|  |
| --- |
| **SPRINT 2: Repositorio de Código - Diseño Base de Datos** |

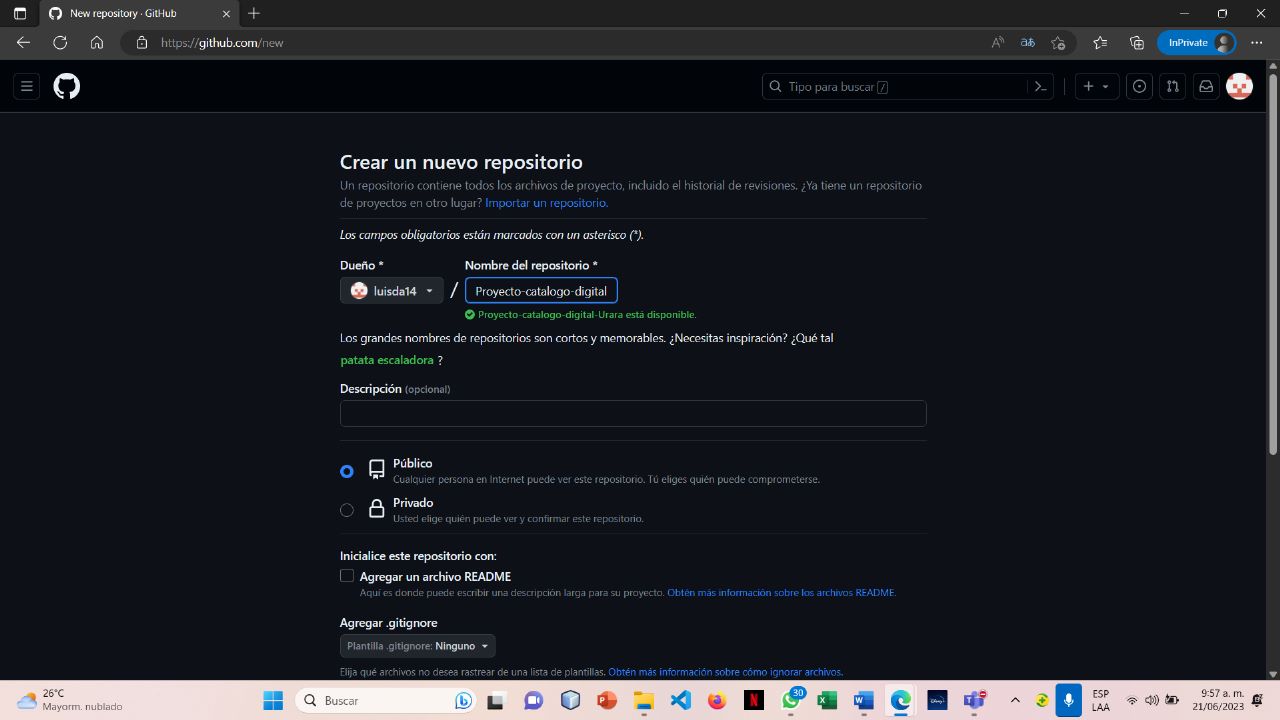
|  |  |
| --- | --- |
| **Identificación Proyecto** | |
| Nombre Proyecto: | Desarrollo de un catálogo digital para ropa de niña urara |
| Número Equipo: | 4 |
| **Integrantes del equipo** | |
| Rol  (Líder-Desarrollador – Cliente) | Nombre |
| Product Owner | Mónica López, Karla Muñoz |
| Scrum Master | Andrés Sánchez |
| Scrum Team | Luis David Caballero, Cristian Camilo Caicedo |
|  |  |
|  |  |
|  |  |
|  |  |

