trabajo, al pasar a una versión más reciente de Django y de algunas librerías de **JavaScript** el selector de fechas para crear actividades (fecha de entrega y fecha de revisión) dejo de funcionar completamente. Una componente muy simple, pero de suma importancia para una función básica de la aplicación.

Viendo el código también pueden existir problemas de compatibilidad, dado que, con el pasar de los años, varias funciones de Django escritas como métodos en Python fueron cambiando, llegando algunas a ser deprecadas, por lo que existe una mezcla entre código actualizado y código con métodos desactualizados.

Con todos estos antecedentes, la mejor solución a futuro si es que se quiere seguir trabajando en este proyecto es realizar Videoclases desde cero. Armando un nuevo diseño con las tecnologías actuales para así tener una implementación moderna y a prueba de errores de compatibilidad.

Con respecto a la validación, este proceso debió haber empezado mucho antes del semestre en donde se desarrolló el trabajo de título, ya que, por temas de tiempo, es muy difícil trabajar con colegios asociados al Ministerio de Transportes, dado que hay que hacer un todo proceso legal que permita al Ministerio introducir una nueva herramienta a los establecimientos, aunque sea a modo de prueba. Por esto, no se logró efectuar una validación con un caso

idóneo para la plataforma. A pesar de esto, se logró validar la aplicación web con profesores y funcionarios públicos perteneciente al Ministerio, pudiendo analizar las funcionalidades y la utilidad del trabajo llevado a cabo durante el semestre. También el uso del formulario ayudo en el proceso de validación tanto de la usabilidad de la plataforma como en la revisión de errores de la misma en un ambiente un poco más cercano a la realidad.

En la siguiente sección se comentará sobre el trabajo futuro en la plataforma Videoclases.

6.3. Trabajo Futuro

Durante el semestre en donde se trabajó en Videoclases para el desarrollo de este trabajo de título, existieron muchas ideas para el diseño y la implementación. También, gracias al proceso de validación, se formularon aún más. Dado que ya se tenía definido un cronograma junto con plan de trabajo correspondiente, se fueron anotando y guardando como posible trabajo a futuro en la plataforma, ya sea por el mismo alumno como por alguien que tome el proyecto nuevamente. Las ideas para el trabajo futuro son las siguientes:

- Actualizar el método de creación de cuentas para profesores y organizadores. Se plantea esto dado que actualmente la única forma de generar un nuevo usuario para ambos es mediante el sitio de administración de Django, provocando que, si una persona quiere usar la plataforma, debe obligatoriamente contactarse con las personas a cargo del proyecto para poder crearle una cuenta. Este registro de usuarios agilizaría de gran manera el proceso de uso inicial de Videoclases.

- Mejorar la creación de cursos debido a la contraseña genérica que se les da a cada alumno. La aplicación al momento de crear un curso pide al profesor que se adjunte una planilla Excel con la información de los alumnos junto con una contraseña para cada uno. El problema de esto último es que el docente tendrá dos opciones: Generar la misma clave para todos y que los alumnos la modifiquen al entrar a la plataforma (así está pensado este proceso en Videoclases) o, generar contraseñas aleatorias para cada alumno y enviarles un correo con sus datos. Este último proceso podría tomar mucho tiempo, por ejemplo, si es que se utiliza la plataforma para un concurso en conjunto con algún Ministerio y los cursos son de más de 40 alumnos. La posible solución a este problema podría ser que, al momento de crear el curso, la misma plataforma les enviara un correo con su contraseña personal para acceder al sitio web y poder modificarla a su gusto.

- Actualizar el diseño visual de la plataforma de forma general. Dados los resultados y comentarios extraídos de la validación, un punto criticado fue lo visual de Videoclases, llegando a confundir o ser monótono. Como trabajo futuro se podría rehacer las interfaces visuales en ciertas vistas para lograr una página web más llamativa.

- Empaquetar la aplicación mediante el uso de Docker. Así, el despliegue sería mucho más sencillo y no dependería de la plataforma en donde se haga, ya que el mismo contenedor de Docker contendría toda la información necesaria.

- Conectar un servicio de mensajería público, como el de Amazon (AWS), para evitar usar el correo personal y estar expuesto a posibles robos y ataques informáticos.
- Validar Videoclases con un caso de uso real, trabajando con alumnos, profesores y un organizador, pudiendo probar el software en su totalidad, extrayendo aún más comentarios e ideas para la plataforma.
- Utilizar herramientas para lograr un testing robusto de la aplicación.
- Continuar el trabajo en conjunto con el Ministerio de Transportes planteado en la Sección 5.1. Demostraron mucho interés en la aplicación web y mencionaron la idea de utilizarla de forma interna para las capacitaciones de los funcionarios. Lo vieron como una herramienta muy útil y prometedora, tanto como para propio Ministerio como los demás.

Como se mencionó en el Capítulo 4: Implementación, Videoclases es de código abierto y la última versión está en un repositorio público de GitHub [14]. Se espera que este proyecto pueda seguir siendo desarrollado, ya que es un muy buen aporte a modo de herramienta para la educación.

Bibliografía

[1] Jorge Ampuero. Pagina videoclases. https://videoclases.teachingislearning.cl. Last accesed 2022-12-01.

[2] Jorge Ampuero. Repositorio videoclases. https://github.com/Videoclases/videoclases. Last accesed 2022-03-20.

[3] Jorge Ampuero. Videoclases wiki. https://github.com/Videoclases/videoclases/wiki, 2016. Last accesed 2022-07-01.

[4] Jérémy Barbay. Pagina web proyecto teaching is learning. https://www.teachingislearning.cl/blogs/ano-2016. Last accesed 2022-03-15.

[5] Jérémy Barbay, Jocelyn Simmonds, Adriana Keiko Nishida, and Monael Pinheiro Ribeiro. "Teaching is Learning": Pedagogical material created and evaluated by students. In *Proceedings of the International Conference Frontiers in Education*, 2016.

[6] Diario Financiero Claudia Saravia. Sector bancario será la industria con mayor inversión en la nube en 2023 en la región. 2022. Last accesed 2022-12-10.

[7] Huawei Cloud. Pagina de huawei cloud services. http://119.8.151.56/. Last accesed 2022-12-18.

[8] CONASET. Informe de jovenes. https://www.conaset.cl/wp-content/uploads/2021/09/Jvenes-2020.pdf, 2020. Last accesed 2022-03-25.

[9] FileZilla. Pagina solución ftp filezilla. https://filezilla-project.org. Last accesed 2022-06-10.

