INFORME DE ACTIVIDADES ANGULAR 1 Y 2

Tema: Angular

Nota

Estudiantes	Escuela	Asignatura
Fabiana Paola Rojas Condori	Escuela Profesional de Ingeniería de Sistemas	Programación Web II Semestre: III Código:

	Tema	
	Angular	-

Semestre académico	Fecha de inicio	Fecha de entrega
2025 - A		05 / 07 / 2025

1. Presentación de la Tarea

- Volver a implementar las clases teóricas en un proyecto en github realizando commits de cada avance. Compartirlo con el profesor sacando un pantallazo de los commits y agregandolo al informe.

2. Equipos, materiales y temas utilizados

- **Visual Studio Code**: Editor de código fuente utilizado como entorno principal de desarrollo.
- **Angular CLI (versión 20)**: Herramienta de línea de comandos para crear componentes, servicios y gestionar el proyecto.
- **Node.js y npm**: Para ejecutar y administrar dependencias del proyecto.
- **Git**: Sistema de control de versiones para registrar los cambios realizados en cada etapa del proyecto.
- **Cuenta en GitHub**: Utilizada para almacenar el repositorio del proyecto con una cuenta institucional.
- **Framework Angular**: Utilizado para desarrollar la aplicación con arquitectura basada en componentes. Se aplicaron conceptos como data binding, servicios, formularios, rutas y uso de componentes standalone.
- **Sistema operativo**: Ubuntu (vía WSL) para ejecutar los comandos en la terminal de forma fluida.
- **LaTeX**: Para la redacción y estructura del informe final técnico.

3. URL de Repositorio Github

- URL del Repositorio GitHub general.
- `https://github.com/FabianaRojasCondori/PW2-TEORIA.git`
- URL para Angular 1 y2 en el Repositorio GitHub.
- `https://github.com/FabianaRojasCondori/PW2-TEORIA/tree/main/proyectoAngular/my-dream-app`
- URL para el proyecto visto en la nube desde vercel
- `https://angular-ekpk.vercel.app/`

4. Estructura del proyecto

- Contenido presente en my-dream-app

```
my-dream-app/
     |----angular.json
     |----dist
     |----node_modules
     |----package-lock.json
     |----package.json
     |----public
     |----src
     |----tsconfig.app.json
     |----tsconfig.json
     +----tsconfig.spec.json
```

- Carpeta src

```
.
|----app
       |----Post.ts
       |----about
              |----about.css
              |----about.html
              |----about.spec.ts
              +----about.ts
       |----app.component.css
       |----app.component.html
       |----app.component.ts
       |----app.config.ts
       |----app.css
       |----app.html
       |----app.routes.ts
       |----app.spec.ts
       |----app.ts
       |----data.service.ts
       |----data.spec.ts
       |----data.ts
       |----hello-world
              |----hello-world.css
```

```
              |----hello-world.html
              |----hello-world.spec.ts
              +----hello-world.ts
       +----user
          |----user.css
          |----user.html
          |----user.spec.ts
          +----user.ts
|----index.html
|----main.ts
+----styles.css
```

5. Commits realizados

A continuación se muestran los commit más relevantes que fueron realizados en cada rama de los integrantes.

```
Commit1: angular1 - Diapositiva 11: Se realizaron las instalaciones y el primer
    desarrollo en angular}
```

En esta primera parte del desarrollo se crearon los componentes requeridos y se empezó con el desarrollo del código indicado en las diapositivas. La visualización de dichos cambios se muestra a continuación.

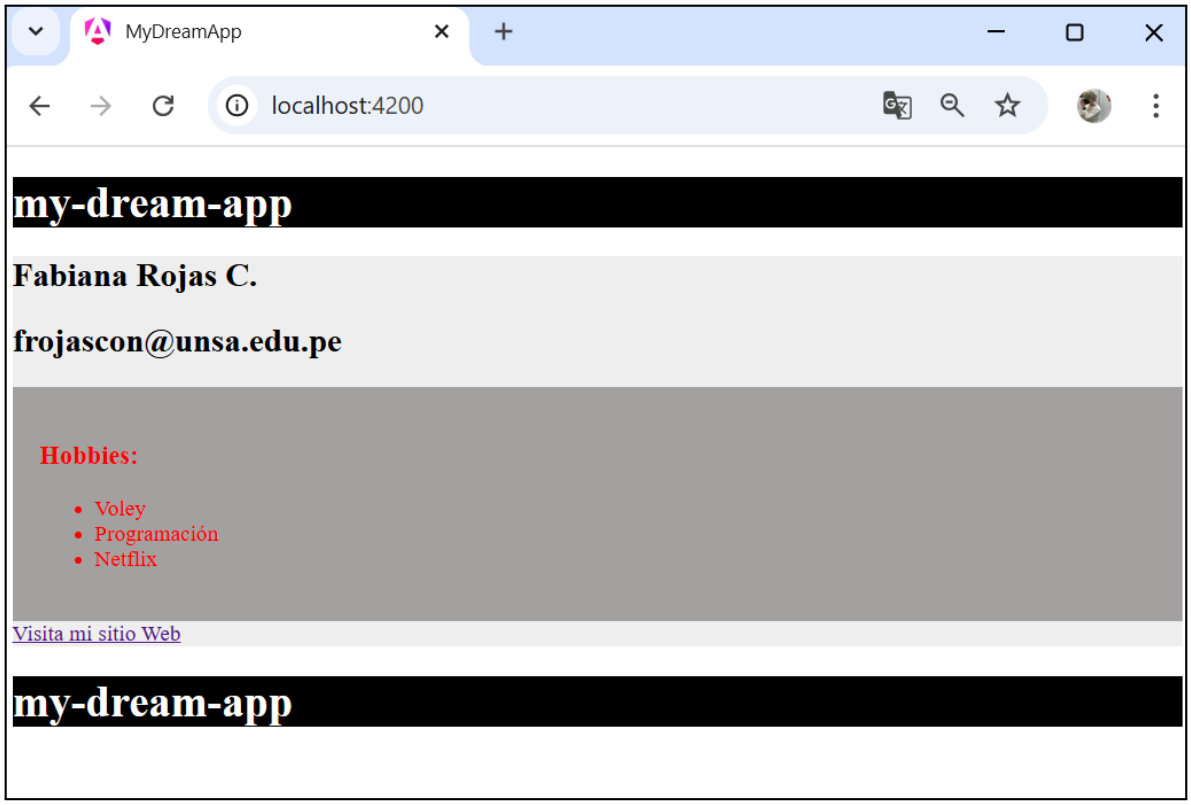
Ejecución

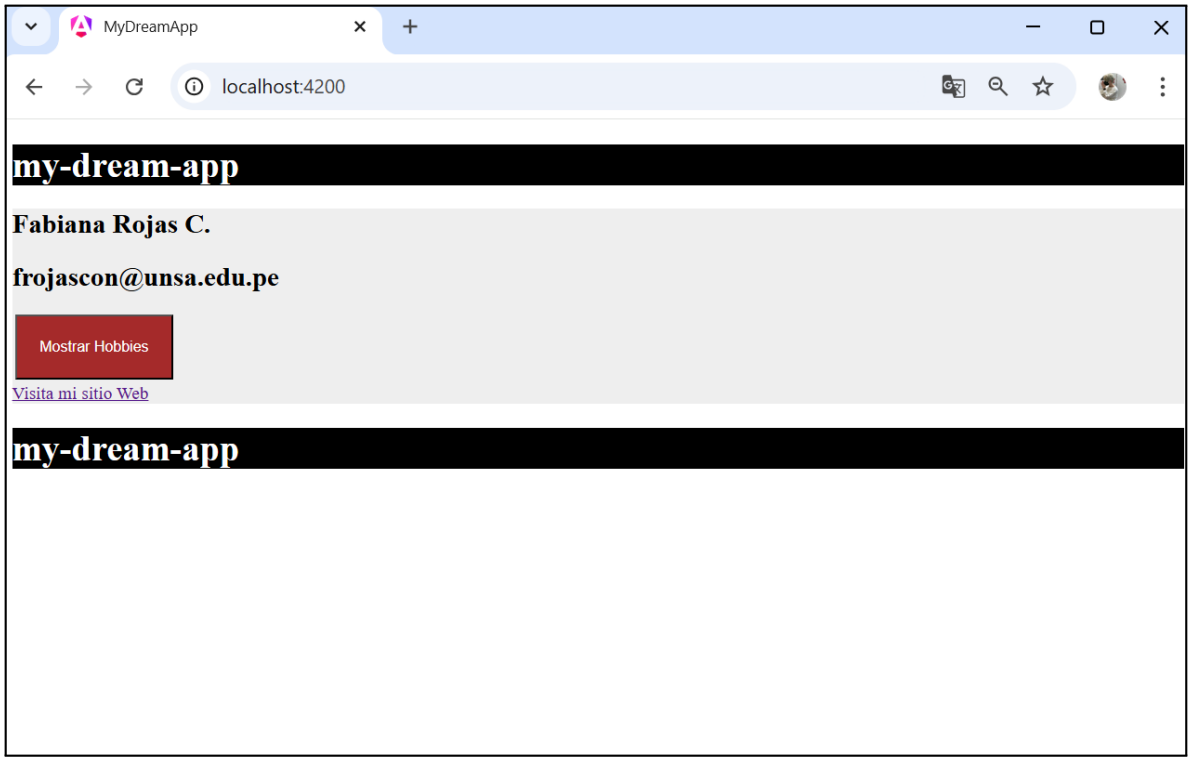

```
Commit 2: Angular1 - Diapositivas 13,14: Se agrego un constructor en app.component.ts y
    se usaron For e Ifs para mostrar los hobbies
```