|  |
| --- |
| **Repositorio de Código GitLab o GitHub** |

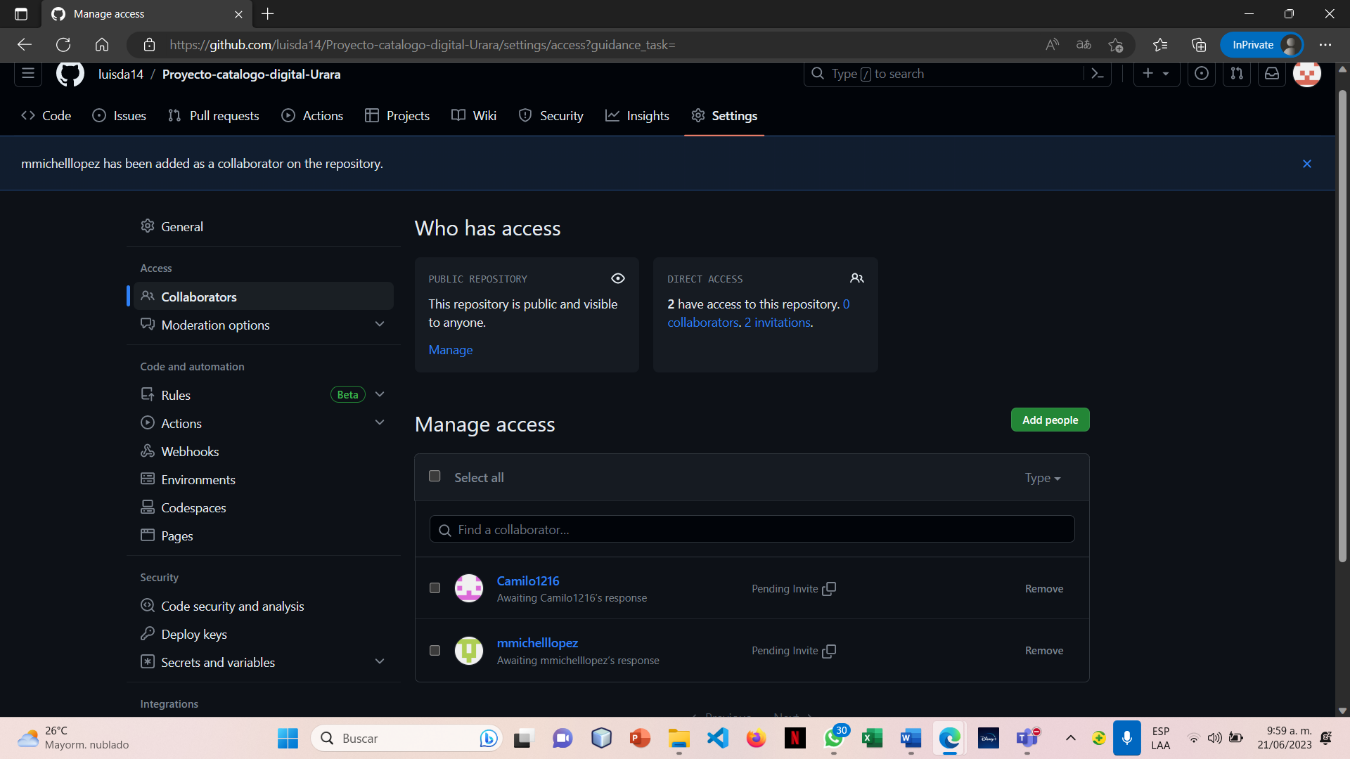
Como evidencia del repositorio de código, creado con GitLab o GitHub, además de la URL del repositorio, se debe presentar capturas de pantalla donde se visualicen aspectos:

Enlace repositorio:

* Creación del proyecto del repositorio.



* Integrantes del equipo invitados.