[10] Stanley Frager and Carolyn Stern. The reading teacher. pages 407–417, 1970. Last accesed 2022-03-20.

[11] INJUV. Problemáticas y desafíos de las juventudes en chiles. https://www.injuv.gob.cl/sites/default/files/injuv2021_problematicas_y_desafios_de_las_juventudes_en_chile_-1.pdf. Last accesed 2022-03-25.

[12] Microsoft. Pagina principal de visual studio code. https://code.visualstudio.com. Last accesed 2022-03-15.

[13] Gustavo Silva. Formulario de validación para videoclases. https://forms.gle/4ycCxfpqeZBG2bWZ7. Last accesed 2022-12-17.

[14] Gustavo Silva. Repositorio videoclases versión actualizada. https://github.com/DawnFall10/VideoClases. Last accesed 2022-12-10.

Capítulo 7

ANEXOS

En este Anexo se añade material adicional requerido para apoyar las explicaciones realizadas en las partes principales de este reporte. Se incluyen cosas como el diseño de la base de datos (Seccion A.1.1), algunos extractos de código (Seccion A.2) y el formulario de validación utilizado junto con las respuestas literales sin la información personal de los usuarios.

Anexo A

Modelo de Datos y código

A.1. Diseño

A.1.1. Esquema de la base de datos

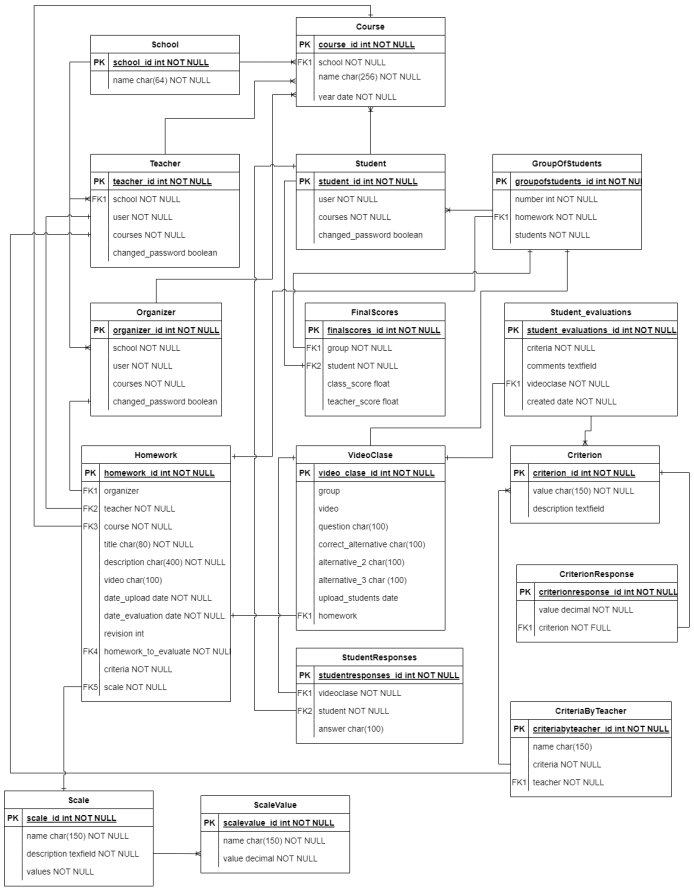

Figura A.1: Diagrama de tipo entidad-relación de la base de datos de la aplicación.

A.2. Implementación

A.2.1. Extractos de código

```

STATIC_ROOT = os.path.join(BASE_DIR, "site_media" ,"static")
STATIC_URL = "/static/"
#STATIC_URL = "~gsilva/static/"
STATICFILES_DIRS = [
    os.path.join(PACKAGE_ROOT, "static"),
]

FIXTURE_DIRS = [
    os.path.join(PROJECT_ROOT, "fixtures"),
]

# Simplified static file serving.
# https://warehouse.python.org/project/whitenoise/
# STATICFILES_STORAGE = 'whitenoise.django.GzipManifestStaticFilesStorage'

AUTHENTICATION_BACKENDS = ("django.contrib.auth.backends.ModelBackend",)
#LOGIN_REDIRECT_URL = "~gsilva/"

CRISPY_TEMPLATE_PACK = "bootstrap3"

INTERNAL_IPS = "127.0.0.1"

#FORCE_SCRIPT_NAME = '/~gsilva'
```

Código A.1: Extracto del archivo de configuraciones de Django.

```
...
class TeacherView(TemplateView):
    template_name = 'teacher.html'

    def get_context_data(self, **kwargs):
        context = super(TeacherView, self).get_context_data(**kwargs)
        current_year = timezone.now().year
        teacher = self.request.user.teacher
        context['homeworks'] = Homework.objects.filter(course__teacher=
    teacher)
        context['courses_without_homework'] = teacher.courses.filter(
    course_homework=None)
        return context

    @method_decorator(user_passes_test(in_teachers_group, login_url='/'))
    def dispatch(self, *args, **kwargs):
        return super(TeacherView, self).dispatch(*args, **kwargs)
...
```