Durante esta etapa, se implementó una sección dinámica para mostrar los hobbies del usuario utilizando Angular. Se incorporaron directivas estructurales como *ngIf y *ngFor, controladas por un

método showhobbies() y una propiedad hobbies definida en el componente. Además, se aplicaron estilos personalizados mediante clases CSS (.black, .ash, .component2) para mejorar la presentación visual del contenido, logrando una interfaz clara y segmentada. Estos cambios permitieron integrar datos personales, navegación fluida y componentes visuales.

Ejecución

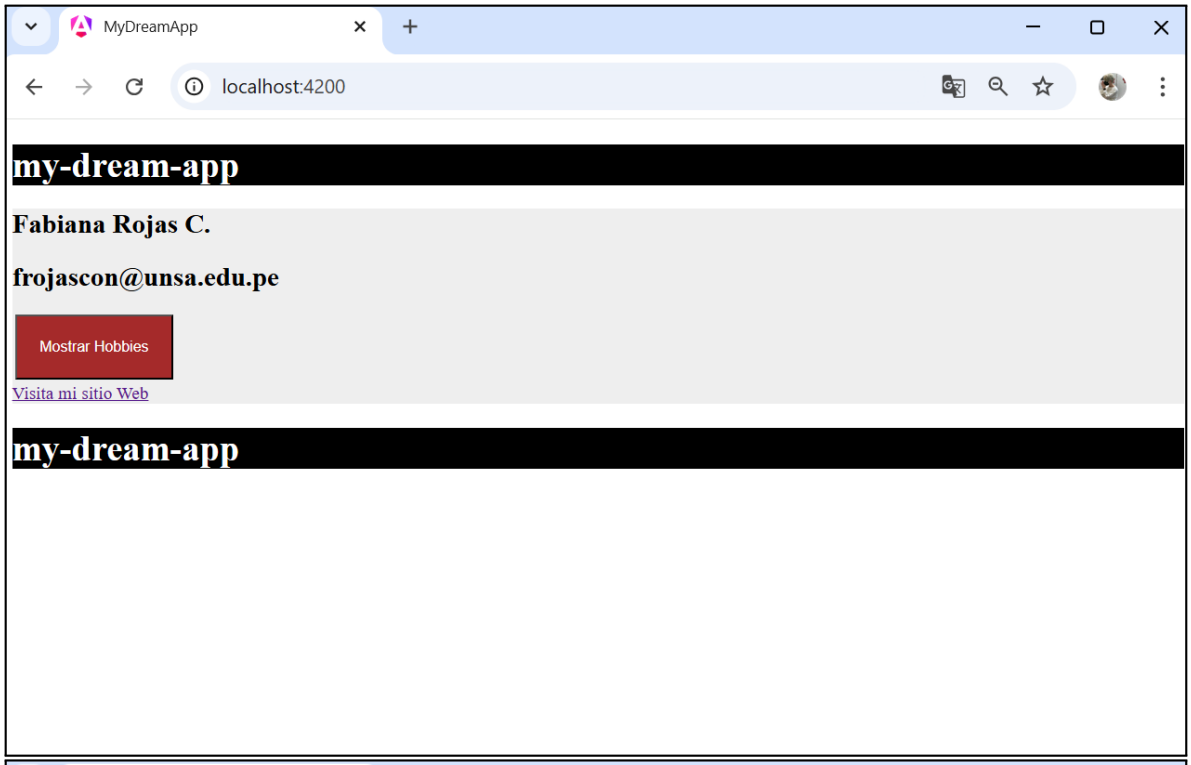


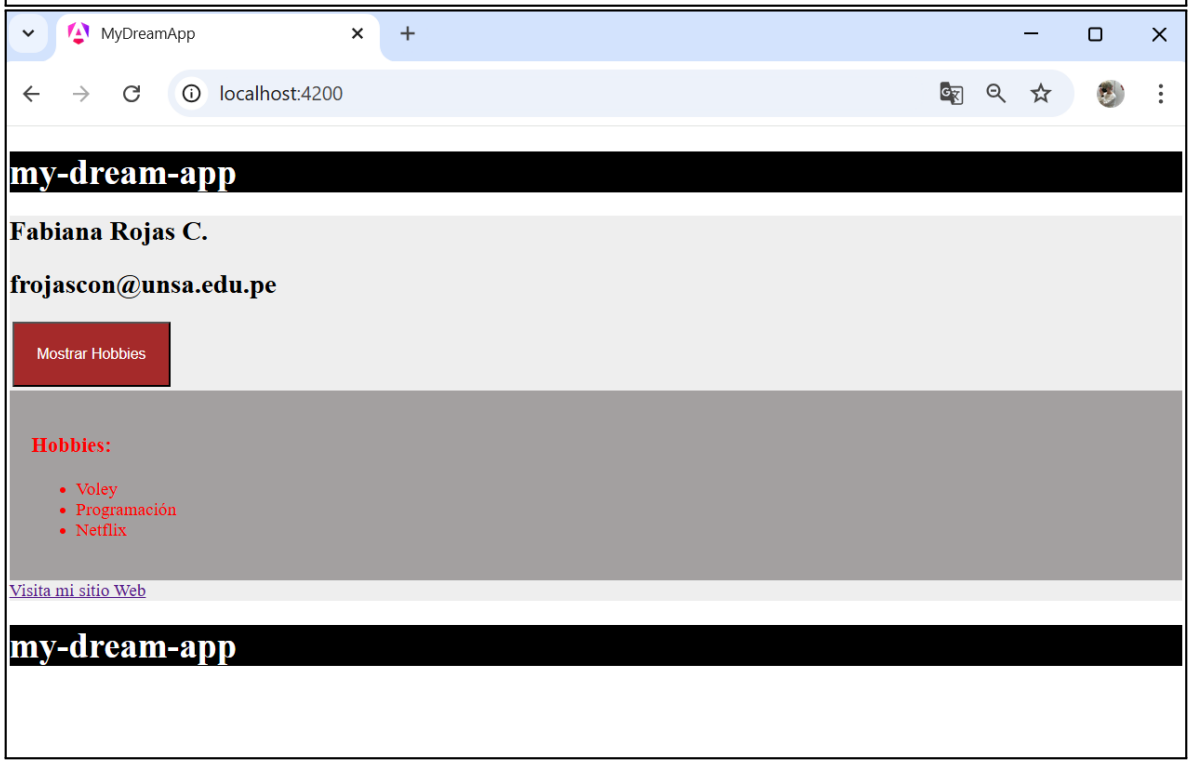



```
Commit 3: "Angular1 - Diapositiva 16: Se agrega un botn para mostrar Hobbies."
```

se añadió interactividad a la sección de hobbies mediante un botón con clase personalizada .btn, el cual permite mostrar u ocultar dinámicamente la lista de actividades favoritas del usuario. Para lograrlo, se implementó en app.component.ts la propiedad booleana showHobbies y el método toggleHobbies(), vinculados al botón mediante un evento (click). Además, se aplicaron estilos visuales distintivos a través de clases CSS (.component2, .btn) que mejoran la presentación visual y refuerzan la experiencia de usuario en la interfaz Angular.

