

# <Code Academy> Nivel 2

## Trabajo Final

### Condiciones de Aprobación:

Fecha límite de entrega: 20/12/2019.

Forma de entrega:

- El código debe estar en un repositorio de github.
- Al repositorio de github debemos tener acceso los profesores:
  - Nicolas Nappe ([nappe@ar.ibm.com](mailto:nappe@ar.ibm.com))
  - Ivan Kuzel ([kuzeliva@ar.ibm.com](mailto:kuzeliva@ar.ibm.com))
  - Diego Tabares ([tabares@ar.ibm.com](mailto:tabares@ar.ibm.com))
  - Nicolas Gomez ([nicgomez@ar.ibm.com](mailto:nicgomez@ar.ibm.com))
  - Nicolas Tcherechansky ([tchere@ar.ibm.com](mailto:tchere@ar.ibm.com))
  - Pablo Cruz ([cruzp@ar.ibm.com](mailto:cruzp@ar.ibm.com))
- Una vez listo para entregar, el link al repositorio debe ser enviado **por mail** a los profesores listados en el punto anterior, utilizando el asunto "Code Academy Nivel 2 – TP Final"

### Enunciado:

Los profesores del curso somos Product Owners que necesitamos una aplicación que cubra las necesidades de nuestro negocio. Por consultas para aclarar el enunciado que se detalla a continuación, deben contactarnos por slack o email.

Necesitamos construir una interfaz web para ABM (Alta/Baja/Modificación) de Solicitudes de Atención Ciudadana de la Ciudad de Buenos Aires.

Contamos con un set de datos con **solicitudes** de la población en formato CSV (Comma Separated Value).

La solución debe ser una aplicación Ruby - Sinatra que cumpla con los siguientes puntos:

- Se deben crear los modelos de clase necesarios para resolver el problema.
- Se deben persistir los objetos necesarios para resolver el problema utilizando YAML-STORE.
- Inicialmente los repositorios de datos en formato ".yaml" deben estar vacíos y se deben inicializar importando los datos provistos en formato csv.
- Se debe contar con una vista donde se muestre el enunciado del TP.
- Se debe contar con un menú de navegación para acceder a las distintas secciones (soluciones a los distintos puntos pedidos).
- Se debe contar con una vista de contacto que contenga información sobre ustedes (incluyendo foto de perfil).
- Se debe aprovechar las ventajas de utilizar un layout.

- Se debe dar feedback al usuario sobre las acciones realizadas en la aplicación. Por ejemplo, al crear una nueva solicitud, la aplicación me debería mostrar un mensaje de confirmación "Solicitud registrada correctamente".
- Se debe crear las rutas y vistas para obtener información y/o mostrar resultados que se piden en los diferentes enunciados.
- Se debe utilizar un framework CSS (Bootstrap por ejemplo).

### Dataset:

- atencion-ciudadana-2019.csv ([atencion-ciudadana-2019.csv](#))
- Todas las solicitudes corresponden al año 2019.
- Las solicitudes poseen distintas categorías.
- Las solicitudes pueden estar Abiertas, Cerradas o Canceladas.
- Validaciones:
  - Los tipos de prestaciones deben ser DENUNCIA, QUEJA, REPORTE, SERVICIO o SOLICITUD.
  - Las Fechas deben ser válidas (definir un formato aceptado y validarlo).
  - El Canal puede ser App, Web, 147 o Comunas.
  - La Calle del Domicilio no debe superar los 255 caracteres.
  - El Número de la Altura del Domicilio debe ser mayor a 0.
  - El Genero debe ser Masculino, Femenino u Otro.
  - Si el Estado de la solicitud es "Cerrado", debe tener Fecha de Cierre.