Código A.2: Vista para los profesores. Entrega los datos a la plantilla para su renderizado.

```
{% extends 'base.html' %}
{% load static %}

{% block teacher-active %}class="menu-top-active"{% endblock %}
```

```
5
6  {% block extra-head %}
7  <script type="text/javascript">
8      function deleteTarea(homeworkId) {
9          if (confirm('  Seguro   que deseas borrar esta tarea?')) {
10
11         }
12     }
13 </script>
14 {% endblock %}
15
16 {% block body-header-buttons %}
17     <div class="col-xs-4">
18         <button onclick="location.href = '{% url 'new_course' %}';" class=
       "btn btn-info">Crear curso
19         </button>
20     </div>
21     <div class="col-xs-4 ">
22         <button onclick="location.href = '{% url 'new_homework' %}';"
       class="btn btn-info">Crear
23             tarea
24         </button>
25     </div>
26     <div class="col-xs-4 ">
27         <button onclick="location.href = '{% url 'new_conceptual_test'
       %}';" class="btn btn-info">Crear
28             Test Conceptual
29         </button>
30     </div>
31 {% endblock %}
32 {% block body %}
33             <div>
34                 <table id="example"  class="table table-responsive table-
       bordered" cellspacing="0" width="100%">
35                     <thead>
36                         <tr>
37                             <th>Curso</th>
38                             <th>Profesor</th>
39                             <th>Tareas</th>
40                             <th>Estado de<br> tarea</th>
41                             <th>Test Conceptual </th>
42                             <th>VideoClases<br> subidas</th>
43                             <th>Fecha de<br> entrega</th>
44                             <th>Fecha de<br> evaluacion</th>
45                             <th></th>
46                         </tr>
47                     </thead>
48                     <tbody>
49                         {% for t in homeworks %}
50                             <tr>
51                                 <td class="table-clickable" onclick="
       location.href = '{% url 'course' t.course.id %}';"><a>{{ t.course.name
       }}</a></td>
52                                 <td>{{ t.teacher.user.get_full_name }}</td
       >
53                                 <td class="table-clickable" onclick="
```

```
    location.href = '{% url 'videoclases_homework' t.id %}';"><a>{{ t.title
     }}</a></td>
                                    <td>{{ t.get_estado_nombre }}</td>
                                    {% if t.pedagogicalquestions %}
                                    <td  class="table-clickable" onclick="
    location.href = '{% url 'pedagogical_questions' t.pedagogicalquestions.
    id %}';">
                                        <a>{{ t.pedagogicalquestions|
    truncatechars:15|default:'No posee test a n '}}</a></td>
                                        {% else %}<td>No posee test a n</td>
                                    {% endif %}
                                    <td>{{ t.get_uploaded_videoclases }} / {{
    t.groups.count }}</td>
                                    <td>{{ t.date_upload|date:'d-m-Y' }}</td>
                                    <td>{{ t.date_evaluation|date:'d-m-Y' }}</
    td>
                                    <td><form action="{% url 'delete_course'
    %}" method="post">
                                            {% csrf_token %}
                                            <input name="homework" style="
    display: none" value="{{t.id}}" />

                                            <input type="submit" value="Borrar
    " class="btn btn-danger" onclick="deleteTarea({{t.id}})" />
                                        </form>
                                    </td>
                                </tr>
                            {% endfor %}
                            {% for c in courses_without_homework %}
                                <tr>
                                    <td class="table-clickable" onclick="
    location.href = '{% url 'course' c.id %}';"><a>{{ c.name }}</a></td>
                                    <td>{{ c.student_set.count }}</td>
                                    <td>Sin tareas</td>
                                    <td></td>
                                    <td></td>
                                    <td></td>
                                    <td></td>
                                    <td></td>
                                    <td></td>
                                </tr>
                            {% endfor %}
                        </tbody>
                    </table>
                </div>
{% endblock %}

{% block extra-script %}
{% include '_aux/teacher-table.html' %}
{% endblock %}
```

Código A.3: Plantilla para renderizar la información entregada por la vista de profesor.

```
...

self.greaterThan = function(value, target) {
        var isValue = value !== undefined && value !== false;
```

```
        var isTarget = target !== undefined && target !== false;
        if (isValue && isTarget) {
            return value > target;
        }
        return false;
    };
...

 fd.append("date_upload", self.homework.date_upload());
            if (self.homework.date_upload().localeCompare(self.
   homeworkInitialData.date_upload()) !== 0) {
                mustSubmit = true;
            }

 fd.append("date_evaluation", self.homework.date_evaluation());
            if (self.homework.date_evaluation().localeCompare(self.
   homeworkInitialData.date_evaluation()) !== 0) {
                mustSubmit = true;
            }
...
```

Código A.4: Validación y guardado para las fechas de las tareas. La fecha de evaluación debe ser posterior a la de subida.

```
...
    const fd = new FormData();
        fd.append("description", self.homework.description());
        if (self.homework.organizer()) fd.append("organizer", self.
   homework.organizer());
        fd.append("course", self.homework.course());
        fd.append("revision", parseInt(self.homework.revision()));
        fd.append("title", self.homework.title());
        fd.append("video", self.homework.video());
        let criteria_arr = [];
        for(let c of self.criteria()){
            criteria_arr.push({name:c.name(),description: c.description()
   });
        }
        fd.append("scale",JSON.stringify({criteria: criteria_arr, scale:
   self.homework.type_scales()}));
...
```

Código A.5: Guardado de criterios y escalas para las tareas.

```
...
        if scale:
            try:
                scale = json.loads(scale)
                model = CriteriaByTeacher.objects.create(teacher=teacher,
   name=homework.full_name())
                model.save()
                for c in scale.get('criteria'):
                    print(c)
                    model.criteria.create(value=c.get("name"), description
   =c.get('description', ""))
                homework.scale = Scale.objects.get(id=scale.get("scale"))
                homework.criteria.add(model)
```

```
                homework.save()
...
```

Código A.6: Guardado de escalas para las tareas.

```
...
def password_reset_request(request):
  if request.method == "POST":
    password_reset_form = PasswordResetForm(request.POST)
    if password_reset_form.is_valid():
      data = password_reset_form.cleaned_data['email']
      associated_users = User.objects.filter(Q(email=data))
      if associated_users.exists():
        for user in associated_users:
          subject = "Password Reset Requested"
          email_template_name = "password_reset_email.txt"
          c = {
          "email":user.email,
          'domain':'127.0.0.1:8000',
          'site_name': 'Website',
          "uid": urlsafe_base64_encode(force_bytes(user.pk)),
          "user": user,
          'token': default_token_generator.make_token(user),
          'protocol': 'http',
          }
          email = render_to_string(email_template_name, c)
          try:
            send_mail(subject, email, settings.EMAIL_HOST_USER , [user.email], fail_silently=False)
          except BadHeaderError:
            return HttpResponse('Invalid header found.')
          return redirect ("password_reset/done/")
  password_reset_form = PasswordResetForm()
  return render(request=request, template_name="password_reset.html", context={"password_reset_form":password_reset_form})
...
```

Código A.7: Vista para la recuperación de contraseñas.

```
...
    path("password_reset", vv.password_reset_request, name="password_reset"),
    path('password_reset/done/', auth_views.PasswordResetDoneView.as_view(template_name='password_reset_done.html'), name='password_reset_done'),
    path('reset/<uidb64>/<token>/', auth_views.PasswordResetConfirmView.as_view(template_name="password_reset_confirm.html"), name='password_reset_confirm'),
    path('reset/done/', auth_views.PasswordResetCompleteView.as_view(template_name='password_reset_complete.html'), name='password_reset_complete'),
...
```