Ejecución



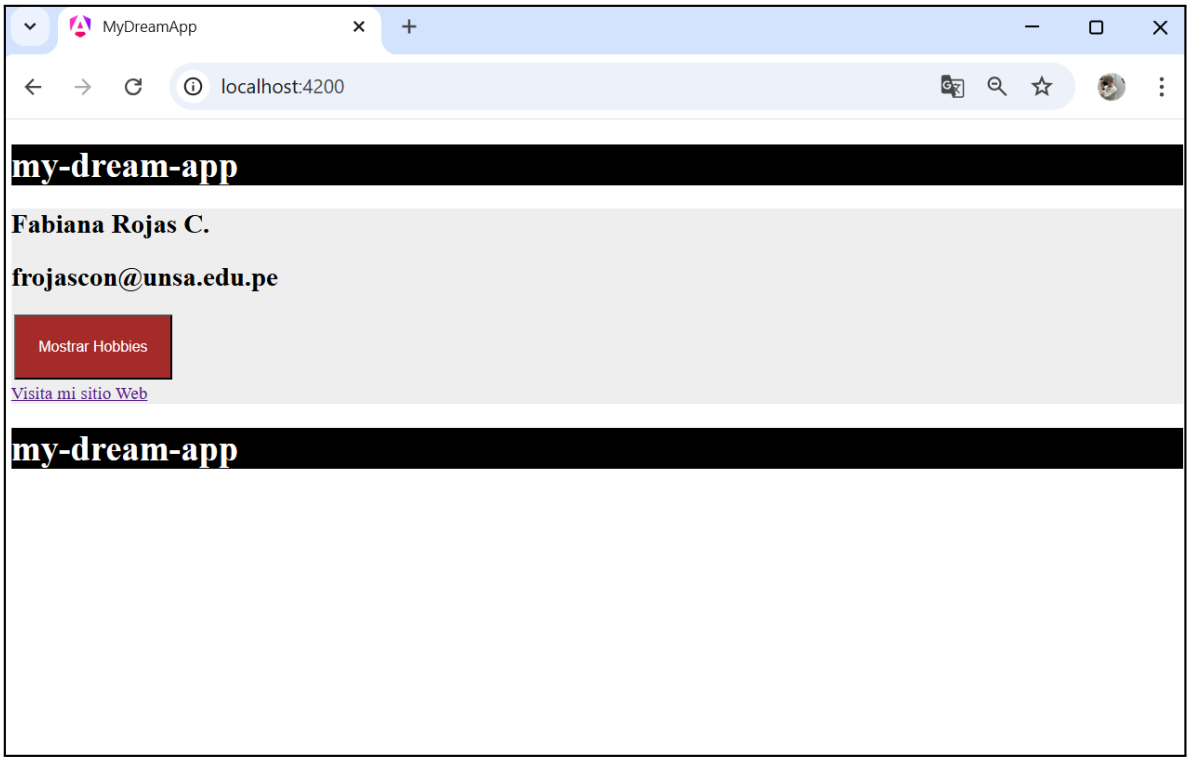


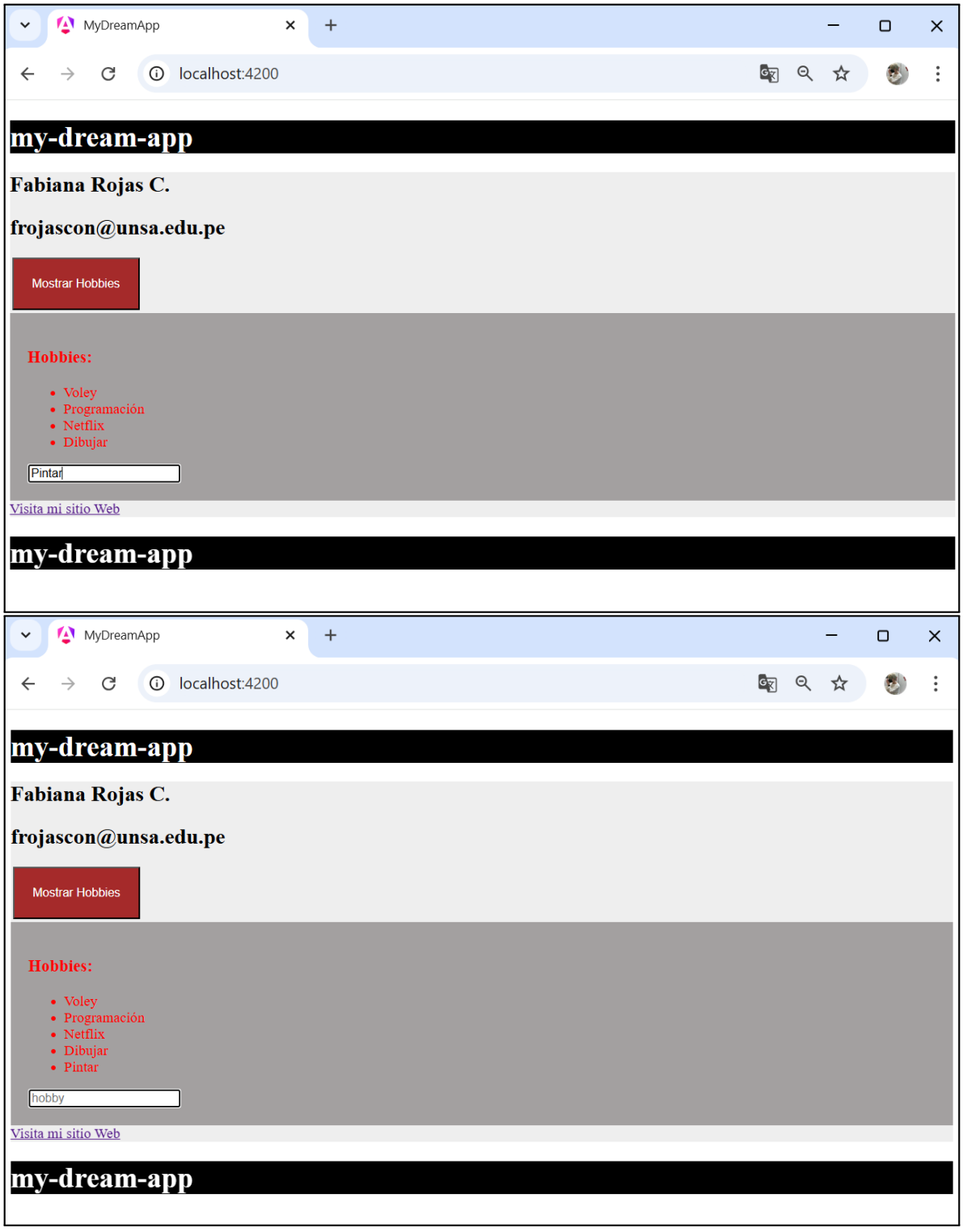

```
Commit 4: "Angular1 - Diapositiva 17: Se agrega un form para poder aadir ms Hobbies."
```

se incorporó un formulario dentro de la sección de hobbies que permite al usuario añadir nuevas actividades de forma dinámica. Para lograr esta funcionalidad, se implementó el método newHobby()

en el componente principal, el cual se activa al enviar el formulario y agrega el valor ingresado al arreglo hobbies. Además, se actualizó el HTML utilizando referencias locales y la directiva (submit) para captar los datos introducidos.