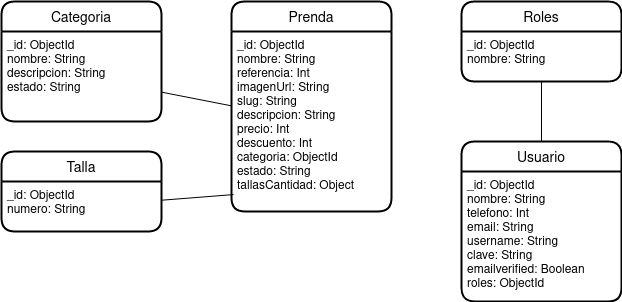
* Evidencia de la realización de alguna actualización (commit), donde se visualice la actualización y el historial de actualizaciones (Versiones)

|  |
| --- |
| **Diseño de la Base de Datos (Proceso de normalización)** |

Como evidencia del Diseño de la Base de Datos, se debe presentar el proceso de normalización efectuado (Formas Normales).

|  |
| --- |
| **Diseño de la Base de Datos no SQL** |

Se debe presentar:



|  |
| --- |
| **Esquema de la Base de Datos no SQL** |

• Categoria

const categoriaSchema = new Schema({

nombre: String,

descripcion: String,

estado: String

},{

timestamps: true,

versionKey: false

});

• Prenda

const tallaCantidadSchema = new Schema({

talla: {

ref: "Talla",

type: String

},

cantidad: Number

},{

\_id: false,

versionKey: false

})

const prendaSchema = new Schema({

nombre: String,

referencia: {

type: Number,

unique: true

},

slug: String,

imagenUrl: {

type: Schema.Types.Array

},

descripcion: String,

tallasCantidad: [tallaCantidadSchema],

precio: Number,

descuento: Number,

estado: String,

categoria: [{

ref: "Categorias",

type: Schema.Types.ObjectId

}]

},{

timestamps: true,

versionKey: false

})

• Roles

const rolesSchema = new Schema({

nombre: String

},{

versionKey: false

})

• Talla

const tallaSchema = new Schema({

numero: String

},{

versionKey: false

})

• Usuario

const usuarioSchema = new Schema<IUser, IUserModel>({

nombre: {

type: String

},

telefono: {

type: Number

},

email: {

type: String,

unique: true,

required: true

},

username: {

type: String,

unique: true

},

clave: {

type: String,

required: true

},

emailverified: {

type: Boolean,

default: false

},

roles: [{

ref: "Role",

type: Schema.Types.ObjectId

}]

},{

timestamps: true,

versionKey: false

})