Código A.8: Rutas para cada pagina asociada a la recuperación de claves. Es importante notar que Django provee la solución a las últimas tres plantillas.

```
...
class OrganizerView(TemplateView):
```

```
    template_name = 'header.html'

    def get_context_data(self, **kwargs):
        context = super(OrganizerView, self).get_context_data(**kwargs)
        current_year = timezone.now().year
        organizer = self.request.user.organizer
        context['schools'] = organizer.school.all()
        context['homeworks'] = Homework.objects.filter(organizer=organizer
    )
        context['courses_without_homework'] = organizer.courses.filter(
    course_homework=None)
        return context

    @method_decorator(user_passes_test(in_organizer_group, login_url='/'))
    def dispatch(self, *args, **kwargs):
        return super(OrganizerView, self).dispatch(*args, **kwargs)

...
```

Código A.9: Vista de organizador. Define la página principal para ete tipo de usuario.

```
...
class EditHomeworkOrganizerView(UpdateView):
    template_name = 'edit-homework-organizer.html'
    form_class = EditHomeworkForm
    success_url = reverse_lazy('organizer')
    model = Homework

    @method_decorator(user_passes_test(in_organizer_group, login_url='/'))
    def dispatch(self, *args, **kwargs):
        return super(EditHomeworkOrganizerView, self).dispatch(*args, **
    kwargs)

    def get_context_data(self, **kwargs):
        context = super(EditHomeworkOrganizerView, self).get_context_data
    (**kwargs)
        homework = Homework.objects.get(id=self.kwargs['homework_id'])
        context['courses'] = self.request.user.organizer.courses.all()
        context['homework'] = homework
        context['types_scales'] = Scale.objects.all()
        context['homeworks'] = Homework.objects.filter(course__in=context[
    'courses']).exclude(id=homework.id)
        context['videoclases_recibidas'] = GroupOfStudents.objects.filter(
    homework=homework) \
            .exclude(videoclase__video__isnull=True) \
            .exclude(videoclase__video__exact='').count()
        return context

    def form_valid(self, form):
        self.object = self.get_object()
        self.object = form.save()

        criteria = self.request.POST.get('criteria', None)
        homework = Homework.objects.get(id=self.kwargs['homework_id'])
        teacher = homework.teacher
        if criteria:
            try:
```

```
                criteria = json.loads(criteria)
                groups_criteria = self.object.criteria.filter(teacher=
    teacher)
                if len(criteria) >0:
                    if groups_criteria.count() == 0:
                        model = CriteriaByTeacher.objects.create(teacher=
    teacher, name=self.object.full_name())
                        model.save()
                        filtered_criteria = [item for item in criteria if
    item.get('editable',None) and not item.get('id', None)]
                        for c in filtered_criteria:
                            model.criteria.create(value=c.get("name"),
    description=c.get('description', ""))
                        self.object.criteria.add(model)
                    else:
                        group = groups_criteria[0]
                        for c in criteria:
                            id = c.get('id', None)
                            editable = c.get('editable', False)
                            if id and editable:
                                original = group.criteria.filter(id=id)[0]
                                if c.get('deleted', False):
                                    original.delete()
                                else:
                                    original.value = c.get('name')
                                    original.description = c.get('
    description',"")
                                    original.save()
                            elif not id and editable:
                                group.criteria.create(value=c.get("name"),
     description=c.get('description', ""))

            except JSONDecodeError:
                pass
        result_dict = dict()
        result_dict['id'] = self.object.id
        return JsonResponse(result_dict)

    def get_object(self):
        obj = get_object_or_404(self.model, pk=self.kwargs['homework_id'])
        return obj
...
```

