1. Anexos

1.1. Manual para puesta en marcha de la aplicación

A continuación, se detallan los pasos necesarios para descargar el código desde el repositorio, proceder con la compilación, la configuración del servidor y la forma de publicar el archivo compilado "***.war***" para poder con esto acceder a los servicios web implementados con vulnerabilidades. Es necesario recalcar que las pruebas de este código deben ser realizadas en equipos locales para no ser víctimas de ataques.

1.1.1. Requisitos

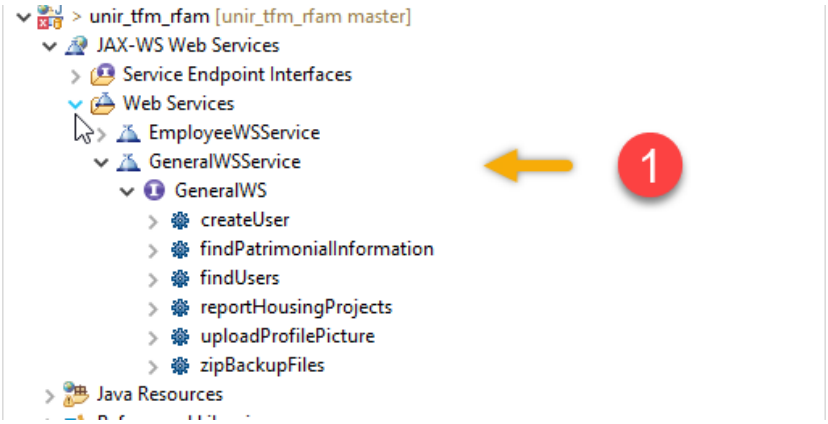
- Java 1.8 o superior
- PostgreSql 9.6 o superior
- Software que permita gestionar el control de versiones GIT
- Eclipse IDE 2020-06
- WildFly 17.0.1 Final

1.1.2. Estructura del proyecto

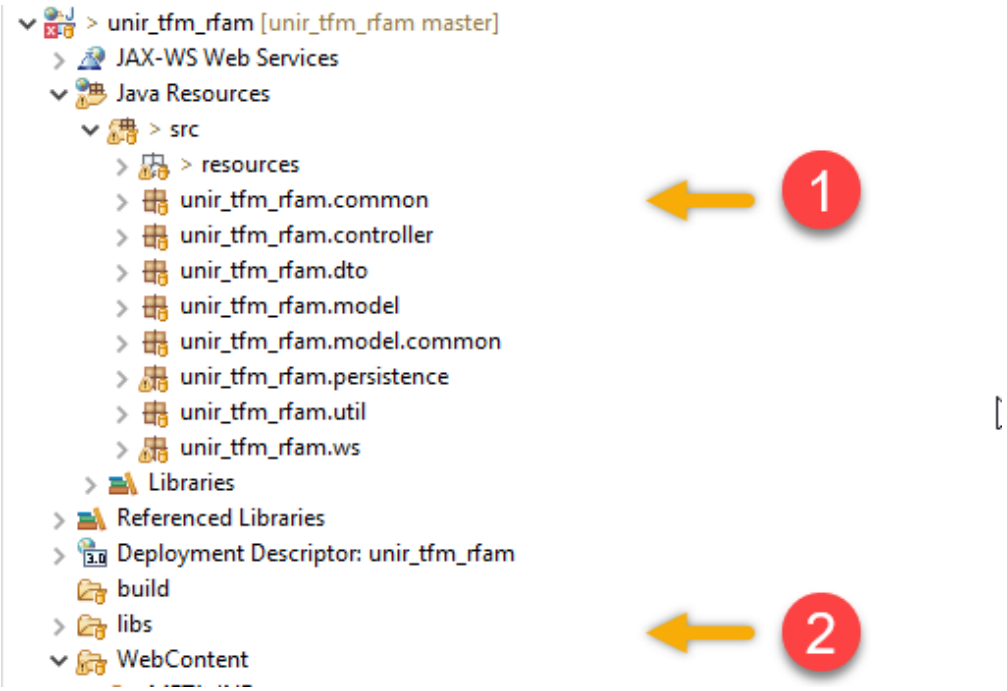
El código se encuentra publicado en GitHub bajo el siguiente enlace: https://github.com/richardmijo/unir_tfm_rfam.git, es un proyecto público por lo que se puede acceder sin ningún problema.

