El futuro digital es de todos

MinTIC

Sprints: Desarrollo Aplicaciones Móviles.

Universidad Pontificia Bolivariana

Vigilada Mineducación

Mision TIC 2022

Generalidades

- Recuerda reunirte con tu equipo de trabajo para determinar los acuerdos en la ejecución de las actividades.
- Ten presente avanzar en el valor agregado de tu proyecto. Si resuelven las historias de usuario en poco tiempo, podrán concentrar sus esfuerzos en los componentes diferenciales.
- Recuerda que estamos trabajando sobre funcionalidades básicas relacionadas con la lógica de negocio.
- Debes implementar la primera versión del prototipo del proyecto en un aplicativo móvil y generar el repositorio compartido con tu equipo de trabajo.

Recomendaciones

1. Incluye las vistas, funcionalidades básicas y elementos de valor agregado en la primera versión del proyecto.
2. Esta entrega se realizará a través del autocalificador codegrade.
3. Verifica los nombres de los archivos de entrega y la extensión.
4. Ingresen al espacio en plataforma “Actividad: Sprint 3” y sigan las instrucciones.
5. Define la estructura MVC para adelantar el proceso de desarrollo del proyecto móvil sugerido en estos enunciados.

Introducción/Información:

Los seres humanos influyen cada vez mas en el clima y la temperatura de la tierra, según FAO, un 7% de las emisiones totales se produce mediante el desperdicio de alimentos que terminan directamente en la basura. Esto agrega enormes cantidades de gases de efecto invernadero a lo que incurre naturalmente en la atmosfera, aumentando el calentamiento global.

XML es uno de los formatos más utilizados para intercambiar información entre aplicaciones de diferentes plataformas. Son ficheros de texto donde los campos o elementos de información que contienen se delimitan mediante pares de etiquetas. Para parsear esos documentos, es decir, para poder leerlos y tratarlos, empleamos principalmente dos modelos: SAX y DOM.

En esta nueva serie de implementación del reto numero 2 vamos a ver estos dos modelos, haciendo hincapié en el tratamiento del DOM a través del desarrollo del proyecto **no desperdicio de alimentos**.

Objetivos:

- Resolver las historias de usuario propuestas como requisitos del proyecto de aplicaciones móviles asignado durante el ciclo.
- Repasar el concepto actividades, intentos, manejo de formularios y validación de datos.

Contexto:

En los últimos años, han surgido varias plataformas que persiguen acabar con el desperdicio alimentario, tanto para aprovechar mejor los recursos como para intentar reducir el impacto de la comida en el cambio climático.

Historias de usuario:

Identificador Historia#:	HU-05	Título:	Diseño Formulario de registro
Descripción	COMO:	Usuario final	
	QUIERO:	Como usuario requiero poder registrarme en la aplicación	
	PARA:	Mantener mi información a lo largo de la aplicación	
Criterios de aceptación	1. Todos los campos son requeridos (**Nombre, Apellido, sexo, correo, foto, ciudad, celular, usuario, password**) 2. El campo sexo debe de ser una lista desplegable 3. El campo correo debe de ser validado que sea un correo valido con la siguiente nomenclatura (ejemplo@ejemplo.com) 4. El botón tomar foto deberá abrir la cámara del dispositivo para tomar una foto o agregar una de la galería (Este punto es **opcional, no será evaluado por el bot de autocalificación. Consulta con tu tutor para validar la configuración**) 5. El campo contraseña deberá tener como mínimo 8 caracteres alfanuméricos 6. El campo usuario debe ser mayor a 5 caracteres 7. Al presionar el botón guardar deberá almacenar la información en un objeto (clase) global para poder persistir los datos en la aplicación 8. Al guardar la información en el objeto la aplicación lo deberá redirigir al home/inicio de la aplicación		

Sprint backlog

Historia de usuario HU-05

Como usuario requiero poder registrarme en la aplicación

1. Todos los campos son requeridos (**Nombre, Apellido, sexo, correo, foto, ciudad, celular, usuario, password**)

- Todos los campos deben ser validador y que también validar si el usuario ingresa espacios en el mismo

Sprint backlog

Historia de usuario HU-05

Como usuario requiero poder registrarme en la aplicación

2. El campo sexo debe de ser una lista desplegable

- La lista deberá contener las opciones que requiera el sistema en la imagen podrá encontrar un ejemplo
- Si el usuario no selecciona nada el sistema deberá validar si el sexo esta vacío o el usuario no ha seleccionado nada

