

[Indicaciones](#)

[Dominio](#)

[Recuperatorio de TP1: Desktop usando Arena](#)

[Recuperatorio de TP2: Web usando Angular](#)

[Recuperatorio de TP3: Mobile usando Android](#)

Indicaciones

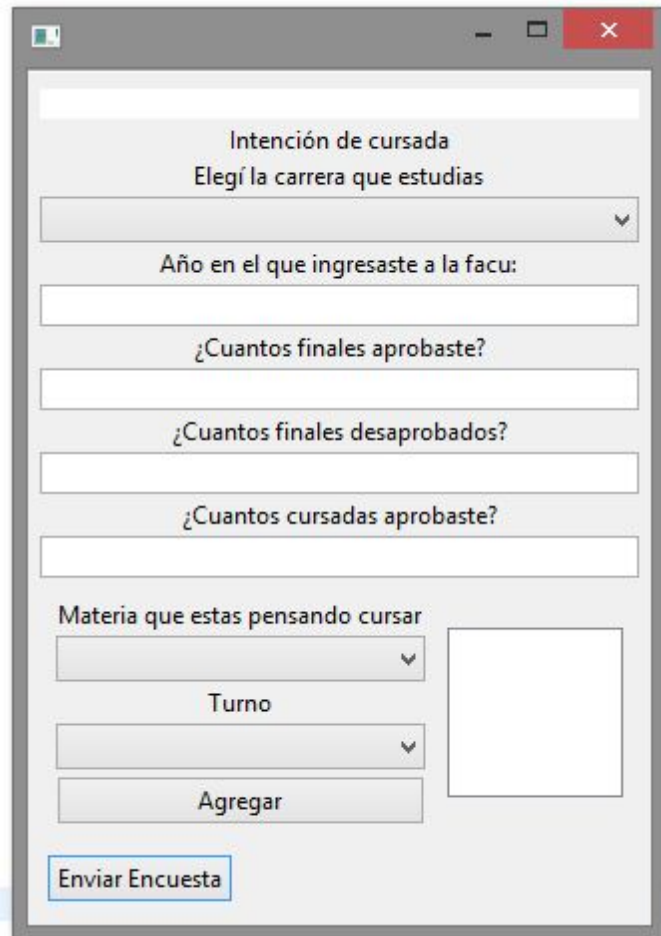
Los trabajos **deberán estar funcionando completamente a la hora de entregar sin errores**. No se aceptarán entregas con errores por lo que pedimos a quienes entregan que si necesitan utilizar las máquinas de la facultad dejen configurado su TP en las mismas con anticipación a la hora de entrega.

Dominio

Para la facu nos pidieron un sistema que ayude a planificar la distribución de cursos, profesores y horarios y detectar posibles cuellos de botella antes del día de la inscripción. Es un sistema de por sí muy complejo por muchas variantes que hay que tener en cuenta, y como tarea del mega proyecto encomendado tenemos que armar una encuesta MUY sencilla para empezar a juntar datos para hacer análisis preliminares. Pusieron la encuesta en funcionamiento y después de un tiempo nos pasaron feedback del comportamiento de la aplicación.

Recuperatorio de TP1: Desktop usando Arena

Pantalla actual



Intención de cursada

Elegí la carrera que estudias

Año en el que ingresaste a la facu:

¿Cuántos finales aprobaste?

¿Cuántos finales desaprobados?

¿Cuántos cursadas aprobaste?

Materia que estas pensando cursar

Turno

Agregar

Enviar Encuesta

Lo que nos dijo el usuario:

1. Agregar mail a cada respuesta de la encuesta, ahora es anonimo y no sabemos cómo distinguir las respuestas
2. En las opciones de turnos dice "MANIANA" debería de decir "Mañana" (sí, con ñ y solo la primer letra en mayúscula)
3. Cuando se agrega una materia para cursar NO se refleja en la pantalla
4. No se están validando los campos obligatorios (carrera, año de ingreso y al menos una materia a la que se piensa anotar)

El técnico que revisó el código nos dijo:

1. El modelo es pésimo:
 - a. Encuesta tiene atributos que sobran, responsabilidades que están solo por la pantalla, variables mal nombradas (carreraSeleccionada, etc) y malas elecciones en los tipos de datos
 - b. Materia confunde el concepto de la materia en sí con la intención de inscripción a un turno ya que tiene turno pero solo tiene datos cuando la persona se quiere inscribir

