Webir

Facultad de Ingeniería 2015

Plataforma Informática para Planificación de Carreras

Integrantes

Alexander Berguer	4.500.321-9	aleberguer@gmail.com
Pablo Guartes	4.572.533-6	pabloguartes@gmail.com
Gonzalo Melgar	4.622.257-3	gmelgaba@gmail.com
Mauricio Piñatares	4.407.245-3	mpinatares@gmail.com

Índice

Introducción y Motivación

Hoy en día los estudiantes nos encontramos con una problemática muy común a la hora de comenzar un nuevo semestre, y ésta es saber que cursos tienen disponibles para poder cursar durante el mismo.

Actualmente el sitio web de bedelías brinda toda la información relacionada a las previaturas, pero a nuestro criterio, de forma poco amigable. En base a la poca claridad con la que son presentadas las previas de las asignaturas, decidimos desarrollar una aplicación web la cual permite visualizar de manera clara y sencilla cuales son las previaturas de las materias que realiza un estudiante, así como su progreso en la carrera en base a los cursos ya realizados.

El estudiante utilizará nuestro sistema apoyado en las funcionalidades de bedelías, pero permitiéndole realizar un seguimiento de su carrera estudiantil y poder planificar con más precisión los pasos a seguir.

Además este proyecto es de utilidad para utilizar técnicas de recuperación de datos en la web, diseño de base de datos, diseños de API’s web, desarrollo de algoritmos de carga de datos, algoritmos de búsqueda de datos y de frontend.

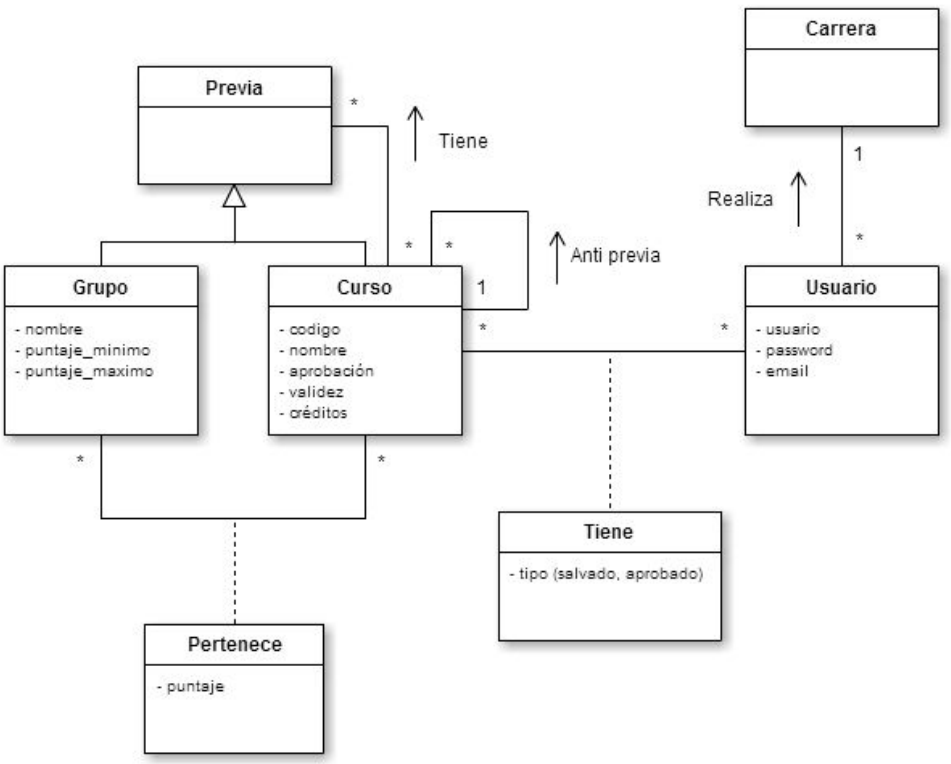
Otro de los aspectos por el cual decidimos realizar este proyectos, es motivar al estudiante a finalizar su carrera viendo su avance hasta el momento. De esta forma además de tener un seguimiento de su carrera, tiene la posibilidad de ver las materias restantes motiva a completarla.