|  |
| --- |
| **Diagrama casos de uso** |

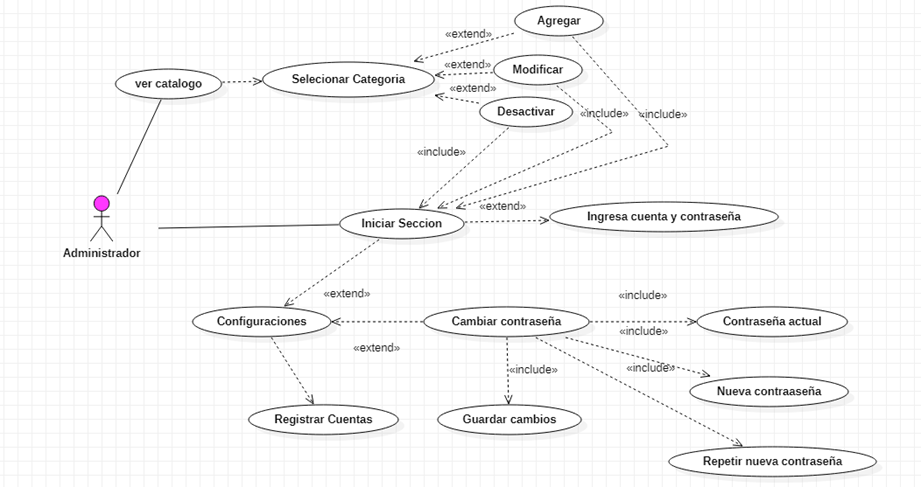


Ilustración : Diagrama casos de uso administrador

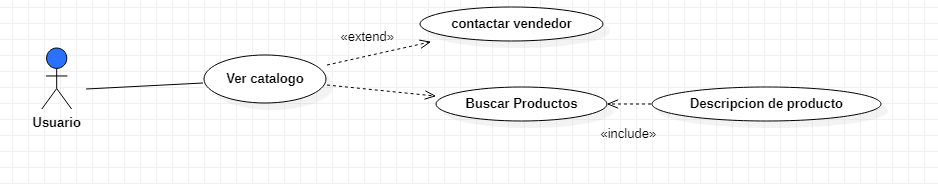


Ilustración : Diagrama casos de uso usuario

|  |
| --- |
| **Diagrama de secuencia** |

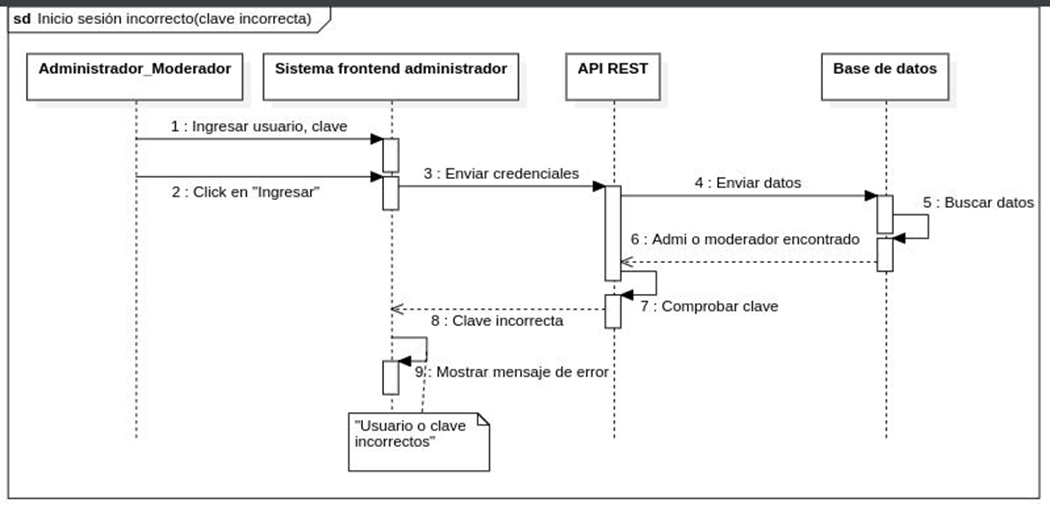


Ilustración : Inicio sesión incorrecto(clave incorrecta)