- c. Falta modelar el concepto de Carrera, ahora se usan solo strings, cuando en realidad las materias dependen de la carrera
 - d. El modelo de GraciasPorResponderWindow carece de todo tipo de sentido
- 2. El uso del framework es pobre
 - a. El manejo que se hace para mostrar los datos en los selectores es absurdo y podrían aprovecharse los bindings y adapters que tiene el framework
 - b. Abrir una nueva ventana para dar una información desaprovecha las ventanas que ya vienen implementadas
 - c. No se hacen las validaciones ni se muestran los errores
 - d. Se hace un mal manejo de bindings en cuanto a la lista de materias aprobadas que provoca que no se refresque la pantalla

Se pide

1. Crear al menos un proyecto que use arena y provea la funcionalidad (la actual y la pedida) buscada:
 - a. Ingresar a la vista que permita responder la encuesta indicando un mail (válido)
 - b. Responder la encuesta. Se deben realizar las validaciones correspondientes y notificar en caso de error.
 - c. Una vez finalizada la encuesta ver la pantalla de agradecimiento
2. Revisar el código actual y hacer los cambios que considere necesario para corregir las críticas que hace el técnico sobre el código actual. Se pueden realizar **todos** los cambios que se deseen (no es necesario mantener la estructura actual del código)

Código de referencia

```
class EncuestaApplication extends Application {

    override protected createMainWindow() {
        new EncuestaWindow(this)
    }

    def static void main(String[] args) {
        new EncuestaApplication().start()
    }
}

class EncuestaWindow extends SimpleWindow<Encuesta> {

    new(WindowOwner parent) {
        super(parent, new Encuesta())
    }

    override protected createFormPanel(Panel mainPanel) {
        new Label(mainPanel).text = "Intención de cursada"
        this.crearInformacionParaPeso(mainPanel)
        this.crearInformacionParaCursadas(mainPanel)
    }

    def crearInformacionParaPeso(Panel panel) {
```

```

        new Label(panel).text = "Elegí la carrera que estudias"
        new Selector<String>(panel) => [
            bindItemsToProperty("carrerasPosibles")
            bindValueToProperty("carreraSeleccionada")
        ]

        new Label(panel).text="Año en el que ingresaste a la facu:"
        new TextBox(panel).bindValueToProperty("anioIngreso")

        new Label(panel).text="¿Cuántos finales aprobaste?"
        new TextBox(panel).bindValueToProperty("finalesAprobados")

        new Label(panel).text="¿Cuántos finales desaprobados?"
        new TextBox(panel).bindValueToProperty("finalesDesaprobados")

        new Label(panel).text="¿Cuántos cursadas aprobaste?"
        new TextBox(panel).bindValueToProperty("cursadasAprobadas")
    }

    def crearInformacionParaCursadas(Panel mainPanel) {
        val columnPanel = new Panel(mainPanel)
        columnPanel.layout = new ColumnLayout(2)

        val nuevaMateria = new Panel(columnPanel)
        nuevaMateria.layout = new VerticalLayout

        new Label(nuevaMateria).text="Materia que estas pensando cursar"
        new Selector<String>(nuevaMateria) => [
            bindItemsToProperty("materiasPosibles")
            bindValueToProperty("materiaSeleccionada")
        ]
        new Label(nuevaMateria).text="Turno"
        new Selector<String>(nuevaMateria) => [
            bindItemsToProperty("turnosPosibles")
            bindValueToProperty("turnoSeleccionado")
        ]
        new Button(nuevaMateria) =>[
            caption = "Agregar"
            onClick = [ |
                modelObject.agregarMateriaSeleccionada()
            ]
        ]

        val materiasAgregadas = new Panel(columnPanel)
        materiasAgregadas.layout = new VerticalLayout

        new List<String>(materiasAgregadas)=> [
            bindItemsToProperty("descripcionMaterias")
        ]
    }

    override protected addActions(Panel panel) {
        new Button(panel) =>[
            caption = "Enviar Encuesta"
        ]
    }

```

```

        onClick = [ | new GraciasPorResponderWindow(this).open]
    ]
}
}

```

```

class GraciasPorResponderWindow extends SimpleWindow<Encuesta> {

    new(WindowOwner parent) {
        super(parent, new Encuesta())
    }

    override protected createFormPanel(Panel mainPanel) {
        new Label(mainPanel).text = "Gracias por responder!"
    }

    override protected addActions(Panel actionsPanel) {
        //nada
    }
}

```

@Observable

@Accessors

```

class Encuesta {

    var List<Materia> materias
    var String carreraSeleccionada
    var String anioIngreso
    var String finalesAprobados
    var String finalesDesaprobados
    var String cursadasAprobadas

    //solo la usamos desde la vista
    Turno turnoSeleccionado
    String materiaSeleccionada

    new(){
        materias = newArrayList
    }

    def agregarMateriaSeleccionada() {
        var materia = new Materia (materiaSeleccionada, turnoSeleccionado)
        materias.add(materia)
    }

    def List<String> getCarrerasPosibles(){
        #["Sistemas - K", "Electronica - Q", "Industrial - Z"].toList
    }

    def List<String> getMateriasPosibles(){
        #["Discreta", "Algebra", "Ingenieria y Sociedad", "Matematica Superior",
"Paradigmas de Programacion", "Algoritmos", "Sintaxis", "Resistencias de
Materiales"].toList
    }
}