Ejecución



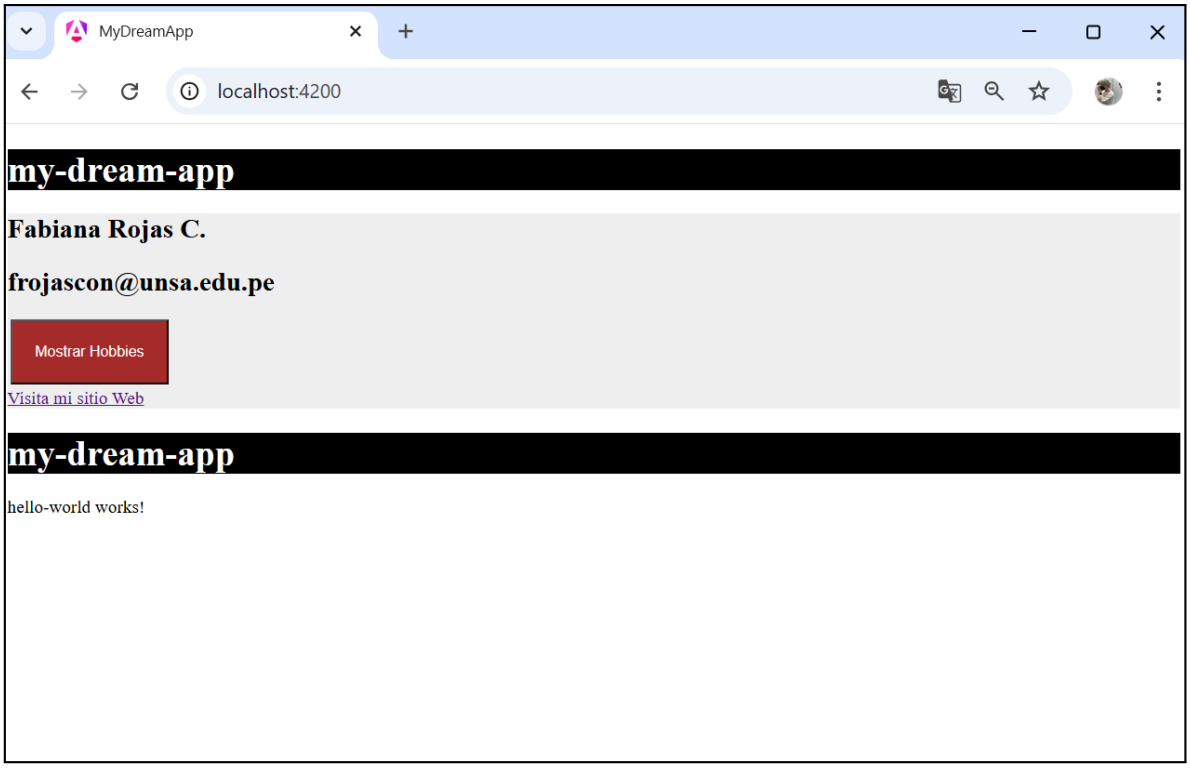



```
Commit 5: "Angular1 - Diapositiva 20: Se genero el componente hello-world."
```

se generó el componente hello-world como parte del proceso de modularización de la aplicación. Este

nuevo componente fue creado con una estructura standalone y se integró al AppComponent tanto en el arreglo de imports como en la plantilla HTML mediante el selector ¡app-hello-world¿. Además, se añadieron sus archivos asociados (.ts, .html, .css, y .spec.ts) para asegurar su correcto funcionamiento, visualización y testeo.

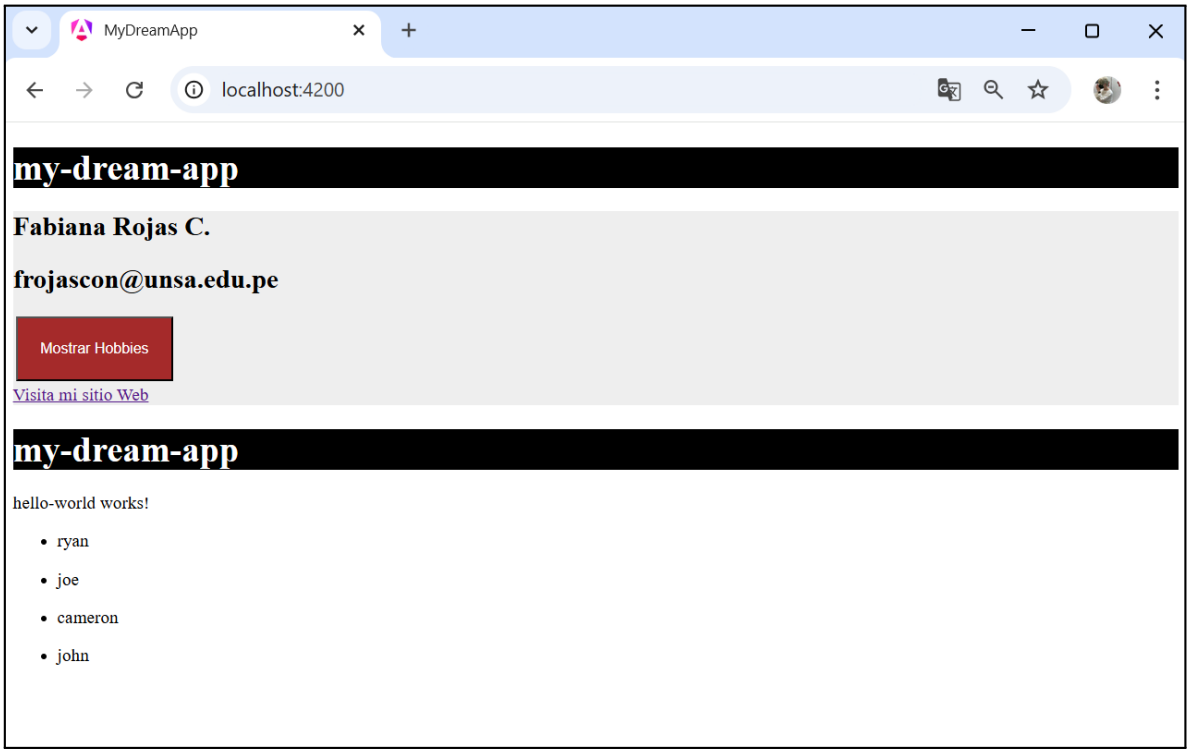
Ejecución




```
Commit 6: "Angular1 - Diapositiva 22: Se pasaron datos al componente user y se crearon
    'ryan, joe, cameron, john' segun indicaciones."
```

se incorporó el componente user al proyecto con el objetivo de representar a cada usuario de manera individual y reutilizable. Se definió una propiedad users en el componente principal (AppComponent) con los nombres 'ryan', 'joe', 'cameron' y 'john', y se utilizó *ngFor para recorrer ese arreglo en la plantilla. A través de @Input(), se pasó dinámicamente cada nombre al componente user, el cual se encargó de renderizarlos dentro de su propia vista (user.html).