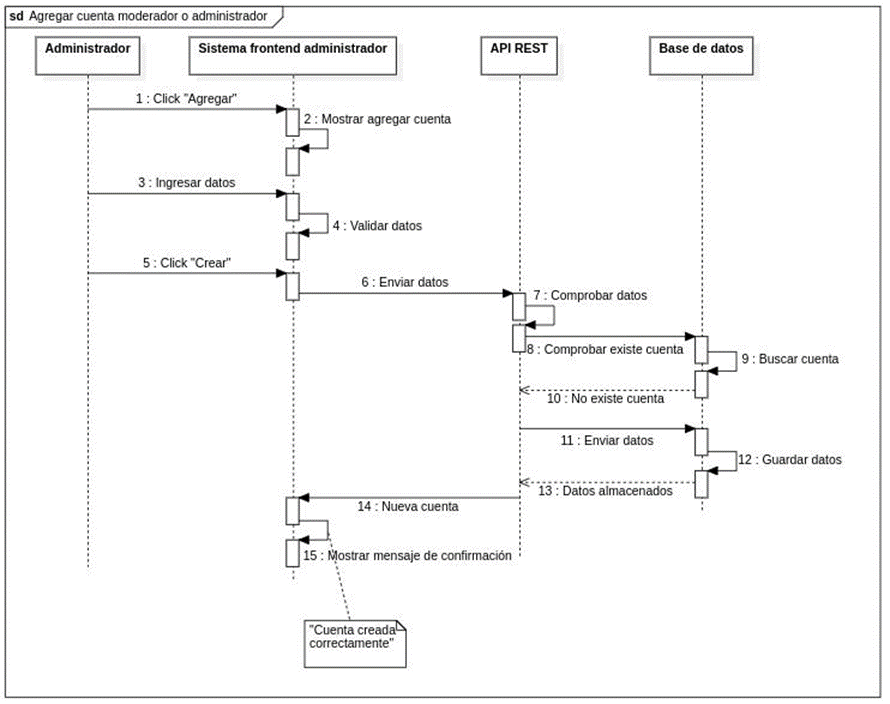


Ilustración : Agregar cuenta moderador o administrador

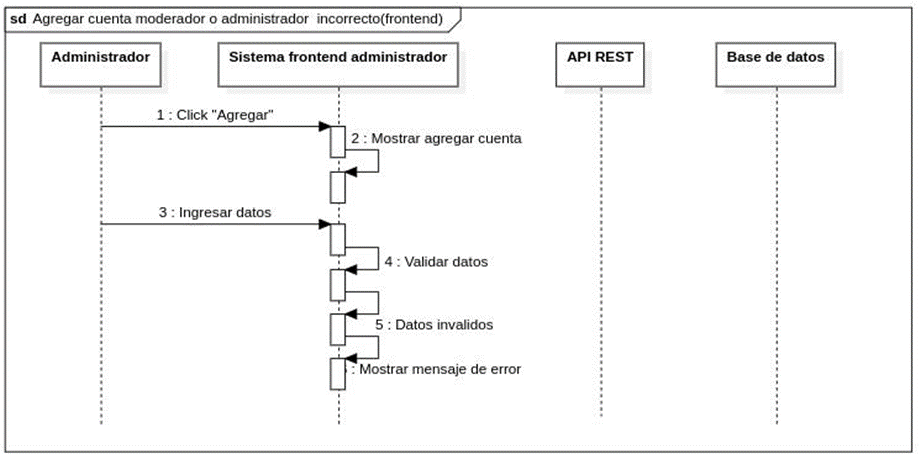


Ilustración : Agregar cuenta moderadora o administrador incorrecto(frontend)

