Proyecto de Software 2023
Trabajo Integrador (TI)
Etapa 2

Aplicación Privada

Registro con Google

Modifique el módulo de autenticación para que permita registrar al usuario utilizando la cuenta de google. En este caso los datos necesarios (nombre, apellido y correo electrónico) serán solicitados durante el proceso de registración.

Aplicación Pública

Se deberá desarrollar **otra aplicación** a la cual accederán los clientes para obtener información de interés sobre el Laboratorio, información Institucional y servicios habilitados.
El código de esta nueva aplicación deberá compartir espacio dentro del mismo repositorio de código de la aplicación Flask. La aplicación pública la deberán realizar utilizando el framework VueJs 3. **Cómo requisito para el correcto funcionamiento en el servidor se deberá ubicar el código correspondiente a esta aplicación dentro del directorio web del proyecto**.

2.1 Home + Información de contacto

La vista principal de la aplicación deberá contar con una explicación resumida del servicio que brinda el portal, y una breve explicación de las acciones que puede realizar un visitante en el mismo. A saber:

- Realizar búsqueda de servicios.
- Una sección donde se listen los servicios prestados por cada institución.
- Iniciar una solicitud de servicio.

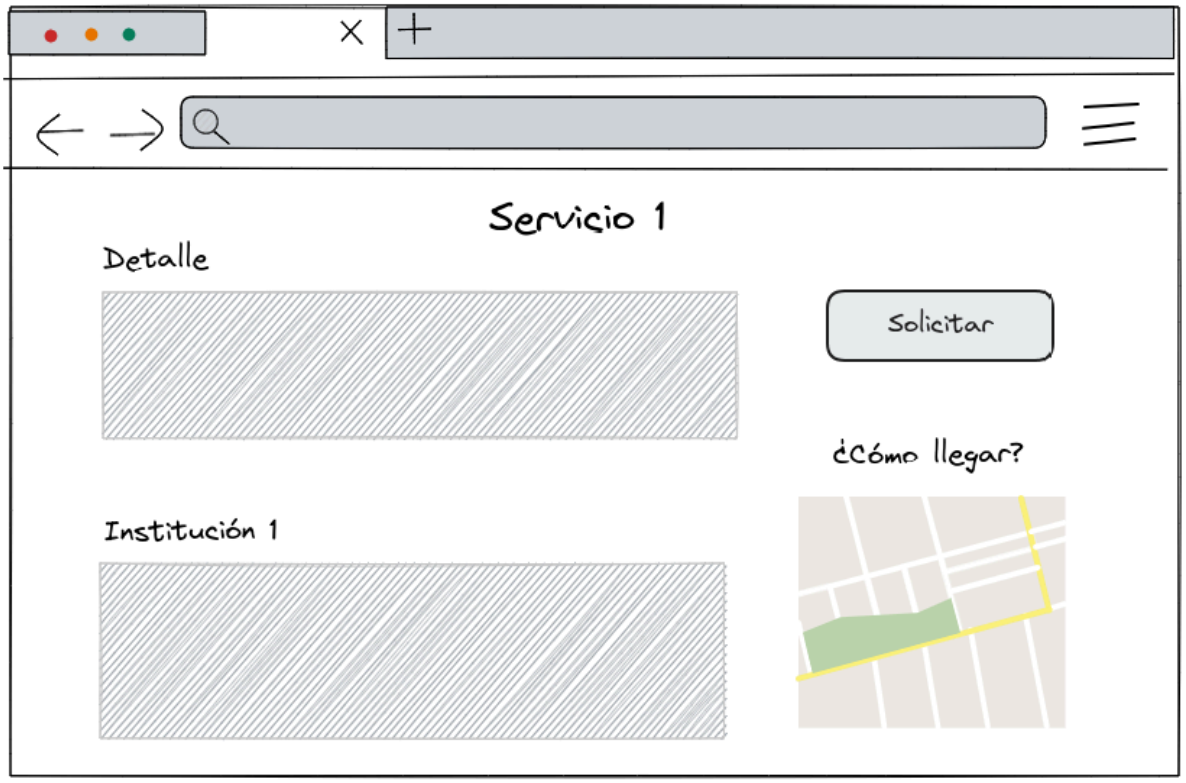
2.2 Buscador de Servicios

En esta sección se visualizará un listado de los servicios y detalle de los mismos. Para obtener la información necesaria del listado de servicios, deberá consultar la **API de Búsqueda.**
Input texto

