*RÚBRICA DE EVALUACIÓN DE ENTREGAS DEL PROYECTO FINAL*

**Python**

Una **rúbrica** es un conjunto de criterios y normas que sirven para evaluar el nivel de desempeño en una tarea. Es una herramienta de calificación que simplifica la tarea de corrección y permite lograr una evaluación más objetiva y transparente tanto para el evaluador como el evaluado.

Adicionalmente, la rúbrica ayuda a igualar y alinear los criterios de evaluación entre los diferentes docentes y tutores.

**En resumen, la rúbrica:**

1. Define los criterios de evaluación de cada entrega de proyecto final.
2. Describe y detalla los puntos clave que debe tener cada entrega.
3. Facilita y transparenta la corrección de cada trabajo.

*¿CÓMO SE UTILIZA LA RÚBRICA?* 🎦[Ver Tutorial](https://www.loom.com/share/ad487462a0bc424cb10355989ae7abe2)

*ENTREGA DEL PROYECTO FINAL*

**Se debe entregar**:

* Se deberá realizar en **duplas o tríos**, crearás una aplicación web estilo blog programada en **Python en Django**. Esta web tendrá administrador, perfiles, registros, páginas y formularios.
* La entrega se realizará enviando el **link a GitHub**, en el ***readme*** de Github deberá estar el ***nombre completo de los dos/tres participantes*** y una descripción de dos o tres renglones contando ***qué hizo cada uno.***
* En el github debe haber un ***video o link a vídeo*** donde nos muestran su web funcionando en ***no más de diez minutos.***
* ***Dentro del Github deberá existir una carpeta con por lo menos 3 casos de pruebas debidamente documentados.***

**Consigna:** Se tiene que crear una Web semejante a un Blog, dicha web deberá contar con usuarios y permisos para ellos. La web deberá contar con:

* Contar con algún acceso visible a la vista de "Acerca de mí" donde se contará acerca de los dueños de la página manejada en el route ***about/*.** (En castellano un acerca de mí que hable un poco de los creadores de la web y del proyecto).
* Contar con algún acceso visible a la vista de blogs que debe alojarse en el route pages/. (Es decir un html que permite listar todos los blogs de la BD, con una información mínima de dicho blog)
* Acceder a una pantalla que contendrá las páginas. Al clickear en “Leer más” debe navegar al detalle de la page mediante un route pages/<pageId>. (O sea al hacer click se ve más detalle de lo que se veía en el apartado anterior)
* Si no existe ninguna página mostrar un "No hay páginas aún". (Aclarando, si en la página hacemos clic en algún lugar que no existe que diga eso, o que lleve a un html con esos mensaje, no dejar botones que no responden)
* Para crear, editar o borrar las fotos debes estar registrado como Administrador.
* Cada blog, es decir cada model Blog debe tener como mínimo, un título, subtítulo, cuerpo, autor, fecha y una imagen (mínimo y obligatorio, puede tener más).
* Piezas sugeridas, no hace falta que estén todas, pero tiene que haber por lo menos un CRUD completo y el módulo de Login debe ser sólido:

| * NavBar * Home * About * Pages * Login * Signup | * Messages * Profile * Logout * Get pages * Get page | * Create page * Update Page * Delete page * Get profile * Update profile |
| --- | --- | --- |

**Inicio:** Al momento de ingresar a la app en la ruta base ‘/’

* Visualizar el home del blog.
* Poder listar todas las páginas del blog, poder ver en detalle cada una, poder crear, editar o borrar páginas del blog.
* Las páginas están formadas por un título, un contenido en editor de texto avanzado (ckeditor por ejemplo), una imagen, fecha de posteo de la imagen.
* Tener una app de registro donde se puedan registrar usuarios en el route accounts/signup, un usuario está compuesto por: email - contraseña - nombre de usuario
* Tener una app de login en el route accounts/login/ la cual permite loggearse con los datos de administrador o de usuario normal.
* Tener una app de perfiles en el route accounts/profile/ la cual muestra la info de nuestro usuario y permite poder modificar y/o borrar: imagen - nombre - descripción - un link a una página web - email y contraseña
* Contar con un admin en route admin/ donde se puedan manejar las apps y los datos en las apps.
* Tener una app de mensajería en el route messages/ para que los perfiles se puedan contactar entre sí.

| **Playground Django (10)** | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| **Criterios** | **Nulo**  AUSENTE  **0 PUNTOS** | **Bajo**  CONFUSO  **0,5 PUNTOS** | **En proceso**  FALTA DE PROFUNDIDAD  **1 PUNTOS** | **Correcto**  PROBLEMAS PUNTUALES  **2 PUNTO** | **Óptimo**  CLARO  **2,5 PUNTOS** | **NOTA** |
| Cumplimiento de consigna | No utiliza Django, solo Python o algún otro framework que no cumple con lo visto en el curso. | Intenta utilizar Django pero el proyecto no funciona o funciona con errores o solo algunos apartados puntuales. | El proyecto funciona. Crea una App que tiene algunas acciones operativas óptimas, pero no cumple con todos los requisitos funcionales pedidos. | En líneas generales el proyecto está bien estructurado y funciona correctamente. Crea varias App dentro del mismo proyecto que funcionan correctamente, aunque se encuentran algunos errores mínimos. Respeta la estandarización propuesta por Django. | Cumple con todos los Estándares de Django, el proyecto está subdividido en varias App y ambas están internamente bien diseñadas y funcionales. No se encuentran errores ni incongruencias. |  |
| Aplicación de lo aprendido | La entrega no muestra similitud alguna con todo lo visto en el curso. | Utiliza Python pero no se entiende la estructura que se intentó usar, la programación no respeta los estándares vistos en el curso, parece que el proyecto fue bajado de internet o realizado sin seguir el curso. | Respeta gran parte de lo aprendido, aunque no cumple con alguno de los estándares planteados. Por ejemplo, privacidad de los elementos, distribución en carpetas, nombre de las variables y métodos, etc. | El proyecto final se encuentra bien generado. Olvido algunos apartados puntuales como por ejemplo camelCase por momentos (o snakeCase), mezcla mucho el inglés y el español, utiliza muchas cosas sacadas de distintas fuentes o no pudo documentar bien los casos de prueba, el resto muy bien generado. | Cumple con todos los requisitos que hacen al curso. No se encontraron déficit importantes en cuanto al diagramado, estructura y desarrollo del proyecto general y su evolución. |  |
| Aspectos estéticos | Los html básicos no funcionan o no son operativos, o son estáticos y no están asociados al Backend Django. | Si bien el curso no se basa en el desarrollo web esta Web es prácticamente imposible de seguir y entender, la disposición de los elementos es confusa y las opciones no responden correctamente. | La web funciona en líneas generales, salvo algunas salvedades. El contenido está distribuido de una forma no muy amigable a la vista y no se entiende perfectamente el flujo que relaciona urls, html y view. | Buena distribución de los elementos web, todos o casi todos operativos y gran uso de los lineamientos MVT. | Excelente los aspectos estéticos, todo funciona correctamente, respeta MVT y además usó las funcionalidades mínimas de Bootstrap o algún otro framework similar. |  |

* 0,25 por originalidad a criterio de quien corrige.