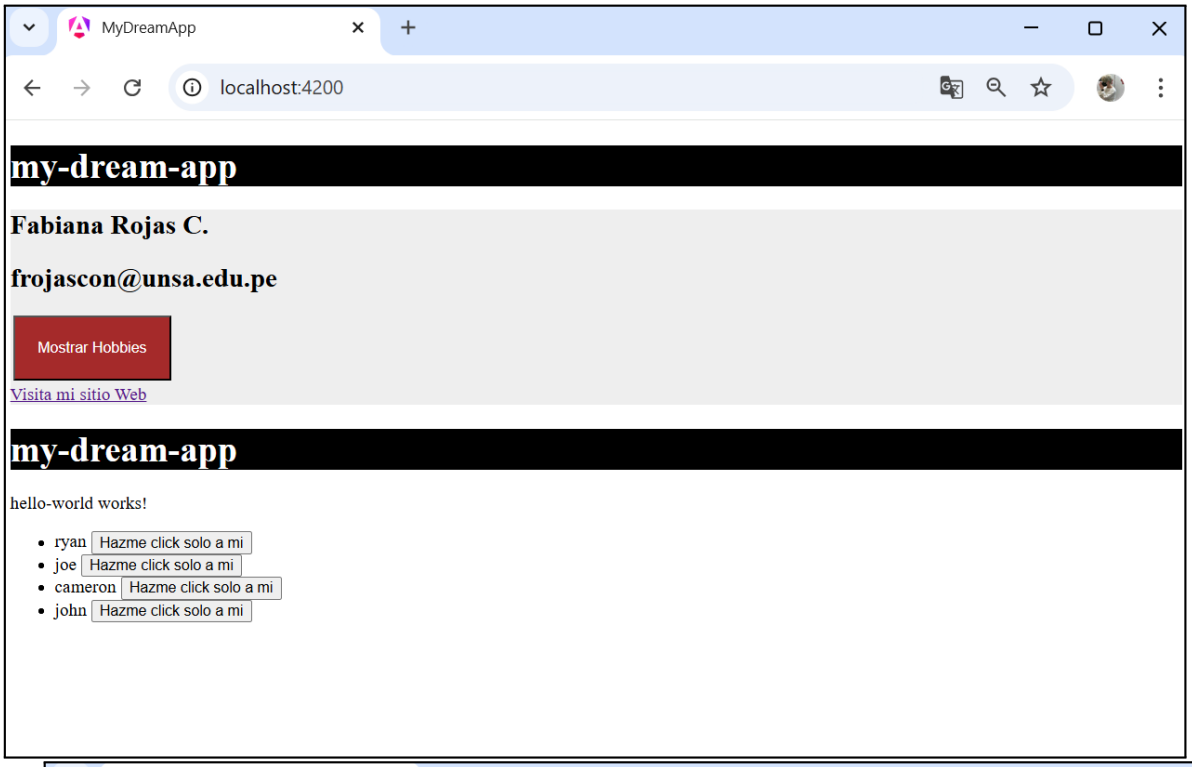
Ejecución

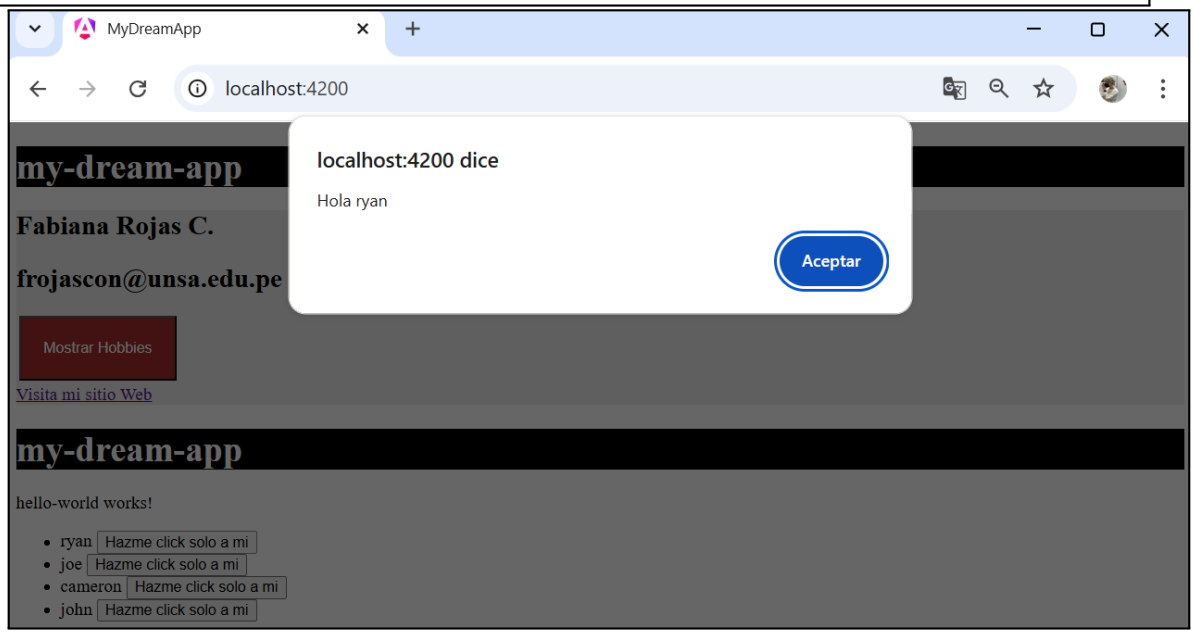


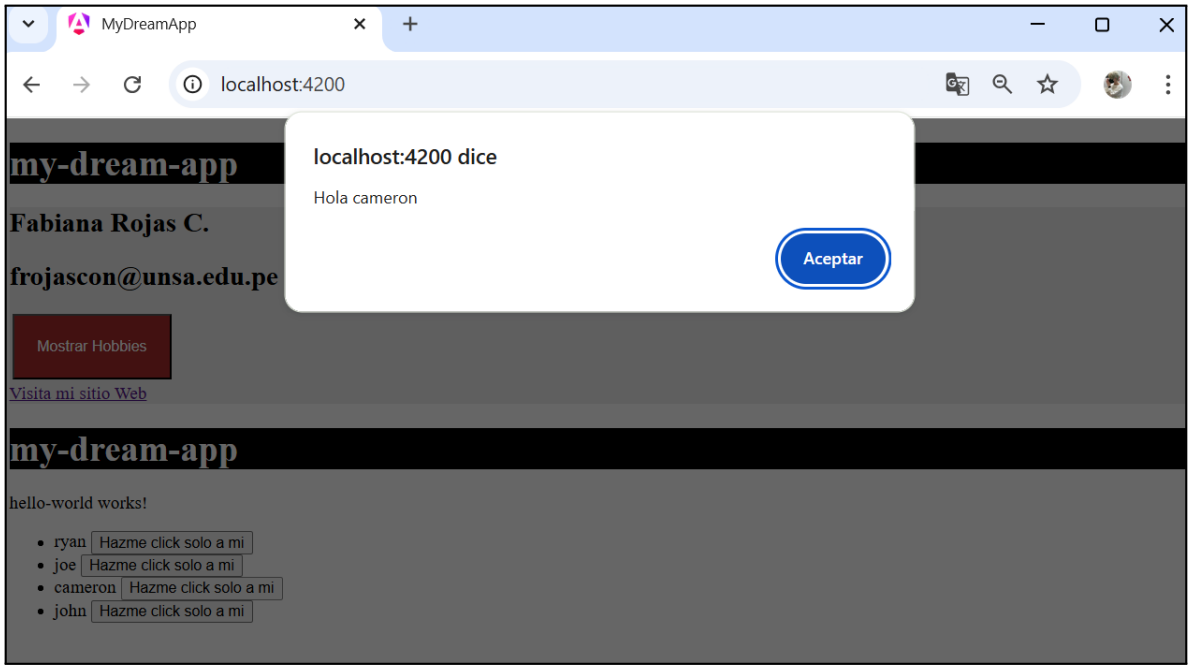

```
Commit 7: "Angular1 - Diapositiva 23: Se agregan botones de alerta a la aplicacion para que
    cada usuario tenga un boton que diga un mensaje('Hazme click solo a mi)."
```

se mejoró la interacción en el componente user incorporando un botón personalizado para cada elemento de la lista de usuarios. Mediante la función sayHello(nameUser), se activó una alerta que muestra un mensaje específico con el nombre del usuario al hacer clic en su botón. Esta funcionalidad fue implementada utilizando @Input() para recibir el dato del nombre y una función de alerta vinculada al evento (click).

Ejecución


MyDreamApp
localhost:4200
my-dream-app
Fabiana Rojas C.
frojascon@unsa.edu.pe
Mostrar Hobbies
Visita mi sitio Web
my-dream-app
hello-world works!
ryan Hazme click solo a mi
joe Hazme click solo a mi
cameron Hazme click solo a mi
john Hazme click solo a mi


MyDreamApp
localhost:4200
localhost:4200 dice
Hola ryan
Aceptar
my-dream-app
Fabiana Rojas C.
frojascon@unsa.edu.pe
Mostrar Hobbies
Visita mi sitio Web
my-dream-app
hello-world works!
ryan Hazme click solo a mi
joe Hazme click solo a mi
cameron Hazme click solo a mi
john Hazme click solo a mi