```

```

    }

    def List<Turno> getTurnosPosibles(){
        Turno.values.toList
    }

    def List<String> getDescripcionMaterias(){
        materias.map[''«it.descripcion» («it.turno.name»)''].toList
    }
}

@Observable
@Accessors
class Materia {
    Turno turno
    String descripcion

    new(String descripcion, Turno turno){
        this.turno = turno
        this.descripcion = descripcion
    }
}

public enum Turno {
    MANIANA,
    TARDE,
    NOCHE
}

```

Recuperatorio de TP2: Web usando Angular

Screenshots de la aplicación



Intención de cursada

Elegí la carrera que estudias

Sistemas (K) ▼

Año en el que ingresaste a la facu

¿Cuántos finales aprobaste?

¿Cuántos finales desaprobaste?

¿Cuántos materias aprobaste la cursada?

¿A que materias pensas Anotarte el proximo cuatrimestre?

Materia

Turno

Agregar

Discreta - Mañana

Algebra - Tarde

Enviar Respuestas

Lo que nos dijo el usuario:

1. Tenemos respuestas de estudiantes duplicadas, necesitamos que se permita una sola respuesta por cada usuario, asumimos que cada mail es de un solo usuario.
2. No está funcionando bien la validación de obligatoriedad de los campos, particularmente la gente de sistemas de alguna forma completan la encuesta sin completar los datos obligatorios

Después de un tiempo... finalmente llegó el feedback del sector de calidad interna de nuestra empresa y como resultado nos informaron de un error que debemos corregir:

- La validación implementadas en el controller no cumplen con nuestro estándar de calidad, debe de distribuir correctamente las responsabilidades en los objetos de dominio.

Se pide

1. Crear un proyecto web que use xtrest y angular, y provea la funcionalidad deseada en la aplicación
 - a. Ingresar a la vista que permita responder la encuesta
 - b. Responder la encuesta, impactando en el servidor (al menos en memoria). Se deben realizar las validaciones correspondientes y notificar en caso de error.
 - c. Una vez finalizada la encuesta ver la pantalla de agradecimiento
2. Revisar el código actual y hacer los cambios que considere necesario para que cumpla con los conceptos vistos en la materia.
3. En particular para los comentarios del usuario y del sector de arquitectura:
 - a. Para cada punto que mencionó el usuario:
 - i. identificar las causas de los problemas (respuestas duplicadas o campos obligatorios no presentes) detectados
 - ii. implementar una solución que resuelva el problema o cumpla con el requerimiento pedido.
 - b. Para el punto detectado por el sector de arquitectura
 - i. relacionar el código del controller con el concepto de MVC
 - ii. implementar los cambios que sean necesarios

Código de referencia

HTML

```
<!-- login.html -->
<div class="home">
  <div class="mensaje">
    <span>Estamos tratando de mejorar la oferta de horarios de la facu</span>
    <span>Ayudanos respondiendo la encuesta, solo te tomara 10 minutos</span>
  </div>
  <div class="login">
    <input type="text" placeholder="Ingresa tu mail" ng-model="user.mail" />
    <button type="submit" ng-click="autenticar()">Responder</button>
  </div>
</div>

<!-- gracias.html -->
<div class="home">
  <div class="mensaje mensaje-final">
    <span>Gracias por participar con tus respuestas!</span>
  </div>
</div>

<!-- responder.html -->
<div class="container">
  <h3>Intención de cursada</h3>
```



```

<span>Elegí la carrera que estudias</span>
<select class="form-control" ng-options="carrera as carrera.label for carrera in
carreras" ng-model="carreraSeleccionada"></select>
<span>Año en el que ingresaste a la facu</span>
<input class="form-control" type="text" ng-model="respuesta.anioIngreso" />
<span>¿Cuántos finales aprobaste?</span>
<input class="form-control" type="text" ng-model="respuesta.finalesAprobados" />
<span>¿Cuántos finales desaprobaste?</span>
<input class="form-control" type="text" ng-model="respuesta.finalesDesaprobados" />
<span>¿Cuántos materias aprobaste la cursada?</span>
<input class="form-control" type="text" ng-model="respuesta.cursadasAprobadas" />
<p>¿A que materias pensas Anotarte el proximo cuatrimestre?</p>
<div class="col-md-8">
  <span>Materia</span>
  <select class="form-control" ng-options="materia as materia.label for materia in
carreraSeleccionada.materias" ng-model="materiaSeleccionada"></select>
  <span>Turno</span>
  <select class="form-control" ng-options="turno as turno.label for turno in turnos"
ng-model="turnoSeleccionado"></select>
  <button type="submit" class="btn btn-primary btn-block"
ng-click="agregarMateria()">Agregar</button>
</div>
<div class="col-md-4">
  <div ng-repeat="porCursar in respuesta.materias">
    <h4>{{porCursar.materia.label}} - <small>{{porCursar.turno.label}}</small></h4>
  </div>
</div>
</div>
<button type="submit" class="btn btn-primary btn-block" ng-click="contestar()">Enviar
Respuestas</button>
</div>