Código A.10: Código para la edición de tareas por parte de un organizador.

```
...
    self.checkFormErrors = function() {
        var errors = false;
        self.formErrors.removeAll();
        if (!self.homework.title()) {
            errors = true;
            self.formErrors.push("Debes ingresar t tulo a la tarea");
        }
        if (!self.homework.description()) {
            errors = true;
            self.formErrors.push("Debes ingresar descripci n a la tarea")
    ;
```

```
12          }
13          if (!self.homework.date_upload()) {
14              errors = true;
15              self.formErrors.push("Debes ingresar fecha de subida");
16          }
17          if (!self.homework.date_evaluation()) {
18              errors = true;
19              self.formErrors.push("Debes ingresar fecha de evaluaci n");
20          } else {
21              if (!self.greaterThan(self.homework.date_evaluation(), self.
    homework.date_upload())) {
22                  errors = true;
23                  self.formErrors.push("La fecha de evaluaci n debe ser
    posterior a la fecha de subida");
24              }
25          }
26          if(!self.model_criteria.scale()){
27                 errors = true;
28                  self.formErrors.push("Debes seleccionar una escale de
    evaluaci n");
29          }
30          for(let c of self.model_criteria.criteria()){
31              if(!c.name() || c.name().length < 4){
32                  errors = true;
33                  self.formErrors.push("Criterios no v lidos");
34                  break;
35              }
36          }
37          return errors;
38      };
39  ...
40
41  self.submitEditHomework = function() {
42          var fd = new FormData();
43          var mustSubmit = false;
44          var hasErrors = self.checkFormErrors();
45          if (!hasErrors) {
46              if (self.homework.title().localeCompare(self.
    homeworkInitialData.title()) !== 0) {
47                  mustSubmit = true;
48              }
49              fd.append("title", self.homework.title());
50              if (self.homework.description().localeCompare(self.
    homeworkInitialData.description()) !== 0) {
51                  mustSubmit = true;
52              }
53              fd.append("description", self.homework.description());
54              if (parseInt(self.homework.course()) !== parseInt(self.
    homeworkInitialData.course())) {
55                  mustSubmit = true;
56              }
57              fd.append("course", parseInt(self.homework.course()));
58              if (parseInt(self.homework.revision()) !== parseInt(self.
    homeworkInitialData.revision())) {
59                  mustSubmit = true;
60              }
```

```
61              fd.append("revision", parseInt(self.homework.revision()));
62              if (self.homework.title().localeCompare(self.
    homeworkInitialData.title()) !== 0) {
63                  mustSubmit = true;
64              }
65              fd.append("title", self.homework.title());
66              if (self.homework.video()) {
67                  if (self.homework.video().localeCompare(self.
    homeworkInitialData.video()) !== 0) {
68                      mustSubmit = true;
69                  }
70                  fd.append("video", self.homework.video());
71              } else {
72                  if (self.homeworkInitialData.video()) {
73                      mustSubmit = true;
74                      fd.append("video", "empty");
75                  }
76              }
77              if(self.homework.homework_to_evaluate() !== self.
    homeworkInitialData.homework_to_evaluate()) {
78                  mustSubmit = true;
79                  if(self.homework.homework_to_evaluate())
80                      fd.append("homework_to_evaluate", self.homework.
    homework_to_evaluate());
81                  else
82                      fd.append('homework_to_evaluate',self.id());
83              }
84
85              //criteria
86              fd.append("scale", self.model_criteria.scale());
87              if(self.model_criteria.scale !== self.model_criteria_initials.
    scale()){
88                  mustSubmit = true;
89              }
90              let editable_criteria = self.model_criteria.criteria().filter(
    d=>d.editable);
91              let original_criteria = self.model_criteria_initials.criteria
    ().filter(d=>d.editable);
92              let results_criteria = [];
93              for(let c of editable_criteria){
94                  if(c.id && c.id()){
95                      let original = original_criteria.filter(d=>d.id && d.
    id() === c.id())[0];
96                      if(c.name() !== original.name() || c.description() !==
     original.description()){
97                          results_criteria.push({id:c.id(),name:c.name(),
    description:c.description(),editable:c.editable()});
98                      }
99                  }else {
100                     //new result!
101                     results_criteria.push({name:c.name(),description:c.
    description(),editable:c.editable()})
102                 }
103             }
104
105             if(editable_criteria.filter(d=>d.id).length !==
```

```
    original_criteria.filter(d=>d.id).length){
106                 //must delete some criteria!
107                 for(let o of original_criteria.filter(d=>d.id && d.id())){
108                     if(editable_criteria.filter(d=>d.id() === o.id()).
    length === 0){
109                         //Deleted element :(
110                         results_criteria.push({id:o.id(),name:o.name(),
    description:o.description(),editable:o.editable(),deleted:true});
111                     }
112                 }
113             }
114 
115             if(results_criteria.length > 0){
116                 mustSubmit = true;
117                 fd.append("criteria", JSON.stringify(results_criteria));
118             }
119 
120             //end criteria
121             fd.append("date_upload", self.homework.date_upload());
122             if (self.homework.date_upload().localeCompare(self.
    homeworkInitialData.date_upload()) !== 0) {
123                 mustSubmit = true;
124             }
125             fd.append("date_evaluation", self.homework.date_evaluation());
126 
127             if (self.homework.date_evaluation().localeCompare(self.
    homeworkInitialData.date_evaluation()) !== 0) {
128                 mustSubmit = true;
129             }
130             if (mustSubmit) {
131                 $.ajaxSetup({
132                     beforeSend: function(xhr, settings) {
133                         if (!csrfSafeMethod(settings.type) && !this.
    crossDomain) {
134                             xhr.setRequestHeader("X-CSRFToken", csrftoken)
    ;
135                         }
136                     }
137                 });
138                 return $.ajax("/organizer/homework/" + self.id() + "/", {
139                     data: fd,
140                     type: "post",
141                     processData: false,
142                     contentType: false,
143                     success: function(response){
144                         $("#editar-group-form-submit").click();
145                     }
146                 });
147             } else {
148                 $("#editar-group-form-submit").click();
149             }
150         } else {
151             self.changeFormErrorsVisible(true);
152             $('html,body').animate({
153                 scrollTop: $("#top-form").offset().top},
154                 'slow');
```

```
155         }
156     };
157
158 ...
```

Código A.11: Extractos de código que valida el formulario de edición de tareas y que luego envía la información.

```
1  ...
2      path('', vv.IndexView.as_view(), name='index'),
3
4      path('organizer', vv.OrganizerView.as_view(), name='organizer'),
5
6      path('organizer/course/<int:course_id>/', vv.CourseOrganizerView.
       as_view(), name='course_organizer'),
7
8      path('organizer/videoclases_homework_organizer/(?P<homework_id>\d+)/',
       vv.VideoclasesHomeworkOrganizerView.as_view(),
9      name='videoclases_homework_organizer'),
10
11     path('organizer/homework/<int:homework_id>/',
12     vv.EditHomeworkOrganizerView.as_view(), name='homework_organizer'),
13
14     path('organizer/edit-homework-form/<int:homework_id>/',
15     vv.EditHomeworkOrganizerView.as_view(), name='
       edit_homework_organizer_form'),
16
17     path('organizer/homework-evaluations/<int:pk>/',
18     hw.HomeworkEvaluationsOrganizerView.as_view(), name='
       homework_evaluations_organizer'),
19 ...
```

Código A.12: Rutas para cada página asociada a los organizadores.

A.3. Validación

A.3.1. Formulario de Validación

El presente formulario tiene como objetivo probar y validar una página web realizada como Memoria para optar por el título de Ingeniero Civil en Computación.

La plataforma está pensada en la idea .^Aprender enseñando", permitiendo a profesores crear tareas para sus alumnos con una descripción y ciertos criterios a evaluar. Los alumnos, entonces, deben subir el enlace de un vídeo pedagógico (subido a YouTube) realizado por ellos de 5 a 10 minutos como respuesta a la tarea asignada por el docente. Luego de esto, los alumnos deben evaluar una cantidad definida de vídeos de sus compañeros según los criterios definidos por el profesor, logrando aprender de sus propios iguales.

Para realizar la validación del trabajo realizado en la plataforma es necesario realizar los siguientes pasos de no más de 15 minutos antes de responder las preguntas:

1. Entrar a Videoclases (click aqui) (se demora un poco la primera vez en cargar).
2. Opcional: Ver el video de presentación del proyecto.
3. Ingresar con el siguiente usuario: Validar, contraseña: demo123.
4. Revisar la Tarea de Prueba 1 ya creada a modo de prueba, haciendo click en su nombre. Luego volver presionando la pestaña Profesor.
5. Crear una Tarea llenando toda la información obligatoria correspondiente: Titulo, descripción, organizador, revisiones mínimas (mantener en 3), fechas límites para subir y evaluar (la segunda debe ser mayor que la primera).
6. Definir la escala de criterios y los mismos con una pequeña descripción. (El título debe ser de más de 4 letras). Pueden agregar los que quieran.
7. En los grupos de alumnos poner el botón Designar Aleatoriamente y bajamos hasta el final para apretar el botón Crear Tarea.
8. Ingresar a la Tarea, viendo que este todo ok (las respuestas de los alumnos estarán blanco) y luego marcar Ver detalle de tarea.
9. Revisar la información para luego marcar Editar tarea y modificar el Título de la tarea y las fechas de subida y evaluación. También modificar un criterio.
10. Guardar la tarea modificada con el botón al final.
11. Cerrar sesión.