- Se puede realizar la descarga del mismo desde la consola usando el comando ***git clone https://github.com/richardmijo/unir_tfm_rfam.git***
- Se puede usar también un cliente con interfaz gráfica para el cual se usará la siguiente URL: ***https://github.com/richardmijo/unir_tfm_rfam.git***

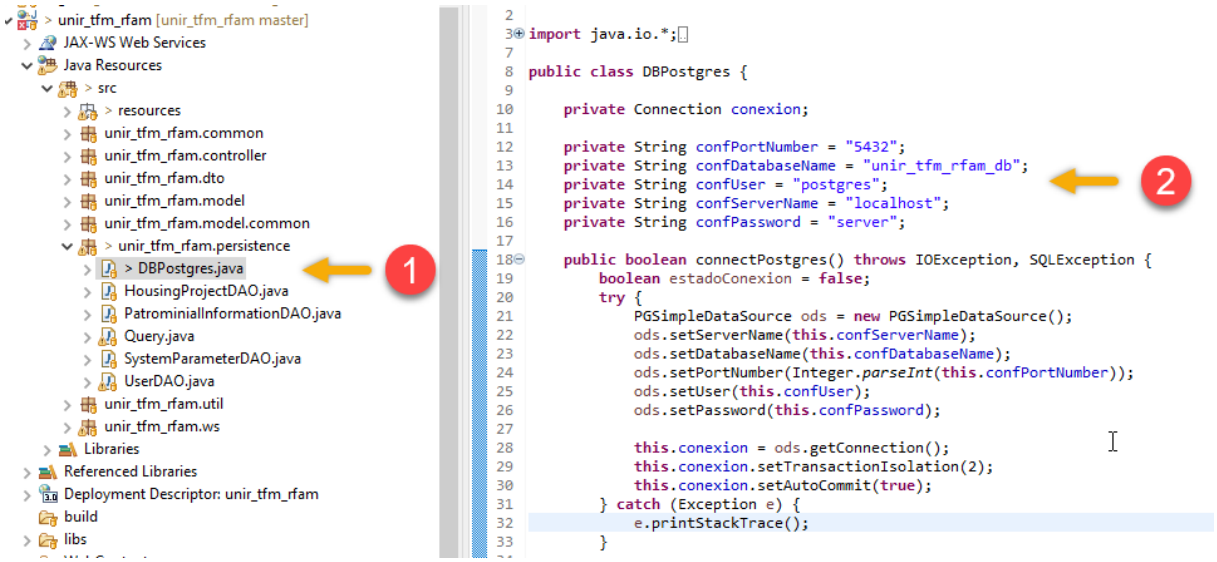
En la imagen 47, se detallan los métodos implementados, se los puede observar en el punto 1, bajo ***GeneralWSService.***



En la imagen 48, se puede observar la estructura de paquetes, en la que se hace referencia al controlador "***controller***", al modelado "***model***" y a la capa de acceso a datos "***persistence***", en el punto 2, está la carpeta "***libs***" que incluye todas las librerías adicionales usadas.



Es necesario configurar el acceso a la base de datos, para esto es necesario ubicar la clase "DBPostgres.java", en el punto 2 de la imagen, se pueden ajustar los parámetros necesarios



1.1.3. Base de datos

Se detalla a continuación el script necesario para generar la base de datos, es necesario recalcar que el usuario por defecto utilizado es "***postgres"***, también se deja un respaldo de la base de datos con información en el repositorio GIT, para que el mismo pueda ser restaurado.