```

JS - comportamiento de la vista

```

/* app.js */
var app = angular.module('encuestaApp', ['ngRoute']);

app.config(function($routeProvider) {
  $routeProvider.when('/', { templateUrl : 'pages/login.html', controller: 'LoginCtrl'})
    .when('/responder/:mail', {templateUrl : 'pages/responder.html',controller :
'ResponderCtrl'})
    .when('/gracias', {templateUrl : 'pages/gracias.html'})
    .otherwise({redirectTo: '/'});
});

app.service('EncuestaService', function($http) {
  return {
    responderEncuesta: function(respuesta,callback) {
      $http.get('/responder', respuesta).success(callback);
    },
    getCarreras: function(){
      //obtiene la lista de carreras con el detalle de las materias y lo devuelve
    },
    getTurnos: function(){
      //obtiene la lista de turnos y lo devuelve
    }
  };
});

```

```

    }
  }
});
app.controller('LoginCtrl', function ($location) {
  $scope.autenticar = function(){
    $location.path('responder/' + $scope.user.mail);
  }
});

app.controller('ResponderCtrl', function($scope, $location, $routeParams, EncuestaService){
  $scope.turnos = EncuestaService.getTurnos();
  $scope.carreras = EncuestaService.getCarreras();
  $scope.respuesta = {mail: $routeParams.mail, materias: []};

  $scope.agregarMateria = function(){
    $scope.respuesta.materias.push({materia: $scope.materiaSeleccionada, turno:
$scope.turnoSeleccionado});
    $scope.materiaSeleccionada = {};
    $scope.turnoSeleccionado = {};
  }

  $scope.contestar = function() {
    //Chequeamos los campos obligatorios
    if($scope.respuesta.materias.length <= 0){
      alert('Debe ingresar materias para continuar');
      return;
    }
    if(!$scope.carreraSeleccionada){
      alert('Debe seleccionar una carrera');
      return;
    }
    if(!$scope.respuesta.anioIngreso){
      alert('Debe indicar el año de ingreso a la facultad');
      return;
    }
    //Todo OK, impactamos en el server
    $scope.respuesta.carreraId = $scope.carreraSeleccionada.id;
    EncuestaService.responderEncuesta($scope.respuesta, function(data){
      $location.path('gracias');
    });
  }
});

```

Xtend - controller

```

/* ResponderController.xtend */
class ResponderController {
  extension JSONUtils = new JSONUtils
  @Get('/carreras')
  def Result carreras(){
    response.contentType = ContentType.APPLICATION_JSON
    ok(RepoCarrera.instance.allCarreras.toJson)
  }
  @Get('/turnos')
  def Result turnos(){

```

```

        response.contentType = ContentType.APPLICATION_JSON
        ok(RepoCarrera.instance.allTurnos.toJson)
    }
    @Post('/responder')
    def Result responder(@Body String body) {
        var Respuesta respuesta = body.fromJson(Respuesta)

        val Carrera carrera = RepoCarrera.instance.findCarrera(respuesta.carreraId)
        if(! respuesta.materias.forall[materia | carrera.tieneEnPlanDeEstudio(materia)] ){
            throw new ErrorEnLaRespuesta("No puede mezclar materias de distintas carreras")
        }
        var Encuesta encuesta = respuesta.generarEncuesta();
        RepoEncuesta.instance.agregarRespuesta(respuesta.mail, encuesta)
        ok();
    }

    def static void main(String[] args) {
        XTRest.start(ApuestasController, 9200)
    }
}

```

Recuperatorio de TP3: Mobile usando Android

La facultad nos pide realizar el sistema de inscripciones en celulares.

Para eso, el diseñador nos envió el .xml de las pantallas que desea.

Se pide:

1. Crear el proyecto Android que tenga las pantallas
2. Conectar la aplicación con el servicio Rest creado en Angular (si no tuvo que recuperar el TP de Angular, es necesario realizar el servicio Rest)