Ahora, ingresaremos como un usuario de tipo Organizador, el cual puede ver todas las tareas de las instituciones a las que esta asociado:

1. Ingresar con el siguiente usuario: Organizador, contraseña: demo123.
2. Revisar alguna de las tareas ya creadas marcando su nombre.
3. Ver los detalles de la tarea y luego modificar el Título, fechas (la segunda debe ser mayor que la primera) y agregar un nuevo criterio con su título y descripción. Se debe mantener el mismo curso.
4. Guardar la tarea, revisando si los cambios hicieron efecto.
5. Cerrar sesión.

A.3.2. Resultados del formulario

Preguntas	Respuestas (escala de 1 a 5)							
El sitio web es fácil de navegar.	4	5	4	4	4	5	4	5
Es fácil encontrar la Información deseada	5	5	5	5	5	5	3	4
Los enlaces y botones son claramente identificados	3	5	5	4	5	5	5	5
Los enlaces y botones funcionan correctamente	4	5	5	5	5	5	5	5
Las páginas se cargan rápidamente (<30 segundos)	5	4	5	5	5	5	5	5
El uso de las imágenes es aceptable	4	4	5	5	3	5	3	3
El uso del color es aceptable	5	4	3	4	4	5	5	3
El diseño general del sitio es apropiado dado su uso.	5	5	5	4	4	5	4	5
La organización de la información del sitio es apropiada	4	5	4	5	4	5	3	5
El contenido del sitio es relevante para su propósito.	5	5	5	5	5	5	5	5
La interfaz (visual) del sitio es placentera	2	5	2	4	3	5	5	3
El sitio tiene todas las funcionalidades esperadas para realizar actividades docentes	4	3	5	5	5	5	5	5
El sitio tiene todas las capacidades esperadas para realizar actividades docentes	4	3	5	5	5	5	5	4
¿Cómo califica globalmente el sitio Web analizado?	4	4	5	4	4	5	4	4

Tabla A.1: Tabla con todas las preguntas y las respuestas del formulario de validación.

Aquí se mostraran las respuestas a las preguntas escritas:

- Justifique su calificación:
  1. Perfectible de manera visual. Botón de (i) no aplica su uso.
  2. El sitio web en general es muy cómodo de usar, es fácil crear nuevas tareas y modificarlas después. El diseño está acorde al objetivo de la aplicación.
  3. A pesar de que no se me hiciera cómoda visualmente la interfaz, cumple perfectamente su propósito y organiza muy bien la información, enlaces y botones que requeriría para crear una tarea, evaluación, etc.
  4. Creo que se puede pulir, agregando diseños más llamativos.
  5. Hay algunas cosas mejorables, pero en general es bueno y hace lo que tiene que hacer.
  6. Me pareció fácil de navegar, posiblemente por mis años de experiencia usando la plataforma U-Cursos.
  7. La página es fácil de ocupar, pero tal vez es muy enredada el cómo poder subir la información/tarea, una vez ya ingresado a la realización de la tarea, esta todo muy claro
  8. La página contiene todo lo que necesita para la función que debe cumplir, en ocasiones la distribución o el tamaño de ciertos componentes (como la nota) podrian ser mejores o más ajustados
- Escriba aquí errores que encontró en la página web.

1. Botón de (i) no aplica su uso.
2. Al momento de usar la aplicación no se encontró ningún error perceptible. Solo algunas sugerencias para una nueva versión (leer parte de sugerencias)
3. El color no siento que me llame la atención, siento que se hace un poco monótona, lo que lleva a que ver la interfaz en si de una sensación de "más de lo mismo".
4. No vi errores
5. .ªl hacer grupos aleatorios de 2 integrantes para la tarea, dejaba a alguien solo. habría sido bueno tener la opción de hacer grupos de 1 integrante más en vez de 1 menos en caso de que la cantidad de grupos no divida exactamente la cantidad de integrantes. Al editar la tarea, pareciera que se borraron las notas. No hay botón de "volver" abajo, habría sido bueno no tener que scrollear hacia arriba al poner las notas (especialmente en cursos grandes)"
6. Al principio me costó entrar. Al parecer la contraseña no me funcionaba, pero puede haber sido error de digitación mío.
7. Ninguno
8. Ninguno

- Sugerencias para una posible segunda versión:

1. Agregar porcentaje de ponderación y escala de dificultad. Además de las rubricas de las propias tareas/evaluaciones, para guiar a los y las estudiantes en sus mandados.
2. "1. Al momento de revisar una tarea los nombres de los alumnos no se notan muy bien. Poner una letra más grande (o en un formato que destaque más) el nombre del alumno para que sea más fácil identificar de quién es cada video. 2. Poner un buscador de nombres de alumnos en las tareas, para tener la opción de encontrar a un alumno en particular en vez de tener que buscar su video en la lista entera. 3. Cambiar los nombres de la categoría 5 criterios, en particular el "cumple muy parcialmente". 4. Dar la opción de mezclar las escalas en los objetivos a evaluar. Por ejemplo evaluar un objetivo con la escala de 3 categorías y otro con la de 5 categorías.
3. Sin respuesta
4. Sin respuesta
5. Al editar la tarea, no tener que definir fechas de nuevo si no que mantenga las anteriores por defecto. Botón de guardar todas las notas al final (puede que esté acostumbrada a eso por ucursos pero encuentro útil no tener que estar guardando cada cambio de a uno)
6. Quizás, que cada grupo diferente del curso que se agregue tenga un color de fondo distinto. Es un capricho mío porque soy muy visual en mi funcionamiento, pero no es tan importante.
7. Mejorar la visual del cómo crear las tareas, y tal vez un pequeño tutorial o explicación a los pasos a seguir
8. Lo dicho de los tamaños, al editar las tareas se pierden las fechas originales, ojala se guardaran

Anexo B

Imágenes y capturas adicionales

En este Anexo se añade material visual adicional para apoyar las secciones de diseño, implementación y validación pero que no fue incluido en los capítulos correspondientes por razones de brevedad. Cada figura incluye en su leyenda una breve explicación de lo que se muestra.

Figura B.1: Código de error 403 (Forbidden) mostrado en la consola del navegador web al ingresar a la página durante el primer deployment en Búho.