### Requerimientos funcionales:

1. Poder dar de alta una solicitud. Basarse aproximadamente en este ejemplo **\*no completo\***:



Formulario de Nueva Solicitud:

Numero: 00001

Categoría: TRANSITO - DENUNCIA VIAL

Tipo de Prestacion: Solicitud

Comuna: 12

Calle: LARSEN

Origen: Web

Estado: Abierto

Genero: ☒ Femenino ☐ Masculino ☐ Otro

Barrio: VILLA PUEYRREDON

Numero: 1234

Fecha Ingreso: 10/01/19 09:00

Botones: Guardar, Volver

2. Poder modificar una solicitud a través de su ID.
3. Poder cerrar una solicitud guardando la fecha de Cierre.
4. Poder Cancelar una solicitud Abierta, agregando la fecha de cancelación como la fecha de cierre.

5. Solo las Solicitudes Abiertas se pueden Modificar, Cerrar o Cancelar.
6. Obtener un listado de Solicitudes (ordenado por fecha de Ingreso en orden descendente, es decir, la solicitud más nueva, debe estar primera) basándose en el siguiente boceto:

Lista de Solicitudes				
Id	Categoría	Estado	Fecha Ingreso	Acciones
00001	TRANSITO - DENUNCIA VIAL	Abierta	10/01/2019	Mostrar - Editar - Cancelar
00002	ALUMBRADO - REPARACION DE LUMINARIA	Abierta	02/02/2019	Mostrar - Editar - Cancelar
00003	CALLES Y VEREDAS - REPARACION DE VEREDA	Abierta	03/03/2019	Mostrar - Editar - Cancelar
00004	CALLES Y VEREDAS - REPARACION DE BACHES	Cerrada	04/04/2019	Mostrar
00005	LIMPIEZA Y RECOLECCION - LIMPIEZA DE VIA PUBLICA	Cerrada	05/05/2019	Mostrar

7. Tener una vista de detalles de una Solicitud (basarse en el siguiente boceto):

### Detalles de la Solicitud

Numero

Categoría

Estado

Tipo de Prestacion

Genero ☒ Femenino ☐ Masculino ☐ Otro

Comuna

Barrio


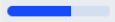


Calle

Numero

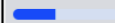



Origen

Fecha Ingreso

8. Poder dar de alta una Categoría.
9. Tener una página con la siguiente información:
  - a. ¿En qué comuna se abrieron más solicitudes?
  - b. ¿En qué fecha se abrieron más solicitudes?
  - c. ¿En qué mes se cerraron más solicitudes?
  - d. ¿Cuál es la comuna con la solicitud que más tiempo lleva abierta?
  - e. ¿Cuántas solicitudes llevan Abiertas más de 1 semana?
  - f. Para cada Género, ¿Cuál es la categoría que más solicitudes tiene?
  - g. Listar solicitudes que iniciaron y finalizaron dentro de un mes (el usuario ingresa en un formulario el mes de consulta).
10. Bajo una sección de estadísticas, mostrar:
  - a. Cantidad de Solicitudes por Comuna (tomar en cuenta el siguiente boceto):

Comuna	Cant. Solicitudes	Gráfico
1	3	
2	10	
3	2	
...		
15	2	

- Listar las 5 comunas que más solicitudes abrieron.
- Listar las 5 Solicitudes que más tardaron en cerrarse.
- Listar las 5 categorías con más solicitudes.
- Cantidad de solicitudes por día de la semana (como se indica en el siguiente boceto):

Comuna	Cant. Solicitudes	Gráfico
Lunes	3	
Martes	1	
Miercoles	10	
...		
Domingo	1	

- Calcular el porcentaje de solicitudes por categoría.
- Calcular el porcentaje de solicitudes por canal.
- Las 5 comunas que poseen el peor tiempo de resolución de solicitudes.

#### Documentación:

- <https://ruby-doc.org/core-2.2.0/Time.html>
- <https://github.com/lambdaweb/pillchart>
- <https://github.com/sinatra/sinatra>
- <https://getbootstrap.com/>
- <https://docs.ruby-lang.org/en/2.4.0/YAML/Store.html>