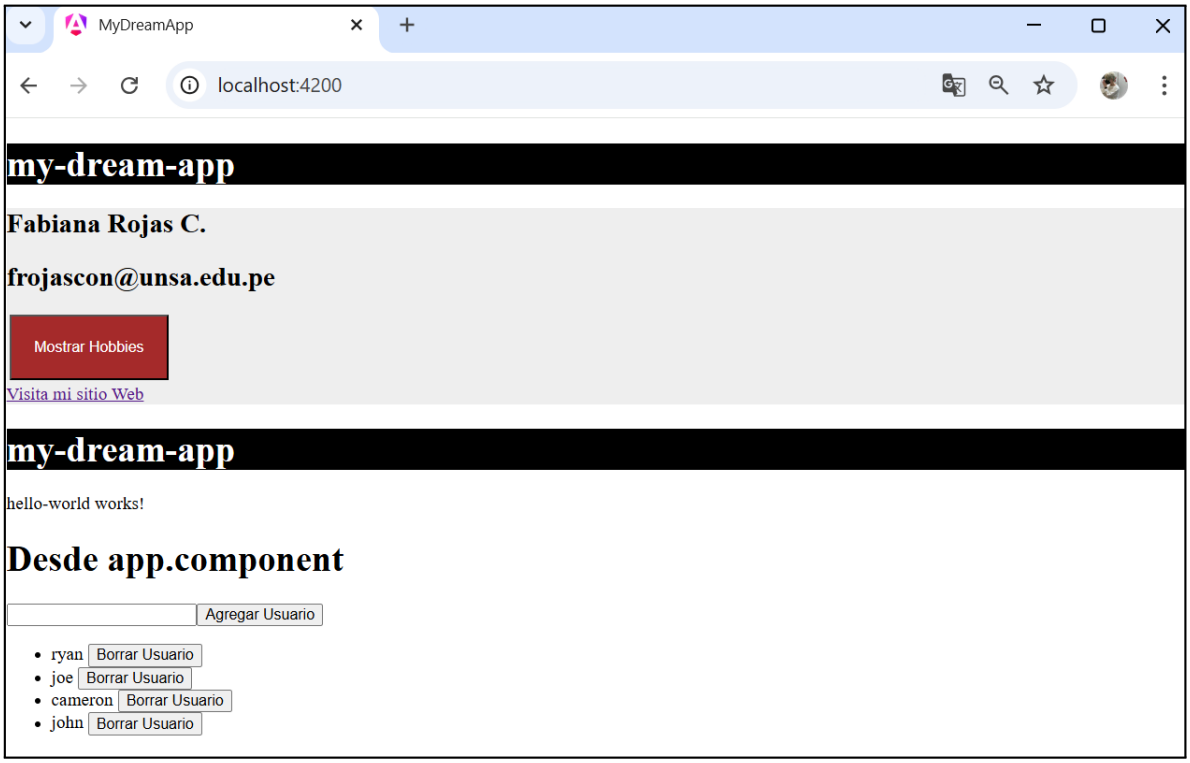

```
Commit 8: "Angular1 - Diapositiva 24: Se agregan botones que borran elementos de una lista
    (usuarios)."
```

En el Commit 8, correspondiente a la Diapositiva 24, se añadió la funcionalidad para eliminar usuarios individualmente desde la vista principal del componente. Para lograrlo, se implementó el método deleteUser(user) dentro de AppComponent, el cual recorre el arreglo users y remueve el elemento seleccionado. Esta lógica fue integrada al HTML mediante un botón por cada usuario, enlazado con (click)=”deleteUser(user)”, lo que permitió una experiencia más interactiva y dinámica, fortaleciendo el manejo de eventos y manipulación de listas en Angular.

Ejecución


```
Commit 9: "Angular1 - Diapositiva 26: Pasando referencias del Formulario - Ahora se puede
    agregar y eliminar usuarios."
```