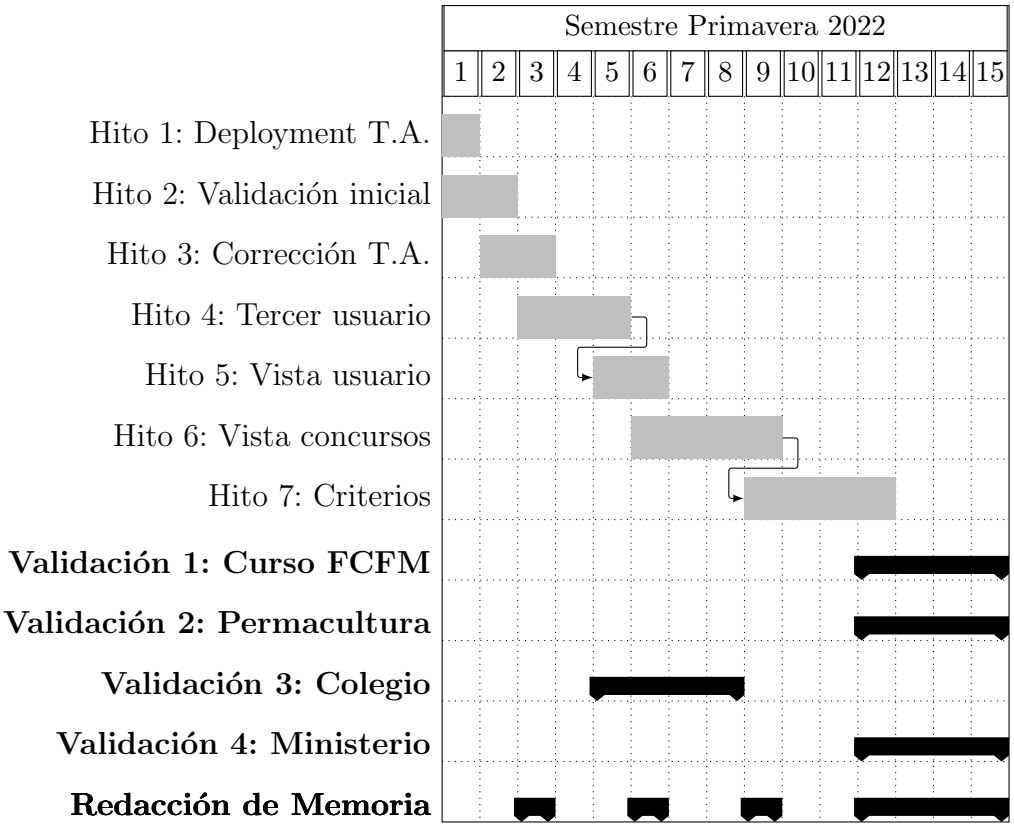

Figura B.2: Carta Gantt de planificación del proyecto. (T.A.: Trabajo adelantado)

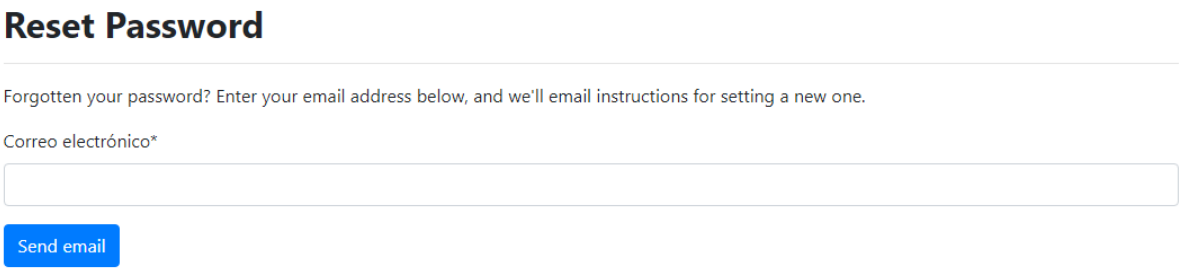

Figura B.3: Vista que recibe un correo para recuperar contraseña.

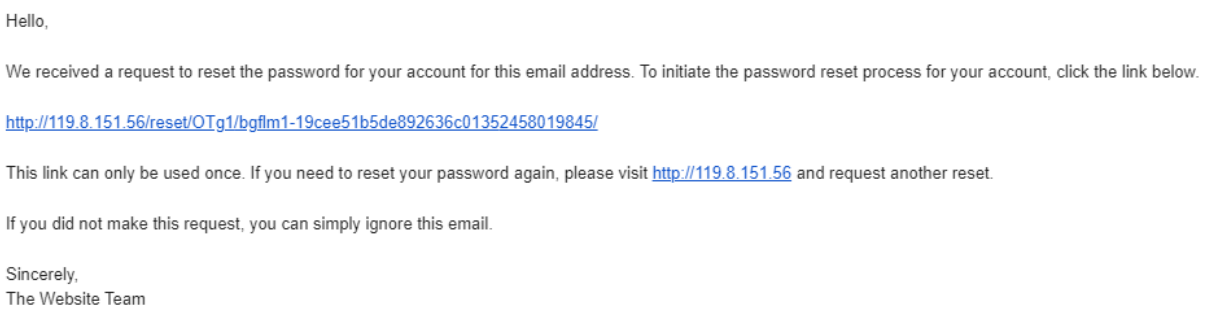

Figura B.4: Vista que permite el cambio de contraseña.