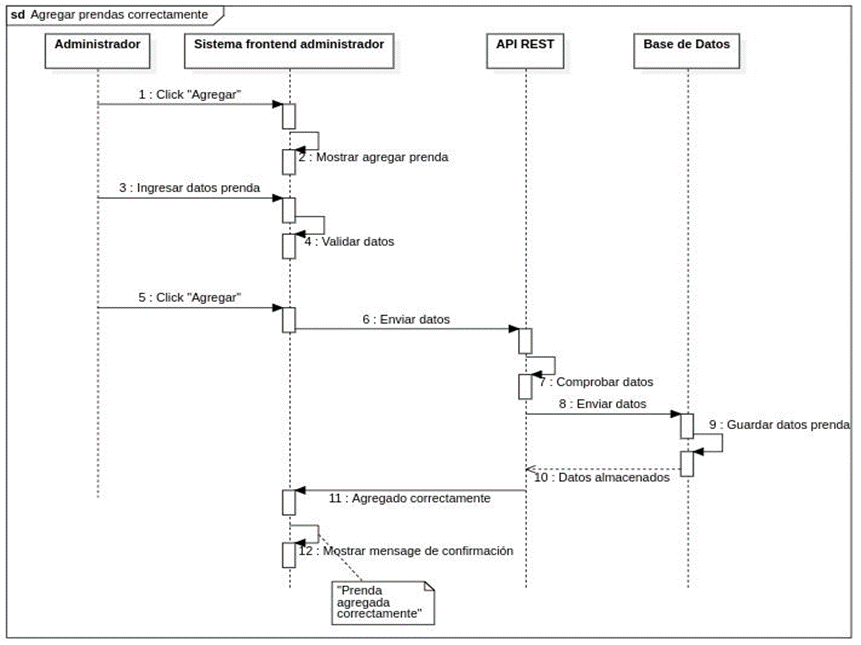


Ilustración : Agregar Prendas Correctamente

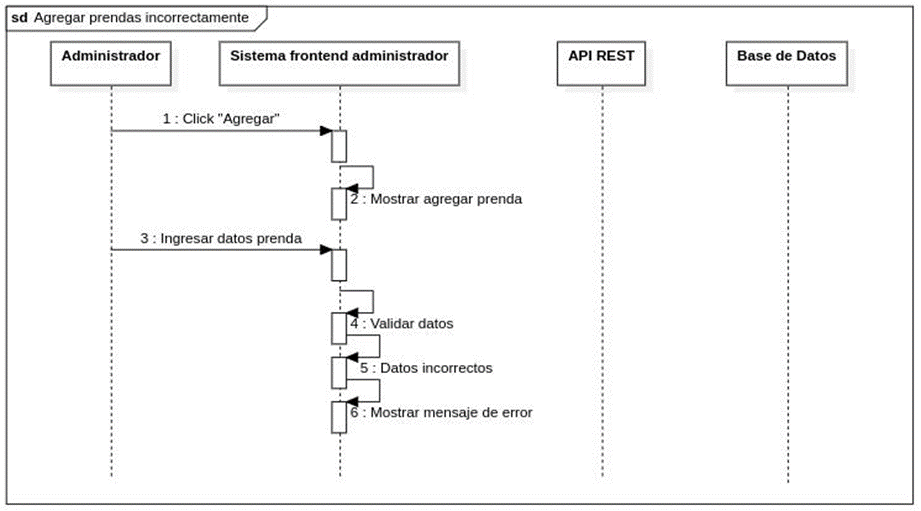


Ilustración : Agregar Prendas Incorrectamente

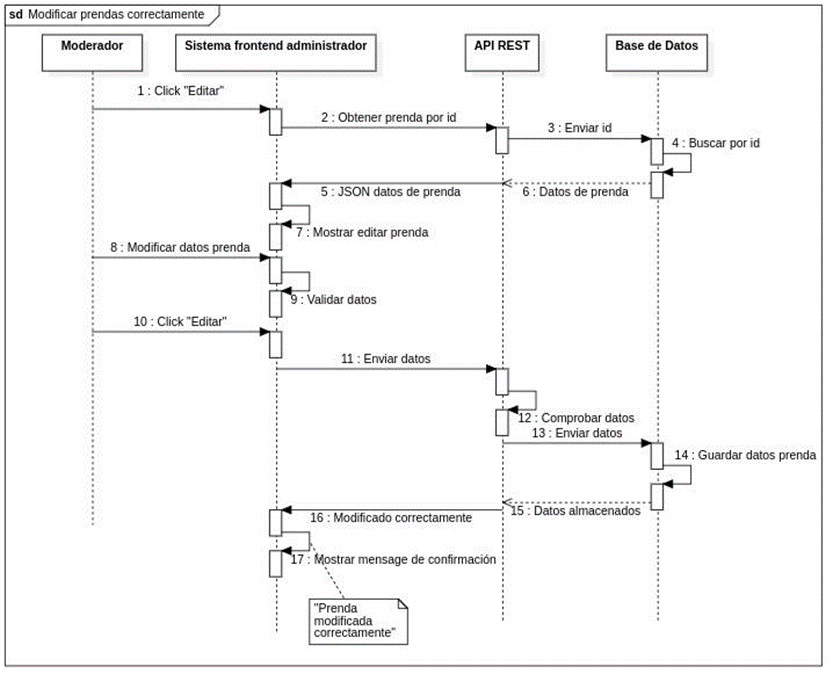


Ilustración : Modificar Prendas Correctamente