- **Título de servicio, descripción del servicio, nombre de institución y tags.**
- **Input tipo de servicio.**
- **Desplegable con la lista de servicios.**

2.3 Visualizar servicio

Esta sección permite visualizar el detalle del servicio seleccionado por el usuario que se encuentra realizando la búsqueda en la aplicación pública. La vista también deberá visualizar información relacionada a la Institución que brinda el servicio y un botón para solicitar el servicio, que en caso de hacer clic, si el usuario no se encuentra autenticado deberá redireccionarlo al login.
Se deberá incorporar un mapa en el que se visualice la ubicación de la institución, que permita visualizar información básica de contacto de la institución luego de hacer clic en el punto.

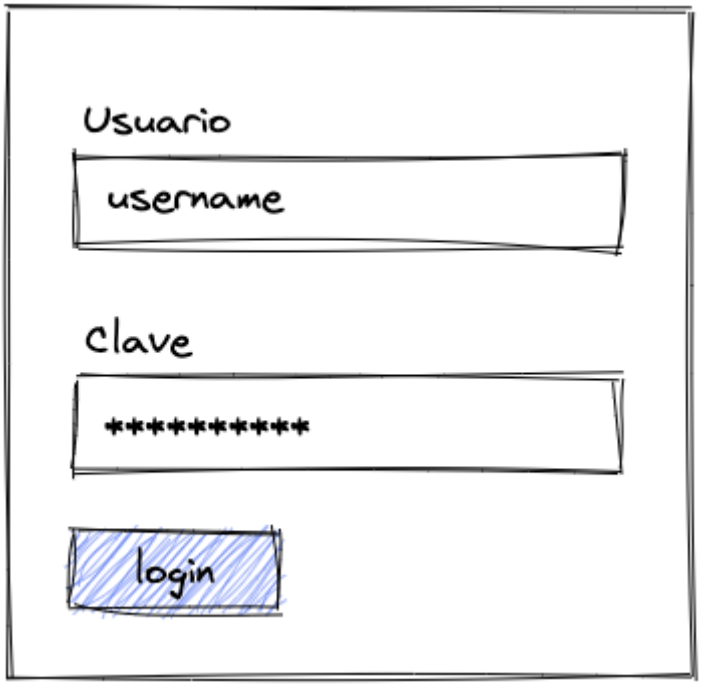


2.4 Solicitar servicio

En esta sección un cliente (autenticado) deberá completar un formulario para solicitar un servicio específico en un centro habilitado incluyendo una resumen del trabajo a solicitar y teniendo la posibilidad de adjuntar archivos complementarios. Para realizar esta operación, deberá utilizar la **API de Solicitudes.**

2.5 Autenticación

Implemente un login basado en JSON Web Tokens (JWT), que permita acceder a la información en la aplicación privada. Esta funcionalidad deberá utilizar la **API de Login** definida en 2.3.1.
El cliente puede loguearse en la aplicación, para esto es necesario visualizar un formulario de login. El formulario utiliza la API de login JWT mencionada anteriormente.



Se deben realizar las modificaciones de los endpoints de la etapa 1 que requieren autenticación para que hagan uso del JWT generado en el login.

2.6 Listado de pedidos del usuario

Esta sección permite visualizar el listado de pedidos solicitados por el cliente que se encuentra logueado en la aplicación pública, pudiendo filtrar y ordenar por fecha o estado . Para obtener la información necesaria del listado deberá consultar la **API de Pedidos.**

Allí el cliente podrá ver el estado del pedido realizado, como así también acceder a las notas que les dejó el administrador del Laboratorio, teniendo posibilidad de responder las mismas.

2.7 Análisis de datos

Esta sección va a permitir visualizar información estadística que pueda ser de utilidad tanto para los SuperAdmin como para los dueños de cada institución.

En esta vista se deberán diseñar y desarrollar 3 gráficos distintos. Queda a criterio de las y los desarrolladores/as con una validación previa del ayudante asignado que tipo de gráficos generar teniendo en cuenta los datos utilizados.
Algunos ejemplos de reportes pueden ser:

- Gráfico de tortas/barras con la cantidad de las solicitudes realizadas agrupadas por estado.
- Ranking de los servicios más solicitados
- Top 10 de las Instituciones con mejor tiempo de resolución[1]
    - *[1] Es el tiempo que se tarda desde que se crea la solicitud hasta que finaliza la misma.*

Tener en cuenta que esta funcionalidad se debe realizar en la aplicación *pública*. Se deberá utilizar un componente que sirva para realizar esta implementación.
Deben implementar los endpoints en el backend que les provean la información necesaria para mostrar esta info.