Análisis del Problema

Esta etapa de análisis se comenzó con los elementos básicos para el desarrollo de un proyecto. Se estudiaron y discutieron profundamente los requerimientos, se planificaron plazos de todas las etapas futuras, y se comenzó con el desarrollo de piezas claves.

Se hizo un profundo estudio del sistema de previaturas de la UdelaR, para tener una idea más clara de lo que se podía lograr y cómo. Procedimos al estudio de cómo estaba la información presentada en la página de bedelías, cómo se consultaba y cómo podíamos obtenerla de manera que nos fuera de utilidad en las etapas siguientes.

A partir de esto, pudimos observar que la estructura de la información era bastante similar y consistente entre las facultades, lo cual pudo responder a la gran interrogante que teníamos antes de comenzar el proyecto que era la de llevar a cabo una plataforma para todas las universidades o sólo para Ingeniería. Con la estructura de los datos, nos aventuramos a implementarla para todas las universidades, salvo para las facultades de Derecho y Veterinaria, las cuales no tienen la información de previaturas disponibles online.
Luego, diseñamos un modelo de dominio el cual representa de forma genérica la realidad existente en las universidades y sus cursos.



(Modelo de Dominio)

En el diagrama se puede apreciar los distintos modelos que forman parte de nuestro sistema. Entre ellos están los usuarios, que representan los estudiantes de las distintas facultades. Dichos estudiantes, realizan una carrera, la cual se representa con una asociación. A su vez, cada estudiante puede tener varios cursos, dependiendo de la carrera que esté cursando, y estos cursos puede estar aprobados ya sea en modalidad de curso o de examen.

En cuanto al modelado de las previas, un curso puede tener como previa otro curso, o un grupo de cursos, en el cual debe de cumplir con cierto puntaje para poder cursar la asignatura. Es por esto que ésta realidad se modeló como una herencia, ya que una previa puede ser de tipo curso o tipo grupo, verificando condiciones distintas.

Otro dato importante en el modelado de la realidad son lo que denominamos *antiprevias*.
Las antiprevias son cursos o grupos de cursos, los cuales un estudiante no puede tener aprobados para cursar una asignatura, es decir esos cursos son excluyentes entre si, al tener un curso aprobado, no se puede cursar el curso que es antiprevia de él.

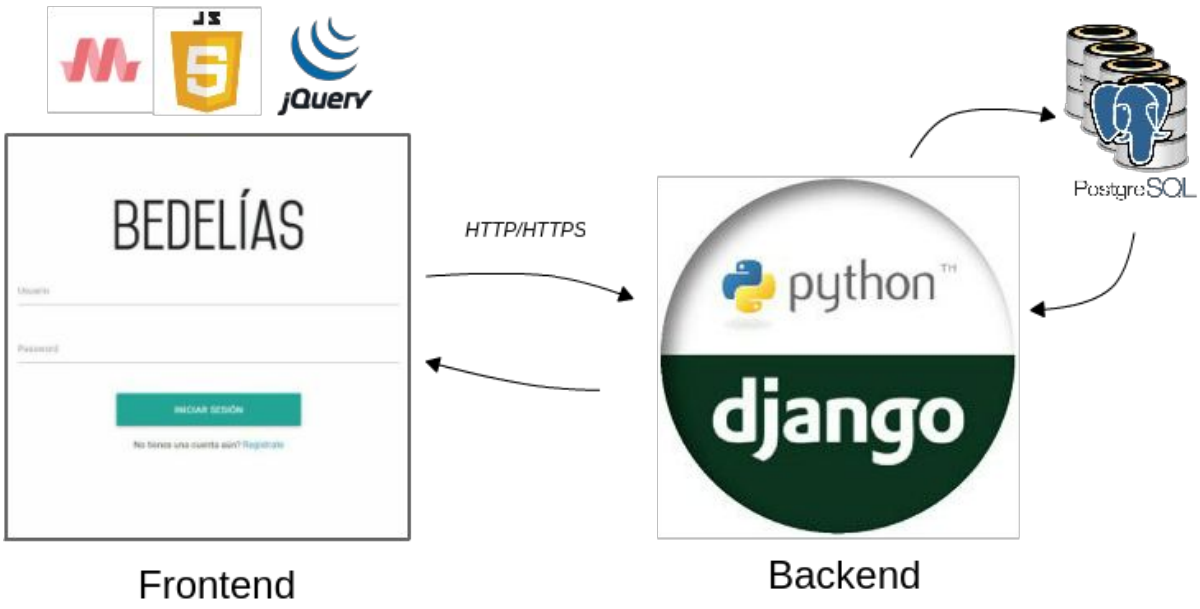
Las antiprevias son las que aparecen con un asterisco en la página de bedelías, y generalmente son cursos equivalentes de distintos planes o modalidades. Por ejemplo, si un estudiante cursó y aprobó Cálculo 1, no puede cursar Cálculo 1 anual, ya que Cálculo 1 es antiprevia de su correspondiente en modalidad anual.