Sprint backlog

Historia de usuario HU-05

Como usuario requiero poder registrarme en la aplicación (**opcional**)

1. El botón tomar foto deberá abrir la cámara del dispositivo para tomar una foto o agregar una de la galería
   - Deberá agregar los siguientes permisos para poder utilizar la cámara y y acceder a la galería del teléfono

<uses-permission android:name="string" android:maxSdkVersion="integer" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"
android:maxSdkVersion="18" />

Sprint backlog

Historia de usuario HU-05

Como usuario requiero poder registrarme en la aplicación

1. Al presionar el botón guardar deberá almacenar la información en un objeto (clase) global para poder persistir los datos en la aplicación

- Puede crear una clase estática en java para persistir en la aplicación
- Para utilizar la clase: **UsuarioDto usuario = UsuarioDto.getInstance();**
- Para pasar datos al objeto: **usuario.setNombre("Nombre");**
- Para obtener los datos: **usuario. getNombre();**

```java
public class UsuarioDto {

    private String nombre;
    private String apellido;
    private String sexo;
    private String direccion;
    private String correo;
    private String foto;
    private String ciudad;
    private String celular;
    private String usuario;
    private String password;

    private static UsuarioDto instance = new UsuarioDto();

    // Getter-Setters
    public static UsuarioDto getInstance() {
        return instance;
    }

    public static void setInstance(UsuarioDto instance) {
        UsuarioDto.instance = instance;
    }

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }
```

Entrega:

Se debe entregar los siguientes archivos con el formato y nombre específicos

1. **UsuarioDto.java**
2. **FormularioDTO.java**
3. **ControladorFormulario.java**
4. **FormularioInterfaz.java**

Recuerda comentar las líneas de código asociadas al paquete y a las importaciones de otras clases.

Entrega:

Se debe entregar los siguientes archivos con el formato y nombre específicos

1. **UsuarioDto.java**

```java
public class UsuarioDto {

    private String nombre;
    private String apellido;
    private String sexo;
    private String direccion;
    private String correo;
    private String foto;
    private String ciudad;
    private String celular;
    private String usuario;
    private String password;

    private static UsuarioDto instance = new UsuarioDto();

    // Getter-Setters
    public static UsuarioDto getInstance() {
        return instance;
    }

    public static void setInstance(UsuarioDto instance) {
        UsuarioDto.instance = instance;
    }

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }
```

Entrega:

Se debe entregar los siguientes archivos con el formato y nombre específicos

1. **FormularioDTO.java**

```java
package com.upb.nodesperdicio.modelos;

public class FormularioDTO {

    private String editNombres;
    private String editApellidos;
    private String editDireccion;
    private String editCorreo;
    private String editCiudad;
    private String editCelular;
    private String editUsuario;
    private String editPassword;
    private String spSexo;

    public String getEditNombres() {
        return editNombres;
    }

    public void setEditNombres(String editNombres) {
        this.editNombres = editNombres;
    }
```

Entrega:

Se debe entregar los siguientes archivos con el formato y nombre específicos

1. **ControladorFormulario.java**

Recuerda comentar las líneas de código asociadas al paquete y a las importaciones de otras clases.

```java
package com.upb.nodesperdicio.controlador;

import ...

public class ControladorFormulario implements FormularioInterfaz.Controlador {

    private final FormularioInterfaz.View view;

    public ControladorFormulario(FormularioInterfaz.View view) { this.view = view; }

    @Override
    public Boolean validarFormulario(FormularioDTO formularioDTO) {...}

    @Override
    public Boolean usuarioGuardarUsuario(FormularioDTO formularioDTO) {...}

}
```

Entrega:

Se debe entregar los siguientes archivos con el formato y nombre específicos

1. **FormularioInterfaz.java**

Recuerda comentar las líneas de código asociadas al paquete y a las importaciones de otras clases.

```java
package com.upb.nodesperdicio.interfaz;

import com.upb.nodesperdicio.modelos.FormularioDTO;

public interface FormularioInterfaz {

    interface View {
        void validarResultadoFormulario(String editText, String mensaje);
        void respuestaGuardadoUsuario(Boolean respuesta);
    }

    interface Controlador {
        Boolean validarFormulario(FormularioDTO formularioDTO);
        Boolean usuarioGuardarUsuario(FormularioDTO formularioDTO);
    }

    interface Modelo {
    }

}
```

Material de apoyo:

- https://developer.android.com/training/testing/ui-testing/espresso-testing
- https://www.vogella.com/tutorials/Mockito/article.html