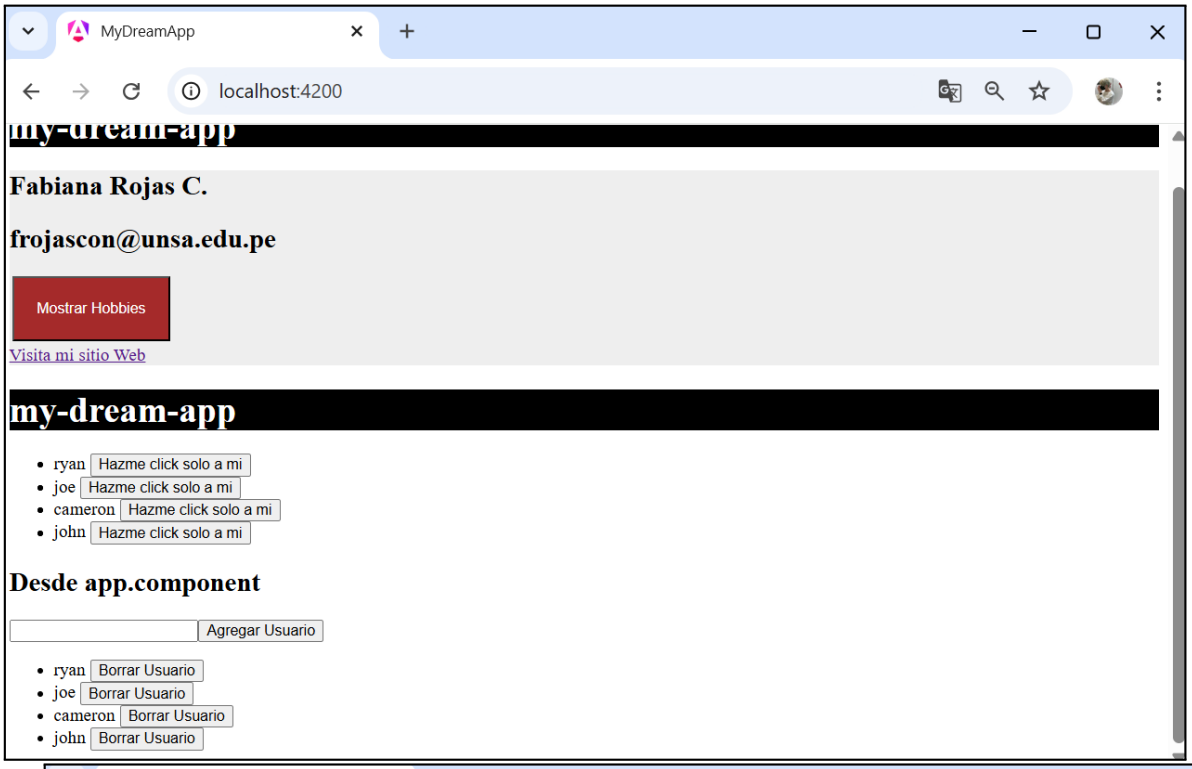
Ejecución

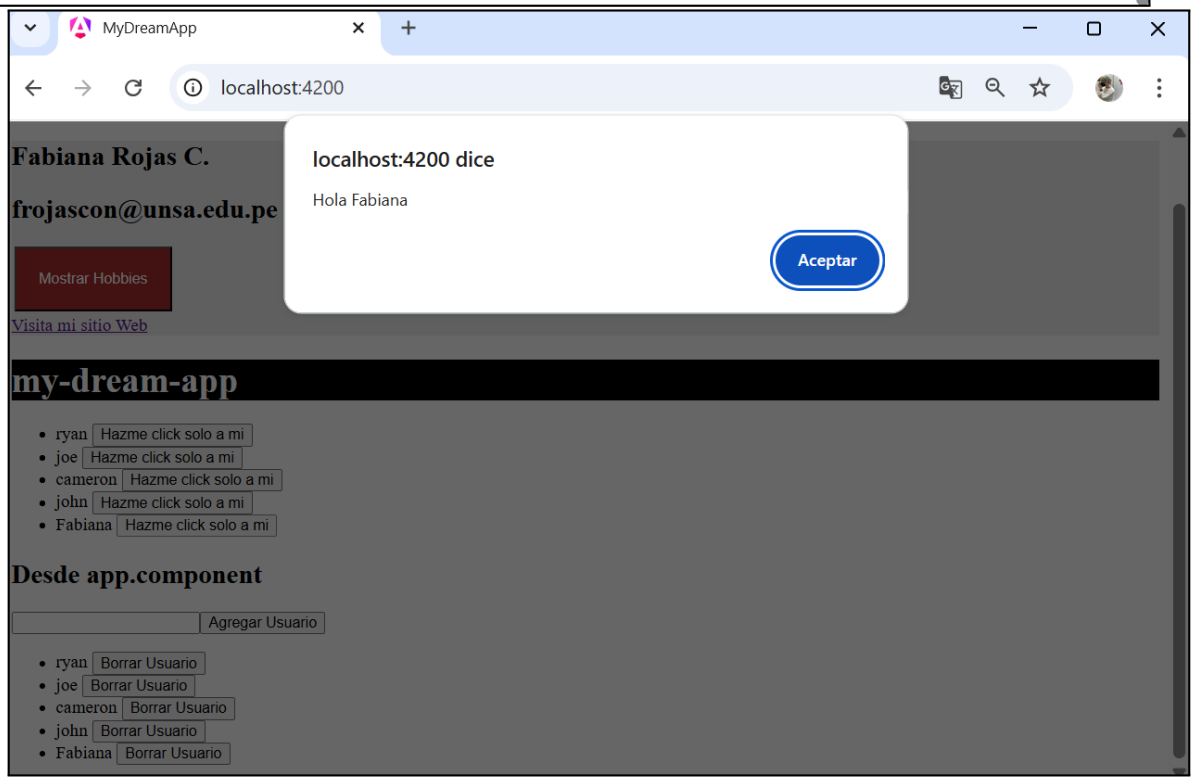


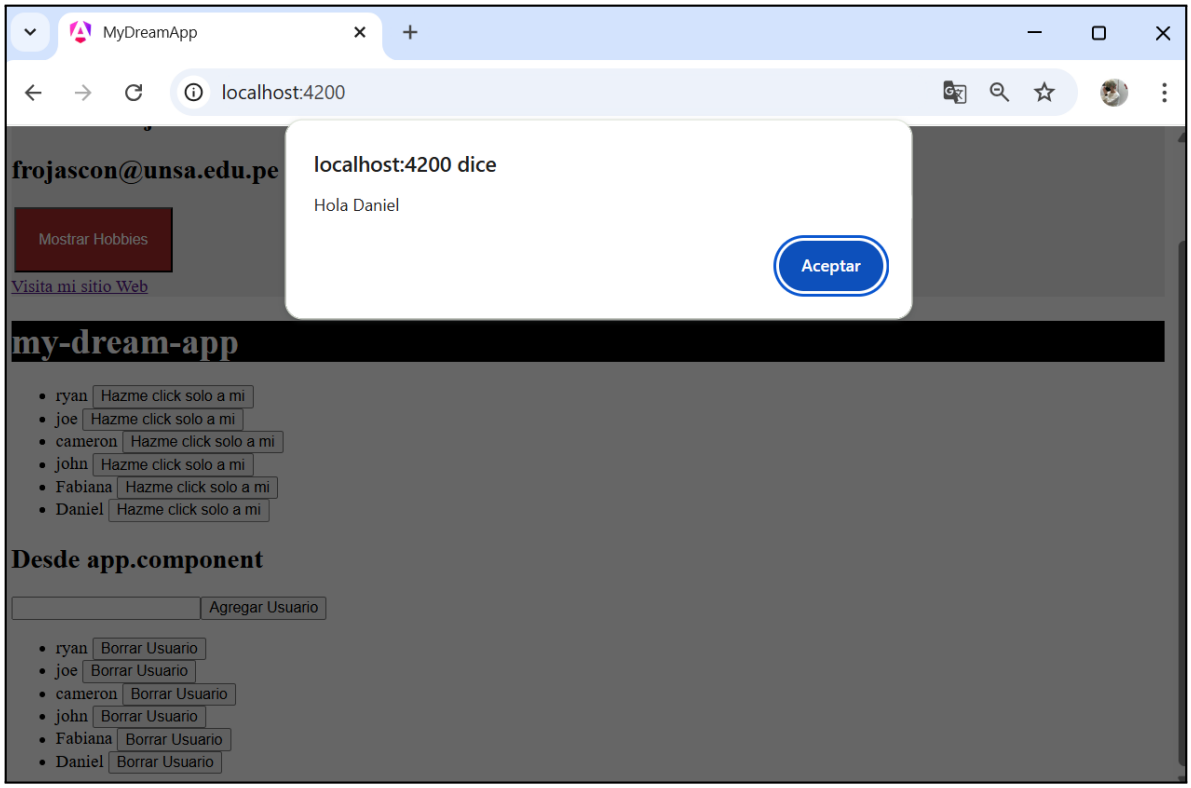

```
Commit 10: "Angular1: Se muestra boton de alerta y se puede agregar y eliminar usuarios"
```

se mejoró la funcionalidad de gestión de usuarios mediante el uso de referencias locales en formularios. Se implementó un formulario que permite añadir usuarios a la lista utilizando newUser como referencia directa al input, lo cual se vinculó al método addUser() mediante la directiva (submit). Además, se mantuvo la posibilidad de eliminar usuarios con el botón asociado a la función deleteUser().

Ejecución


MyDreamApp
localhost:4200
Fabiana Rojas C.
frojascon@unsa.edu.pe
Mostrar Hobbies
Visita mi sitio Web
my-dream-app
ryan Hazme click solo a mi
joe Hazme click solo a mi
cameron Hazme click solo a mi
john Hazme click solo a mi
Desde app.component
Agregar Usuario
ryan Borrar Usuario
joe Borrar Usuario
cameron Borrar Usuario
john Borrar Usuario

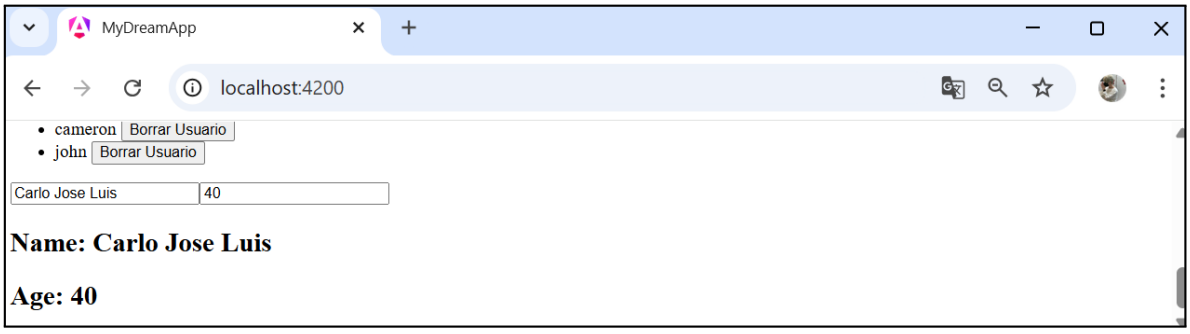

MyDreamApp
localhost:4200
localhost:4200 dice
Hola Fabiana
Aceptar
Fabiana Rojas C.
frojascon@unsa.edu.pe
Mostrar Hobbies
Visita mi sitio Web
my-dream-app
ryan Hazme click solo a mi
joe Hazme click solo a mi
cameron Hazme click solo a mi
john Hazme click solo a mi
Fabiana Hazme click solo a mi
Desde app.component
Agregar Usuario
ryan Borrar Usuario
joe Borrar Usuario
cameron Borrar Usuario
john Borrar Usuario
Fabiana Borrar Usuario



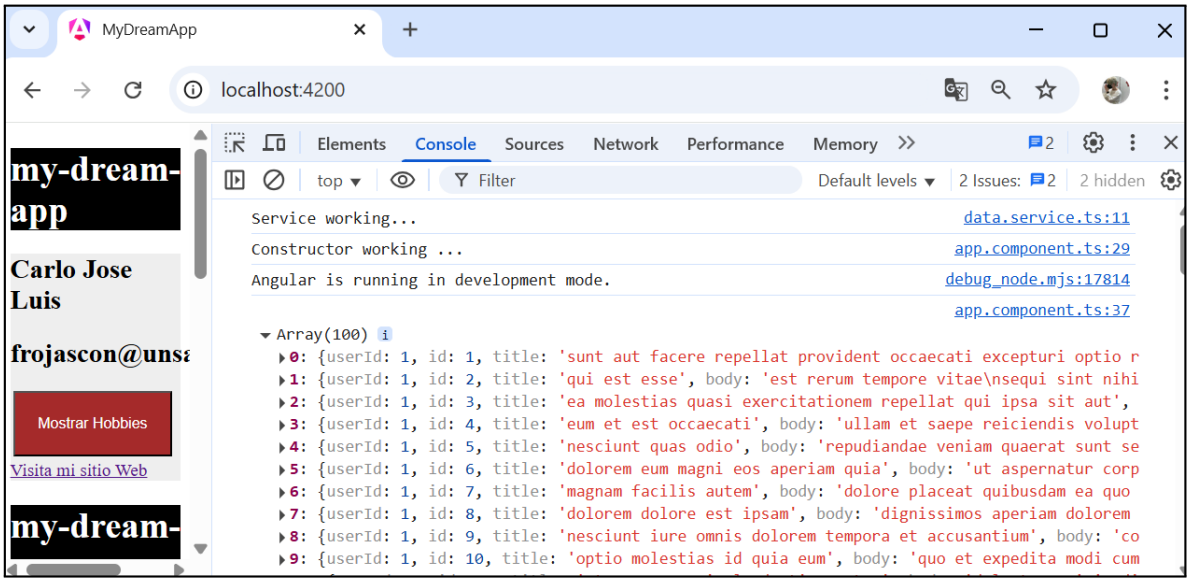

```
Commit 11: "Angular2 - Diapositiva 4: Haciendo DataBinding con datos del formulario, se
    realizaron las indicaciones."
```

se implementó la funcionalidad de data binding utilizando el formulario para capturar y reflejar datos del usuario en tiempo real. Para ello, se incorporó FormsModule en el componente standalone y se utilizaron las directivas [(ngModel)] para enlazar los campos name y age con sus respectivas propiedades en la clase AppComponent. Al escribir en el formulario, los valores se actualizan automáticamente en la vista gracias a la vinculación bidireccional.