Diseño de la Solución

Luego de la etapa de análisis, el siguiente paso de nuestro proyecto es el diseño del sistema. Decidimos realizar una plataforma web frente a implementar una aplicación móvil, ya que es accesible por cualquier persona que disponga de internet, ya sea en computadoras, smartphones, tablets etc, y así no restringir el uso dependiendo del dispositivo o sistemas operativos que el estudiante tenga. No obstante, el sistema diseñado hace pública una API, la cual puede ser la base a la implementación aplicaciones móviles en un futuro.

Con estas decisiones, básicamente se desarrollaron dos aplicaciones: una aplicación web diseñada para el uso del estudiante, y otra aplicación que maneja la lógica del sistema y las consultas a la base de datos (aplicaciones frontend y backend respectivamente).

A continuación se presenta un diagrama de la solución propuesta indicando cada componente y sus tecnologías a las cuales haremos referencia más adelante:



(Diagrama del diseño)

El **backend** es un proyecto encargado del manejo de la información, comunicándose directamente con la base de datos, y publicando los respectivos puntos de acceso para acceder a la misma. Básicamente, es una API que publica servicios que luego son consumidos por la aplicación web.

Dentro de las herramientas del backend utilizadas tenemos:

Para la implementación de la lógica de la aplicación se utilizó el lenguaje Python, en su versión 2.7, ya que es un lenguaje de fácil manejo, intuitivo y con una gran rapidez para implementar y ver resultados. Otra característica que fue determinante para utilizar Python fue la facilidad que ofrece para manipular los datos que recuperamos desde la web de bedelías.
Para poder utilizar python para aplicaciones web utilizamos el framework Django, en su versión 1.8.

Django es un framework web, que utiliza el patrón modelo-vista-controlador, que permite la creación de sitios web complejos utilizando python. Django trae facilidades a la hora del manejo de modelos y usuarios, así como también facilita el desarrollo, ya que ofrece herramientas como la implementación de un servidor local, y un facil manejo de la base de datos mediante comandos (migraciones, carga de datos, usuarios de bases de datos, roles etc.).

Sistema de gestión de bases de datos relacional orientado a objetos y libre. Se decidió utilizar esta base, debido a su fácil integración con el framework Django.

El **frontend** es el proyecto que le permite al usuario consumir los servicios publicado por el backend mediante pedidos HTTP, realizando las operaciones requeridas y desplegando los resultados al usuario. Se tuvieron en cuenta varios aspectos de usabilidad web, diseñando un sistema que se adapte funcionalmente a las necesidades del usuario, brindándole así un sitio amigable, disminuyendo el tiempo de aprendizaje de uso.

Dentro de las herramientas del frontend utilizadas tenemos:

Materializecss es un framework web front-end moderno y responsive basado en el esquema de diseño "Material Design" desarrollado por Google. Este framework es sencillo y fácil de usar, ideal para visualizar un sitio limpio y de forma amigable.