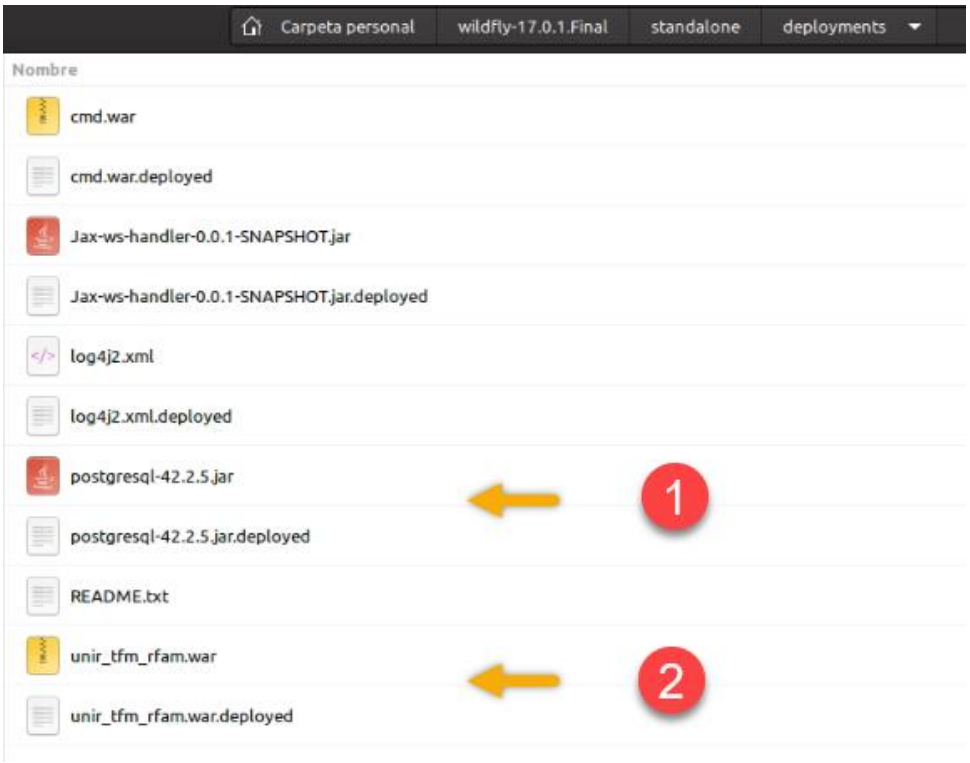
```
/*
PostgreSQL Backup
Database: unir_tfm_rfam_db/public
Backup Time: 2020-09-18 15:42:46
*/

DROP TABLE IF EXISTS "public"."_user";
DROP TABLE IF EXISTS "public"."authority";
DROP TABLE IF EXISTS "public"."bank_information";
DROP TABLE IF EXISTS "public"."debt_information";
DROP TABLE IF EXISTS "public"."housing_project";
DROP TABLE IF EXISTS "public"."patrimonial_information";
DROP TABLE IF EXISTS "public"."postulant";
DROP TABLE IF EXISTS "public"."system_parameter";
DROP TABLE IF EXISTS "public"."user_authority";
CREATE TABLE "_user" (
  "name" varchar COLLATE "pg_catalog"."default",
  "surname" varchar COLLATE "pg_catalog"."default",
  "dni" varchar(15) COLLATE "pg_catalog"."default",
  "email" varchar(80) COLLATE "pg_catalog"."default",
  "username" varchar(20) COLLATE "pg_catalog"."default",
  "password" varchar COLLATE "pg_catalog"."default",
  "isactive" bool,
  "id" int8 NOT NULL,
  "profile_picture_url" varchar(100) COLLATE "pg_catalog"."default",
  "profile_picture_name" varchar(100) COLLATE "pg_catalog"."default"
)
;
ALTER TABLE "_user" OWNER TO "postgres";
CREATE TABLE "authority" (
  "name" varchar(50) COLLATE "pg_catalog"."default" NOT NULL,
  "id" int8 NOT NULL
```

```
)
;
ALTER TABLE "authority" OWNER TO "postgres";
CREATE TABLE "bank_information" (
  "id" int8 NOT NULL,
  "name" varchar(100) COLLATE "pg_catalog"."default",
  "account_number" varchar(20) COLLATE "pg_catalog"."default",
  "account_type" varchar(15) COLLATE "pg_catalog"."default",
  "balance" numeric(19,2),
  "patrimonial_information_id" int8
)
;
ALTER TABLE "bank_information" OWNER TO "postgres";
CREATE TABLE "debt_information" (
  "id" int8 NOT NULL,
  "institution_name" varchar(100) COLLATE "pg_catalog"."default",
  "start_date" date,
  "debt_term" int8,
  "amount" numeric(19,2),
  "obligation_type" varchar(25) COLLATE "pg_catalog"."default",
  "credit_card_number" varchar(19) COLLATE "pg_catalog"."default",
  "patrimonial_information_id" int8
)
;
ALTER TABLE "debt_information" OWNER TO "postgres";
COMMENT ON COLUMN "debt_information"."institution_name" IS 'nombre de la
institución financiera';
COMMENT ON COLUMN "debt_information"."debt_term" IS 'plazo en meses';
COMMENT ON COLUMN "debt_information"."obligation_type" IS 'tipo de
obligacion de pago, prestamo, tarjeta crédito';
CREATE TABLE "housing_project" (
  "id" int8 NOT NULL,
  "name" varchar(100) COLLATE "pg_catalog"."default",
  "number" varchar(10) COLLATE "pg_catalog"."default",
  "is_enabled" bool
)
;
ALTER TABLE "housing_project" OWNER TO "postgres";
COMMENT ON COLUMN "housing_project"."id" IS 'Identificador';
COMMENT ON COLUMN "housing_project"."name" IS 'Nombre del proyecto';
CREATE TABLE "patrimonial_information" (