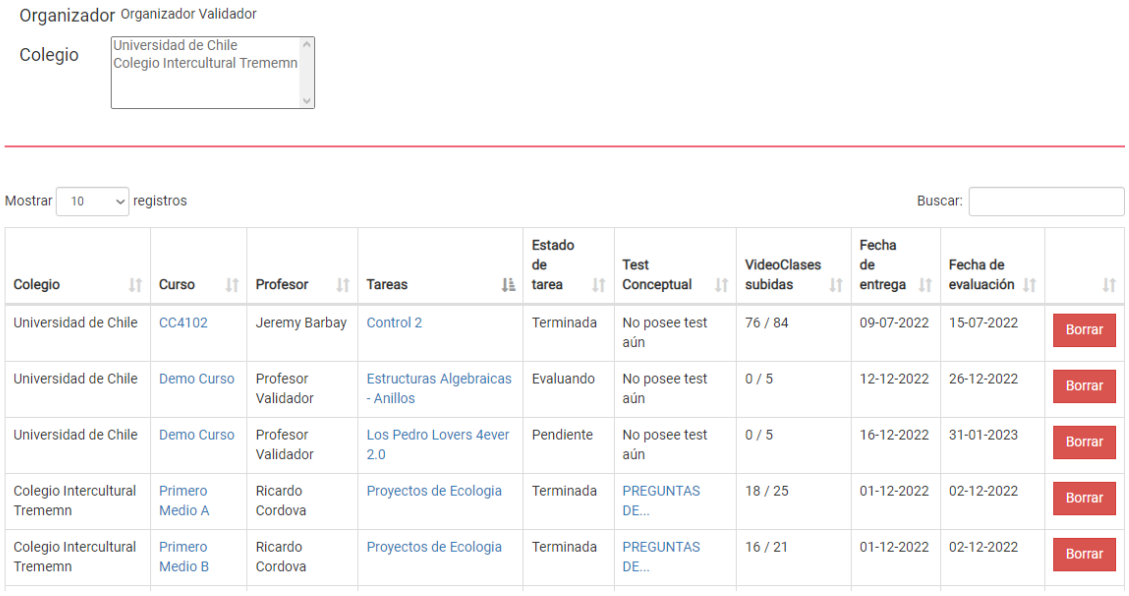

Figura B.5: Vista para los organizadores. Muestra más información que la de los profesores.

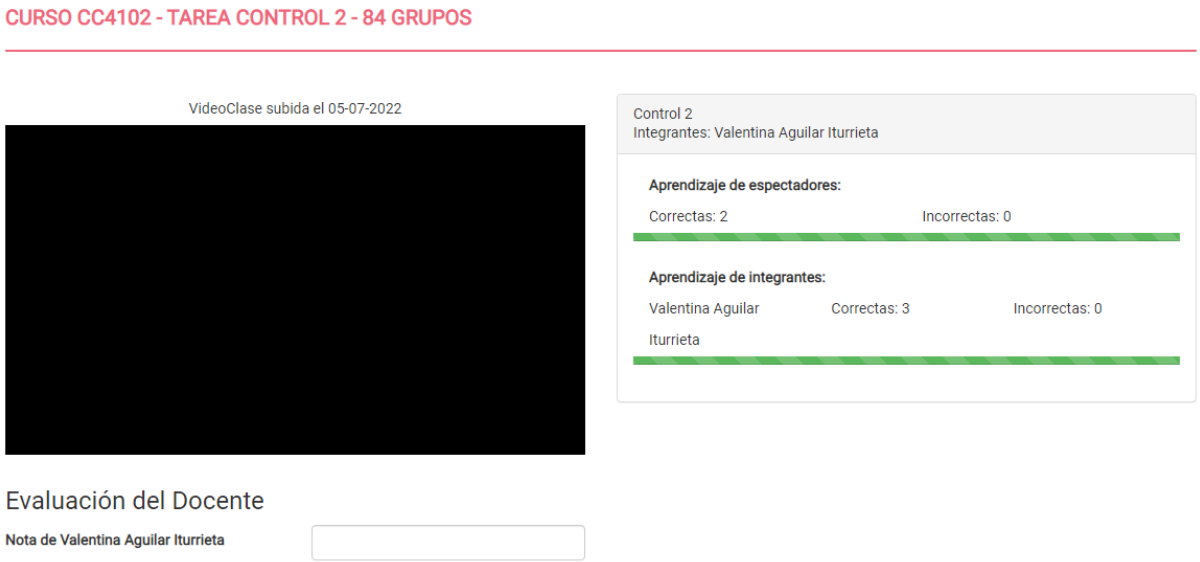

Figura B.6: Ejemplo de la vista de tareas. Se ve una cápsula de vídeo de un alumno junto con las respuestas correctas a las preguntas subidas.

Figura B.7: Pagina que muestra un curso presente en Videoclases.

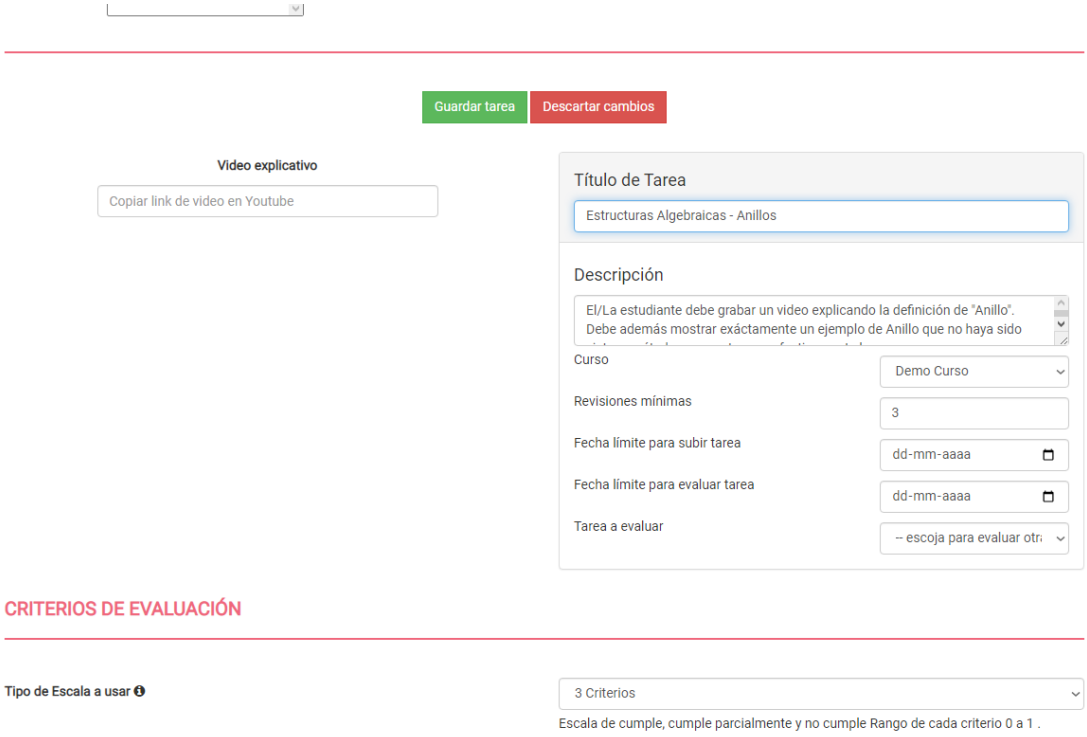

Figura B.8: Vista de revisión de los detalles de una tarea y edición de la misma.

Índice de Ilustraciones

Lista de Códigos