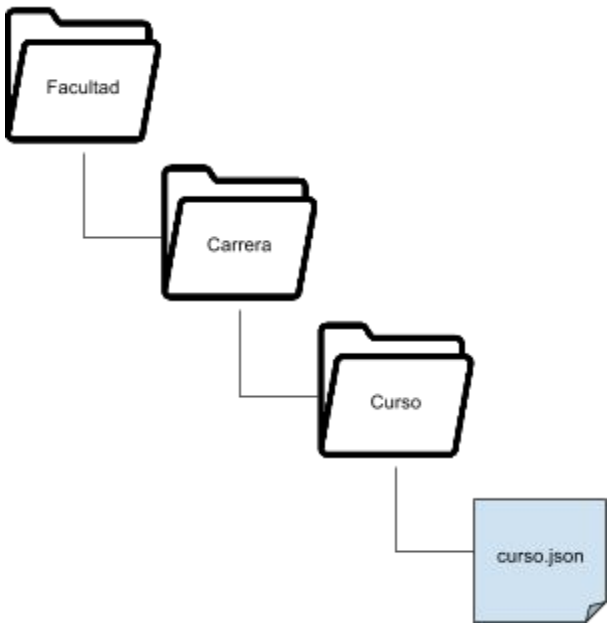
JQuery es una librería de JavaScript que permite simplificar la manera de interactuar con los elementos del DOM en una página HTML. Simplifica mucho a la hora de realizar pedidos HTTP y seleccionar la información de la página.
En esta librería fue que se basó fuertemente la comunicación de la aplicación frontend con la API ofrecida por el backend

Implementación

La implementación del sistema se dividió en dos grandes etapas: la etapa de scraping y la implementación del sistema web.

Etapa 1: Web Scraping

En esta primera etapa se realizó un estudio de la estructura de la página de bedelías de forma tal que fuera posible obtener la información relevante a la hora de realizar nuestro sistema. Dicha información fue persistida en archivos de tipo JSON, divididas en una estructura de directorios por niveles en los cuales el nivel más alto es la facultad, el siguiente la carrera dentro de la facultad seleccionada y dentro carpetas con los cursos, las cuales contienen la información del mismo (como por ejemplo, nombre, codigo, creditos, etc).



(Estructura de directorios de scraping)

Luego de realizar el scrapping de la información previamente mencionado, lo siguiente fue crear un parser que lea dicha información y la agregue a la base de datos de nuestro sistema. Básicamente nuestro parser recorre la estructura de directorios previamente mencionada insertando todas las carreras, cursos, etc; con su respectiva información permitiéndonos luego utilizar la información desde nuestra base de datos evitando consultar la información de la página de bedelías online cada vez que se ingresa al sistema.

Dentro de las herramientas utilizadas en esta etapa tenemos:

- *Selenium*: Es una herramienta orientada al testing automatizado de páginas web, pero por sus cualidades de navegación automática y obtención de datos resulta ideal para obtener la información de la página de bedelías obteniendo los nodos relevantes mediante XPath
- *Python*: La herramienta Selenium se puede usar en diversos lenguajes y decidimos usar python por su simpleza.
- *PhantomJS*: PhantomJS es un browser headless con soporte javascript que no renderiza en pantalla, por lo que aumenta la performance de selenium considerablemente por encima de los browsers convencionales.

Etapa 2: Implementación Frontend/Backend

La etapa 2 corresponde a la implementación de un sistema web que permita la utilización de la información obtenida en la etapa previa, para que el usuario pueda utilizar el servicio de forma eficiente. Se desarrollaron 2 proyectos, uno que contiene todo lo relacionado al frontend y otro proyecto backend que publica determinados puntos de acceso para consumir la información necesaria mediante pedidos http.

El frontend es un proyecto HTML, dividido en módulos por páginas para realizar los pedidos de manera independiente. Cada acción hace un pedido al backend, cumpliendo con lo pedido, para luego mostrar al usuario su progreso, entre otras cosas.

Dentro de las funcionalidades básicas del frontend se encuentran las siguientes:

- Registro/Login/Logout de usuario
- Guardar cursos y materias salvadas CRUD.
- Recomendaciones de materias a cursar.
- Resumen de créditos obtenidos (en caso de tener esta información disponible).
- Buscador de cursos con sugerencias.

El backend es un proyecto MVC implementado en Python Django, encargado de publicar los puntos de acceso necesarios para el frontend, accediendo a la información de la base de datos de forma eficiente. Básicamente se publican determinada cantidad de urls que van a ser consumidas desde el cliente frontend. También es la parte encargada de implementar los algoritmos de búsqueda de cursos habilitados para cursar, la definición de los modelos del framework y las consultas hacia la base de datos.