  "id" int8 NOT NULL,
  "presentation_date" date,
  "observation" varchar(255) COLLATE "pg_catalog"."default",
  "housing_project_id" int8,
  "postulant_id" int8
)
;
ALTER TABLE "patrimonial_information" OWNER TO "postgres";
CREATE TABLE "postulant" (
  "id" int8 NOT NULL,
  "name" varchar(100) COLLATE "pg_catalog"."default",
  "dni" varchar(20) COLLATE "pg_catalog"."default",
  "address" varchar(200) COLLATE "pg_catalog"."default",
  "postal_code" varchar(10) COLLATE "pg_catalog"."default",
  "phone_number" varchar(15) COLLATE "pg_catalog"."default",
  "email" varchar(100) COLLATE "pg_catalog"."default"
)
;
ALTER TABLE "postulant" OWNER TO "postgres";
```

```
CREATE TABLE "system_parameter" (
  "id" int8 NOT NULL,
  "name" varchar(100) COLLATE "pg_catalog"."default",
  "value" varchar(100) COLLATE "pg_catalog"."default"
)
;
ALTER TABLE "system_parameter" OWNER TO "postgres";
CREATE TABLE "user_authority" (
  "user_id" int8 NOT NULL,
  "authority_id" int8 NOT NULL
)
;
ALTER TABLE "user_authority" OWNER TO "postgres";
ALTER TABLE "_user" ADD CONSTRAINT "_user_pk" PRIMARY KEY ("id");
ALTER TABLE "authority" ADD CONSTRAINT "authority_pk" PRIMARY KEY ("id");
ALTER TABLE "bank_information" ADD CONSTRAINT "bank_information_pk" PRIMARY
KEY ("id");
ALTER TABLE "debt_information" ADD CONSTRAINT "debt_information_pk" PRIMARY
KEY ("id");
ALTER TABLE "housing_project" ADD CONSTRAINT "housing_project_pk" PRIMARY
KEY ("id");
ALTER TABLE "patrimonial_information" ADD CONSTRAINT
"patrimonial_information_pk" PRIMARY KEY ("id");
ALTER TABLE "postulant" ADD CONSTRAINT "postulant_pk" PRIMARY KEY ("id");
ALTER TABLE "bank_information" ADD CONSTRAINT "bank_information_pi_fk"
FOREIGN KEY ("patrimonial_information_id") REFERENCES
"public"."patrimonial_information" ("id") ON DELETE NO ACTION ON UPDATE NO
ACTION;
ALTER TABLE "debt_information" ADD CONSTRAINT "debt_information_pi_fk"
FOREIGN KEY ("patrimonial_information_id") REFERENCES
"public"."patrimonial_information" ("id") ON DELETE NO ACTION ON UPDATE NO
ACTION;
ALTER TABLE "patrimonial_information" ADD CONSTRAINT
"patrimonial_information_hp_fk" FOREIGN KEY ("housing_project_id")
REFERENCES "public"."housing_project" ("id") ON DELETE NO ACTION ON UPDATE
NO ACTION;
ALTER TABLE "patrimonial_information" ADD CONSTRAINT
"patrimonial_information_post_fk" FOREIGN KEY ("postulant_id") REFERENCES
"public"."postulant" ("id") ON DELETE NO ACTION ON UPDATE NO ACTION;
ALTER TABLE "user_authority" ADD CONSTRAINT "user_authority_a_fk" FOREIGN
KEY ("authority_id") REFERENCES "public"."authority" ("id") ON DELETE NO
ACTION ON UPDATE NO ACTION;
ALTER TABLE "user_authority" ADD CONSTRAINT "user_authority_u_fk" FOREIGN
KEY ("user_id") REFERENCES "public"."_user" ("id") ON DELETE NO ACTION ON
UPDATE NO ACTION;
```