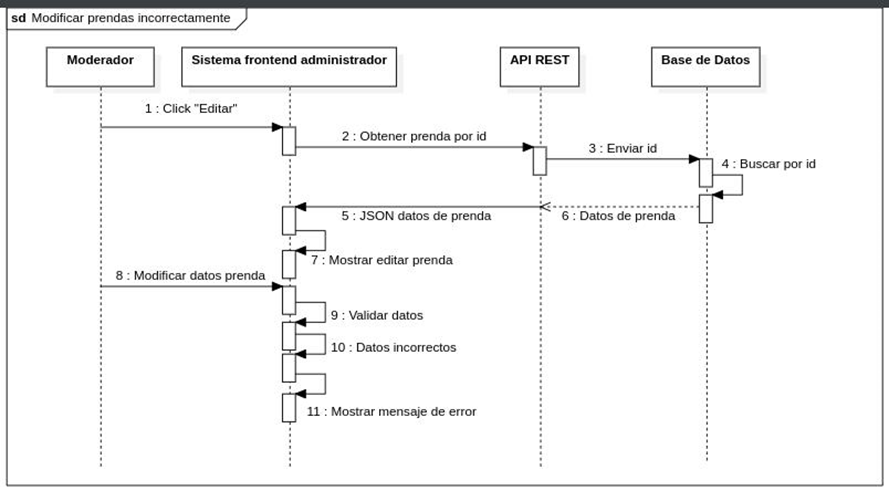


Ilustración : Modificar Prendas Incorrectamente

|  |
| --- |
| **Evidencia JIRA (Seguimiento del proyecto)** |

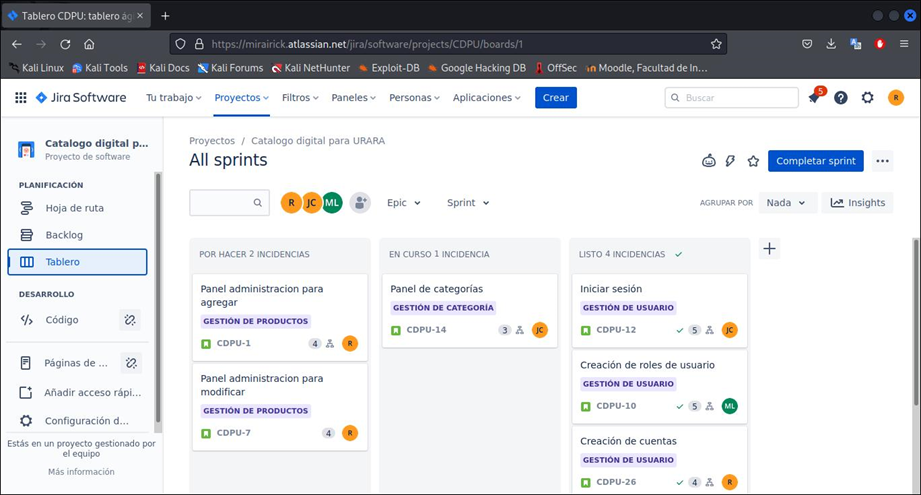


Ilustración : Todos los Sprint

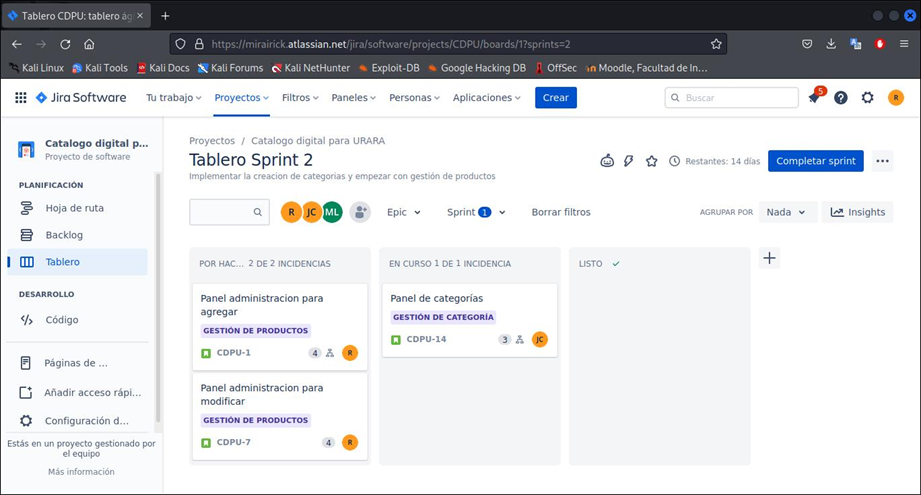


Ilustración : Segundo sprint