Para el manejo de la base de datos, como las consultas, el agregado y borrado de datos, etc. se utilizó el Object-Relational Mapper (ORM) provisto por django, el cual facilita enormemente el manejo de la base de datos sin tener que lidiar con sentencias SQL puras.

Django trabaja con Modelos, los cuales representan entidades de la realidad que se está implementando. Por ejemplo el modelo de Carrera en django es el siguiente:

```
class Carrera(models.Model):
    nombre = models.CharField(max_length=200)
    codigo = models.CharField(max_length=20)
    plan = models.CharField(max_length=200, blank=True, null=True)
    facultad = models.ForeignKey('Facultad', related_name='carreras')
```

Dado este modelo, el ORM de django se encarga de crear su tabla correspondiente en la base de datos, y mapearla con el modelo, así cada cambio que se hace en el modelo, se ve reflejado en la base, sin necesidad de realizar consultas a bajo nivel.
La tabla que Django crea para este modelo es la siguiente:

```
CREATE TABLE Carrera (
  "id" serial NOT NULL PRIMARY KEY,
  "nombre" character varying(200) NOT NULL,
  "codigo" character varying(20) NOT NULL,
  "plan" character varying(200),
  "facultad_id" integer,
  FOREIGN KEY (facultad_id)
      REFERENCES facultad (id) MATCH SIMPLE
)
```

Django se encarga también de crear índices sobre las claves foráneas para optimizar la búsqueda

```
CREATE INDEX Carrera_00e3be6f
  ON Carrera
  USING btree
  (facultad_id);
```

Django permite definir diferentes todos los tipos de relaciones entre modelos, los cuales se mapean con relaciones en la base de datos. Dichas relaciones son por ejemplo, foreign key, one-to-one, one-to-many, many-to-many etc.

En cuanto a las operaciones en la base de datos, consultas, guardado y borrado de datos, Django provee operaciones para facilitar su uso, ejemplo de una inserción de una Carrera:

```
carrera = Carrera()    //se define un objeto carrera
carrera.codigo = cod   //se agregan los atributos
carrera.facultad = fac
carrera.plan = plan
carrera.save()         //guarda la instancia como una nueva fila en la
tabla
```

Este fragmento de código, internamente hace un INSERT en la tabla carrera con los valores dados.
Un ejemplo de consulta en Django puede ser la siguiente, que retorna las carreras de una facultad.

```
carreras = Carrera.objects.filter(facultad=facultad_id)
```

Que lo que hace internamente es la siguiente consulta sql.

```
SELECT * FROM Carrera WHERE facultad_id = 'facultad_id';
```

Con estas operaciones, se nos hizo mucho más amena la tarea de comunicación con la base de datos, evitando hacer consultas sql extensas y menos entendibles.
En base a estas operaciones, se implementaron todas las funcionalidades de búsqueda de cursos, así como también las asociaciones de un estudiante con los cursos aprobados y salvados.

Problemas Enfrentados

Durante el desarrollo del sistema el equipo se encontró con dificultades de diversa índole y de considerable complejidad, impidiendo el avance del proyecto a un ritmo eficiente.

Entendimiento de la metodología de las previas de las universidades:

Si bien como estudiantes estamos acostumbrados a revisar la página de bedelías para determinar qué asignaturas podemos cursar o no, debíamos entender a fondo cómo funcionaba el sistema de previaturas para lograr llevar a cabo este sistema.

Tiempo de ejecución de la técnica de scrapping:

Para realizar el scrapping completo de previas se deben de hacer dos recorridas, una para previas y otra para créditos. Esto para cada materia de cada carrera y de cada facultad se hace muy costoso, aunque el scraper tuvo una buena performance los tiempos son bastante altos, cerca de 20 horas de scraping con 4 threads activos.