1.1.4. Compilar e iniciar aplicación

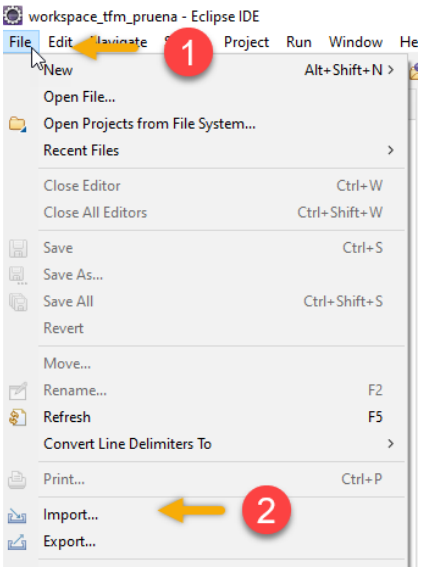
Como servidor de aplicaciones se está usando ***wildfly***, con la finalidad de implementar servicios inseguros se utilizó una conexión directa a la base, es decir sin el uso de ningún ORM, por esta razón es necesario ubicar el driver de PostgreSql bajo el directorio ***wildfly-17.0.1.Final/standalone/deployments***, se puede observar en la imagen siguiente el despliegue del driver en el punto 1 y el despliegue del proyecto en el punto 2.



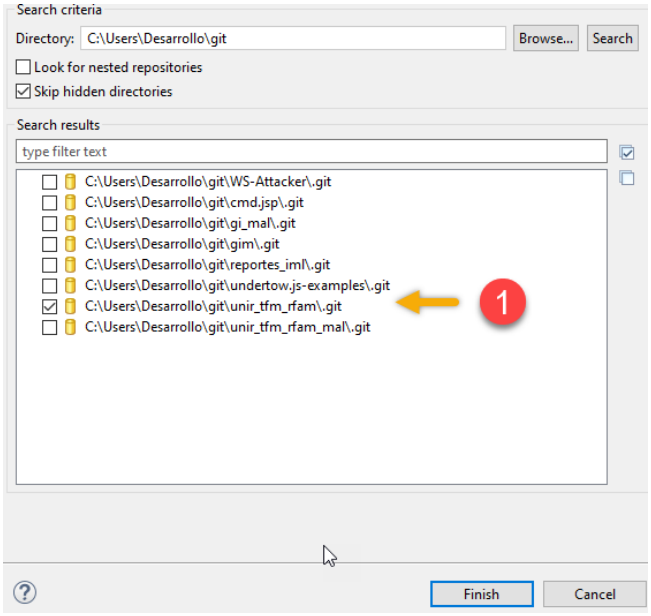
Una vez configurado el driver para la base de datos, es necesario iniciar el servidor de aplicaciones, para lo cual se debe ejecutar el siguiente comando:

```
./wildfly-17.0.1.Final/bin/standalone.sh -c standalone-full.xml
```