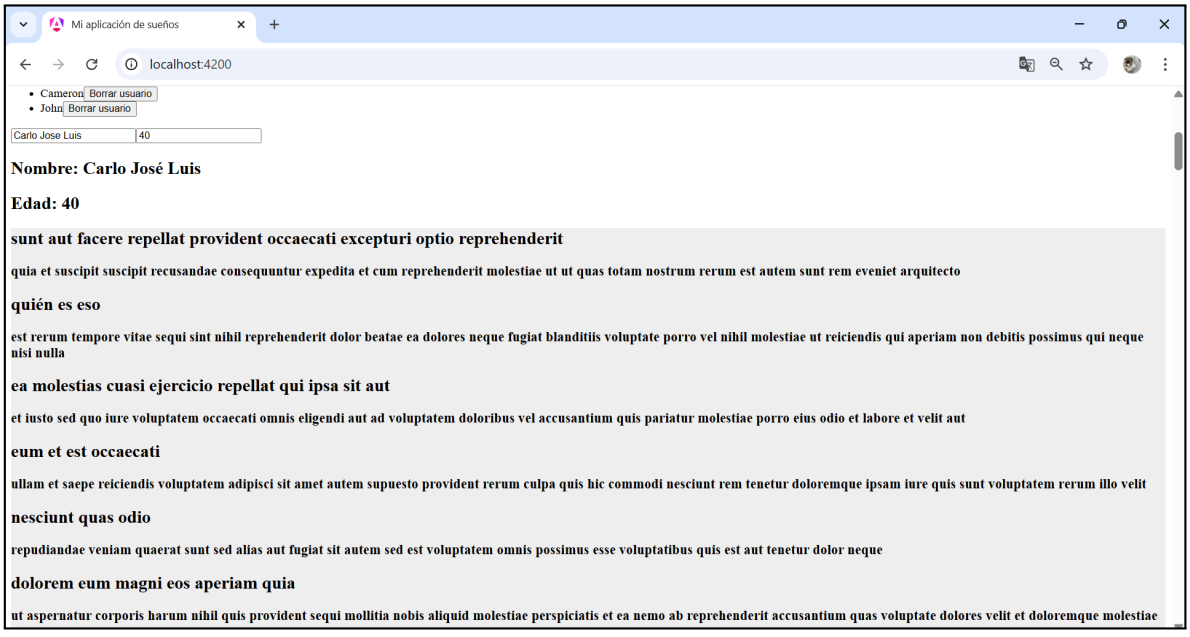
Ejecución



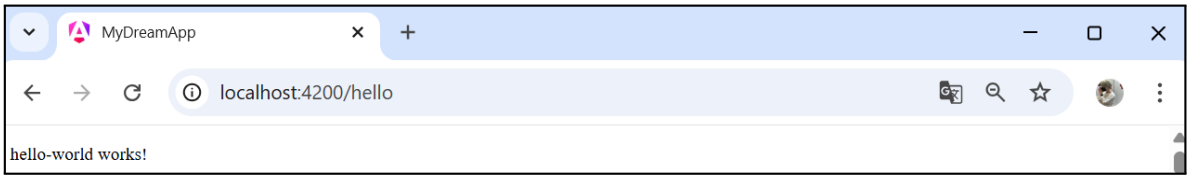

```
Commit 12: "Angular2 - Diapositiva 9,10,11: Insertando un Servicio para llamar a Json."
```

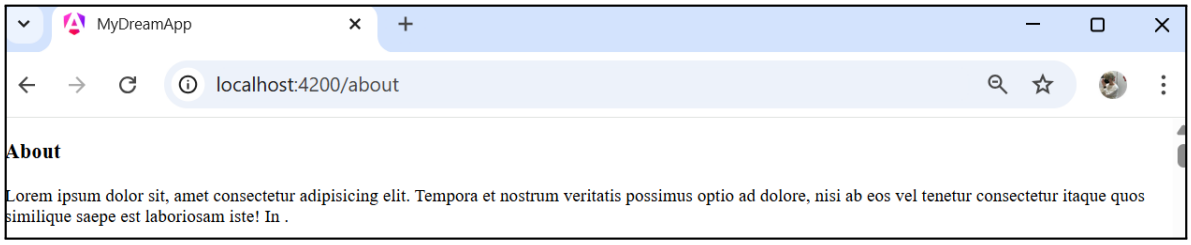


Commit 13: "Angular2 - Diapositiva 12: Insertando un Servicio para llamar a Json."

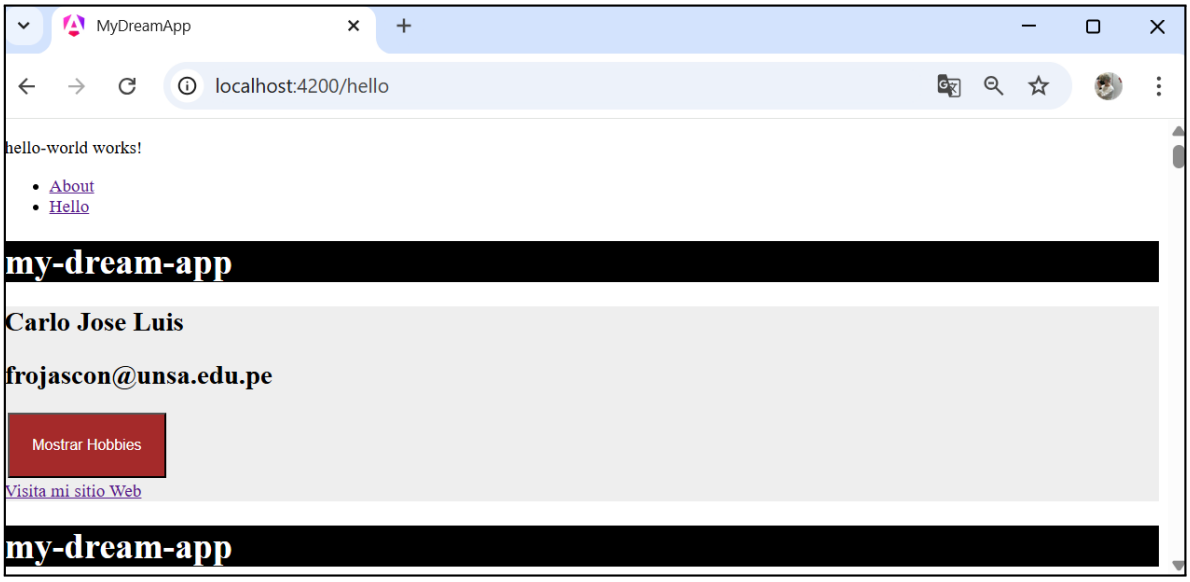


Commit 14: "Angular 2 - Diapositivas 15,16,17: Insertando Ruteos /about y /hello al proyecto para navegar a traves de la url"






```
Commit 15: "Angular2 - Diapositiva 18: Se incorporo un menu para ruteos "
```



6. Referencias

- https://docs.google.com/presentation/d/1OaXlgzMSVr1M2mOqtqbT79x6ynWZjMdT/edit?slide=id.p1#slide=id.p1
- https://docs.google.com/presentation/d/1__PTJpPALSxu0leYlpGcqNKrzz4q2TZf/edit?slide=id.p1#slide=id.p1