Generalización del sistema para todas las universidades:

Si bien el sistema de previas es similar, hay facultades que adaptaron sus previas a este sistema de créditos de tal manera que los alumnos deban tener todo un año aprobado para cursar las del siguiente año, lo que ocasionó que el scraping de cada materia para esas carreras sea de un tiempo muchísimo mayor con respecto a otras facultades y los archivos json generados también un tamaño considerablemente mayor (64KB contra 3 KB)

Información problemática:

Algunos de los campos de los cuales extrajimos la información en Bedelía incluían espacios, lo que ocasionó que se guardaran con ese espacio en la base de datos generando algunos problemas en el testing. Además existe una asignatura que posee un asterisco en su nombre, lo que provocó la caída del scraper que tengamos que manejar ese caso en el procesamiento.

Complejo modelado de datos

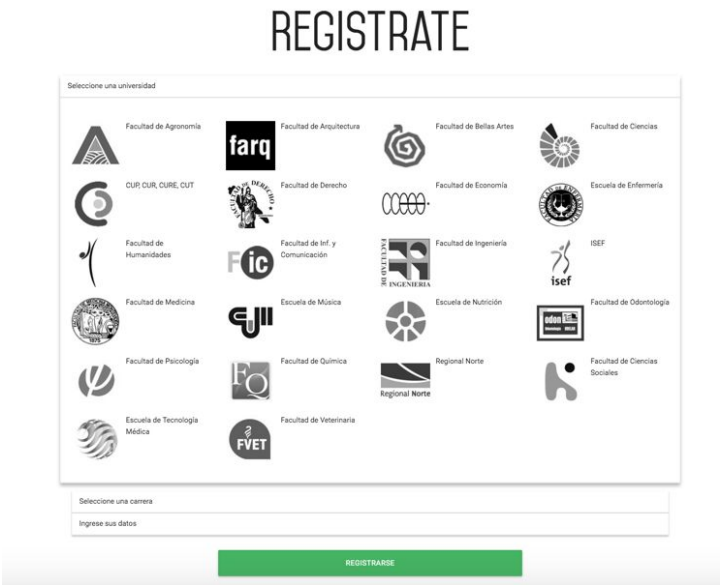
- Asignaturas con créditos mínimos en lugar de previas
- No todas las carreras tienen previas en la página
- No todas las carreras tienen los créditos de las materias como información disponible

Ilustración del Sistema

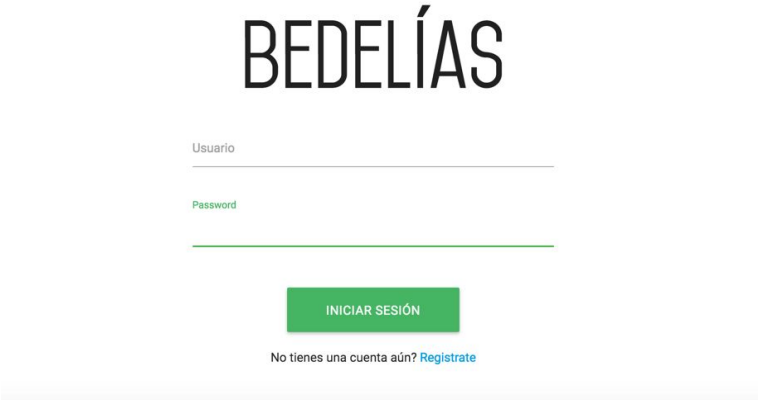
A continuación se presentan una serie de capturas de pantalla del sistema a modo ilustrativo para tener un pantallazo general de cómo es nuestro sistema desde el punto de vista del usuario, mostrando las funcionalidades principales con algún ejemplo.

Registro y Login de Estudiante

Para el registro de usuario, se implementó un wizard en el que se solicita los datos requeridos para un estudiante, como lo son la facultad a la cual pertenece y la carrera que está cursando, así como también los datos personales del estudiante.