Consideraciones generales:

- El prototipo debe ser desarrollado utilizando Python, JavaScript, HTML5, CSS3 y PosgreSQL, y **respetando el modelo en capas MVC**.
- El código deberá escribirse siguiendo las guías de estilo de Python.
- El código Python deberá ser documentado utilizando docstrings.
- **El uso de jinja como motor de plantillas es obligatorio para la aplicación privada**.
- Se debe utilizar Flask como framework de desarrollo web para la aplicación privada.
- Se deberán realizar **validaciones de los datos de entrada** tanto del lado del cliente como del lado del servidor. Para *las validaciones del lado del servidor se deben realizar en un módulo aparte* que reciba los datos de entrada y devuelva el resultado de las validaciones. En caso de fallar el controlador debe retornar la respuesta indicando el error de validación.
- Podrán utilizar librerías que facilitan algunas de las tareas que deben realizar en el trabajo como pueden ser: conexión a servicios externos, librerías de parseo, librerías con patrones para buenas prácticas, validaciones de datos, etc. **Pero todos los miembros del equipo deben demostrar en la defensa**

pleno conocimiento del funcionamiento de estas librerías y una idea de cómo solucionan el problema.

- Para la implementación del Login **no se podrá utilizar librerías como Flask-Login** dado que consideramos que ocultan implementación que queremos asegurarnos que entiendan en el transcurso de esta materia.
- Para la interacción con la base de datos se deberá utilizar el ORM SQLAlchemy que nos permita además tener una capa de abstracción con la BD.
- Para la aplicación pública se debe utilizar el Framework web VueJS versión 3. Se deberá utilizar la librería vue-router para implementar el ruteo de la aplicación.
- No pueden utilizar un framework/generador de código para el desarrollo de cada una de las API requeridas en el enunciado.
  Debe tener en cuenta los conceptos de Semántica Web proporcionada por HTML5 siempre y cuando sea posible con una correcta utilización de las etiquetas del lenguaje.
  El trabajo será evaluado desde el servidor de la cátedra que cada grupo deberá gestionar mediante Git. **NO se aceptarán entregas que no estén realizadas en tiempo y forma en el servidor provisto por la cátedra**.
  Cada entrega debe ser versionada por medio de git utilizando el sistema de Versionado Semántico para nombrar las distintas etapas de las entregas. Ejemplo: para la etapa 1 utilizar la versión v1.x.x.
  Deberán visualizarse los aportes de cada uno/a de los/as integrantes del grupo de trabajo tanto en Git como en la participación de la defensa.
  El/la ayudante a cargo **evaluará el progreso y la participación** de cada integrante mediante las consultas online y el seguimiento mediante GitLab.
  Seleccione una vista (**HTML5** y **CSS3**) para realizar validaciones de las especificaciones de la W3C (http://validator.w3.org/ y https://jigsaw.w3.org/css-validator/ respectivamente). En esta oportunidad puede utilizar **Bootstrap** u otro framework similar. En caso de que alguna de las vistas no valide, deberá realizar un breve informe indicando cuales son los errores encontrados (este informe deberá realizarse para la etapa 1).
  Todas las vistas deben cumplir ser web responsive y visualizarse de forma correcta en distintos dispositivos. Al menos deben contemplar 3 resoluciones distintas:

  - res < 360
  - 360 < res < 768
  - res > 768

  La entrega es obligatoria. Todos y todas los/as integrantes deben presentarse a la defensa.
  El sistema no debe ser susceptible a SQL Injection, XSS ni CSRF.
  Importante: el proyecto podrá ser realizado en grupos de tres o cuatro integrantes (será responsabilidad de los y las estudiantes la conformación de los equipos de trabajo). Todos y todas los/as estudiantes cumplirán con la totalidad de la consigna, sin excepciones.

Información del servidor

Tener en cuenta que el servidor de la cátedra utiliza los siguientes servicios y versiones:

- Servidor de Base de Datos: Postgres 13
- Intérprete Python: Python 3.8.10
- Servidor web: Nginx 1.18.0-0ubuntu1.3

---

[1] Es el tiempo que se tarda desde que se crea la solicitud hasta que finaliza la misma.