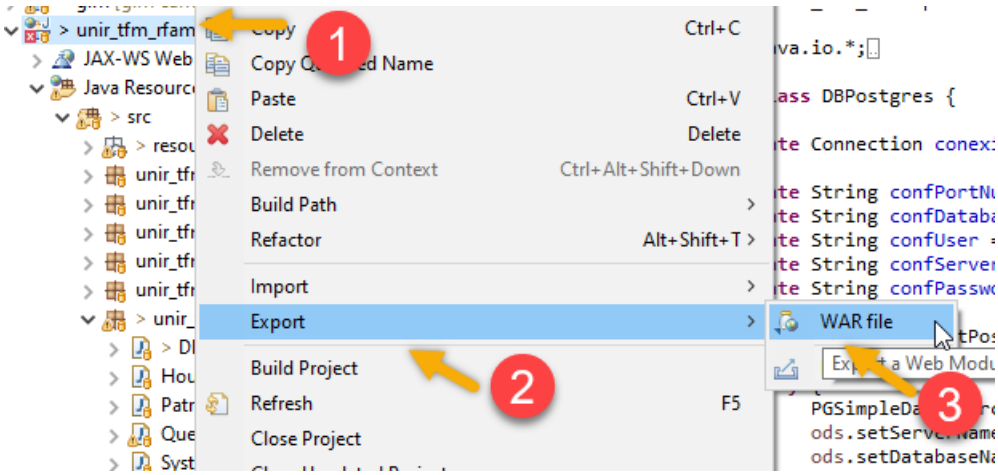
Ajustados los parámetros para acceso a la base de datos y configurado el servidor de aplicaciones, es necesario compilar el proyecto y generar el archivo ***.war*** que será desplegado. Para lo cual se necesita importar el proyecto en Eclipse, bajo "***File***" y luego en "***Import***" se puede importar el proyecto de un repositorio git ya existente.



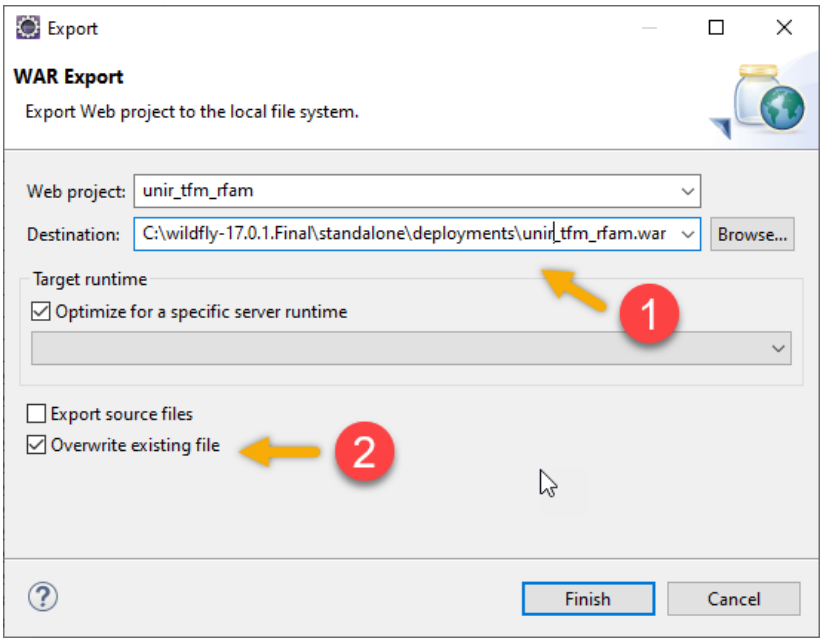
Bastará con seleccionar el proyecto y finalizar la importación, con esto se estará listo para compilar y crear el archivo ***.war***.



Con la finalidad de crear el archivo .war es necesario dar clic derecho sobre el proyecto, luego en exportar y seleccionar "***WAR file***"



Esto hará que se presente una nueva pantalla, donde seleccionaremos la ubicación para generar el archivo ***.war***, en el caso de hacer cambios sobre el código podemos elegir la opción "***Overwrite existing file***" que sobre escribe el archivo ya generado con los nuevos cambios. Por defecto la ruta para crear el archivo es la carpeta ***deployments*** del ***wildfly*** lo que desplegará de forma automática cada vez que el mismo se genere.



Terminado el proceso, se puede acceder al servicio web mediante la siguiente ruta:

```
http://localhost:8080/unir_tfm_rfam/GeneralWS?wsdl
```

1.2. Enlaces de descarga de herramientas usadas en el presente proyecto

Herramienta	Link de descarga
Java	https://www.oracle.com/java/technologies/javase/javase-jdk8-downloads.html
PostgreSQL	https://www.postgresql.org/download/
Eclipse	https://www.eclipse.org/downloads/
Dbeaver	https://dbeaver.io/download/
Soap UI	https://www.soapui.org/downloads/soapui/
WireShark	https://www.wireshark.org/download.html
ReadyAPI	https://www.soapui.org/downloads/download-soapui-pro-trial/