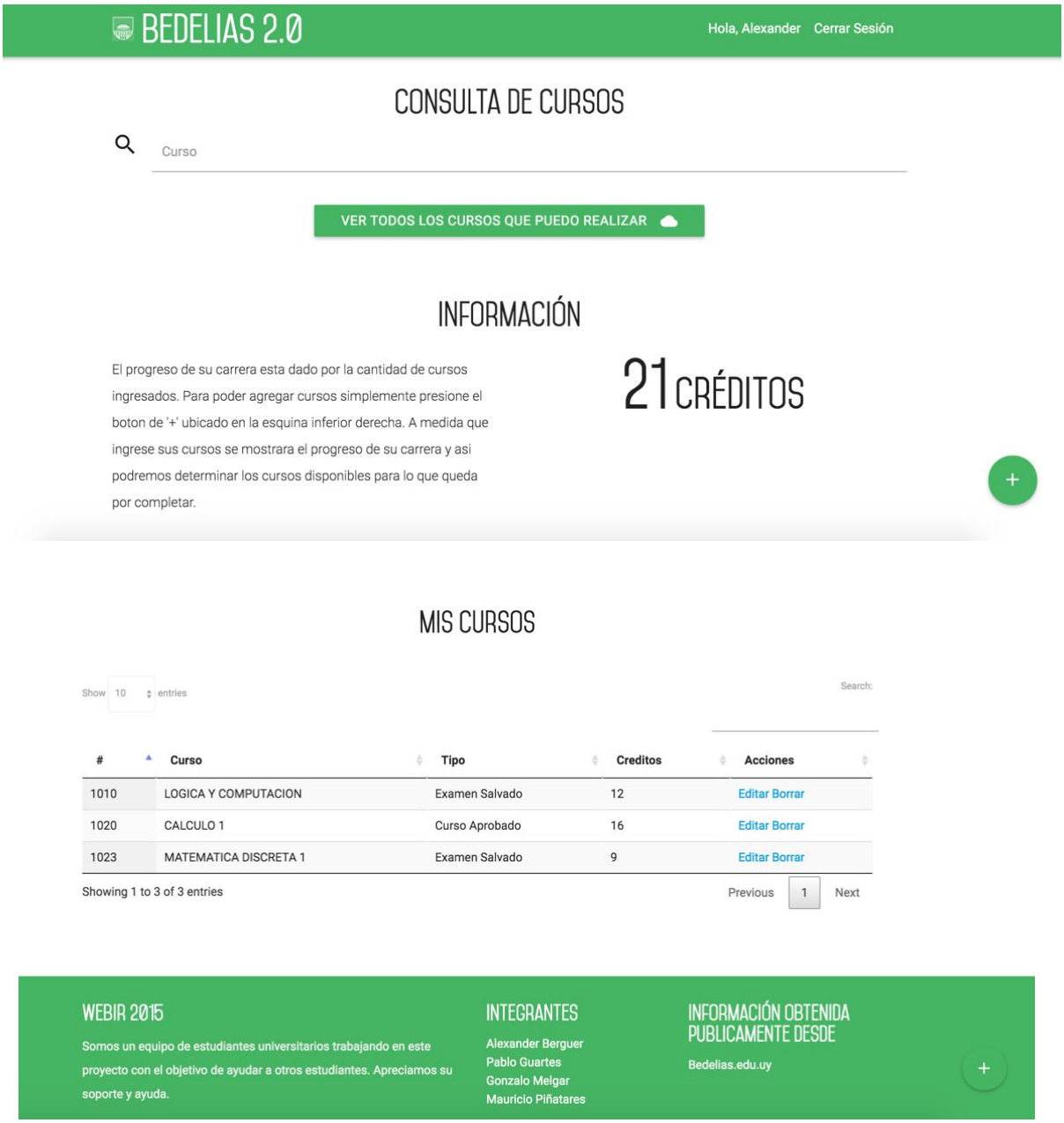
La funcionalidad de iniciar sesión la mantuvimos simple, minimalista.



Home

En la página inmediata al inicio de sesión, tenemos las funcionalidades principales de la plataforma, estas son, consultar información de los cursos, consultar los cursos a los cuales el estudiante está habilitado a cursar, un resumen de los créditos que tiene el estudiante, una tabla donde se visualizan los cursos que ha ingresado el usuario, ya sea que tenga el curso aprobado o el examen salvado.

A su vez, tenemos un botón flotante en la parte inferior derecha en donde se abre un modal en el cual el estudiante puede agregar cursos a su lista.



Cursos habilitados a cursar

Esta sección contiene los cursos que un estudiante puede cursar dependiendo de los cursos que agregó a su lista, teniendo en cuenta las previas de curso y examen, antiprevias, grupos de previas, puntajes etc.

Codigo	Nombre	Creditos	Validez
1022	CALCULO 2	16	20
1030	GEOMETRIA Y ALGEBRA LINEAL 1	9	20
1052	CALCULO 1 (ANUAL)	16	20
1053	GEOMETRIA Y ALGEBRA LINEAL 1 (ANUAL)	9	20
1120	FISICA GENERAL 1	13	20
1151	FISICA 1	10	20
1213	PLANIF.DE CLASES:DISEÑO DE UNID.DIDACT.	2	999
1223	CIENCIA, TECNOLOGIA Y SOCIEDAD	8	999
1224	ECONOMIA	7	20
1225	POLITICAS CIENTIFICAS EN INF.Y COMP.	3	999

Showing 1 to 10 of 27 entries Previous 1 2 3 Next

La tabla muestra el código y nombre de la asignatura, los créditos que otorga y la validez en meses del curso.

Trabajo a futuro

Por cuestiones más que nada de tiempo, no hemos podido implementar o mejorar algunas características del sistema. En esta sección se mencionan algunos aspectos del sistema a mejorar así como alguna característica que nos hubiera gustado agregar al mismo.

Actualización de información

Por motivos de consistencia de la información, un punto que se debería reforzar es el mantenimiento de la información de cursos, carreras, etc; para mantener la información desplegada al usuario consistente. Normalmente no son muchos las variaciones en la información pero por ejemplo, si un curso pierde su validez o modifica sus créditos, esta información debería de actualizarse en nuestro sistema.

Generar escolaridad

Nos resulta interesante poder generar la escolaridad del estudiante en base a la información ingresada, siguiendo el mismo formato que utiliza bedelía a la hora de entregarla mediante estilos HTML.

Seguridad de usuarios

Otro de los aspectos que nos gustaría mejorar es la seguridad de usuarios. Si bien alguna encriptación de contraseñas al almacenar los passwords en la base de datos sucede, nunca es bueno escatimar en este aspecto, por ejemplo mejorar acceso de endpoints, ocultar ids de usuarios en las url, entre otros.

Visualización de currícula y materias obligatorias.

Actualmente la información de la currícula de las distintas carreras no se encuentra centralizada en ninguna página web, por lo que no se pudo automatizar este trabajo y sería algo que habría que construir manualmente. A su vez hay ciertas materias obligatorias que para obtener títulos o títulos intermedios y no contamos con esa información ya que no se encuentra en la página oficial de bedelías.

Notificaciones por correo

A la hora de registrarse en el sistema, se debe ingresar determinada información personal, entre ella el correo electrónico. Nos hubiera gustado agregar como funcionalidad una notificación por correo cada determinado tiempo que indique tu progreso en la carrera, así como otra información relevante al usuario.

Deploy a producción

Nos resultó muy interesante la realización de este proyecto, y una de las tareas que nos gustaría concretar es poner en marcha nuestro sistema. Para poder realizarlo hay algunos problemas vitales que debemos solucionar, en especial la seguridad de nuestros usuarios. Tampoco descartamos una asociación con bedelías para aplicar nuestro motor en su página oficial.

Conclusión

Creemos que es de los pocos cursos a lo largo de la carrera que te permite proponer un proyecto propio, totalmente libre de restricciones tecnológicas, imponiendo de esta forma la toma de decisiones por lo que nosotros creemos que es la mejor opción para desarrollar un sistema de estas características.

Si bien la página de bedelías contiene toda la información necesaria para determinar qué previas se pueden cursar, la poca amigabilidad de la la página hace que esta sea una tarea engorrosa, y difícil de entender, es por esto que consideramos que nuestro proyecto es una de las posibles mejoras que se podrían incorporar al sitio web de bedelías, permitiendo así un fácil manejo y seguimiento de la actividad del estudiante.

Fue interesante aprender nuevas tecnologías tanto como para realizar la extracción de la información, como al momento de implementar los servicios.

Esperamos que la implementación de éste proyecto sea de ayuda para las posibles mejoras a futuro, permitiendo a los estudiantes facilitar lo más que se pueda a la hora continuar su carrera.