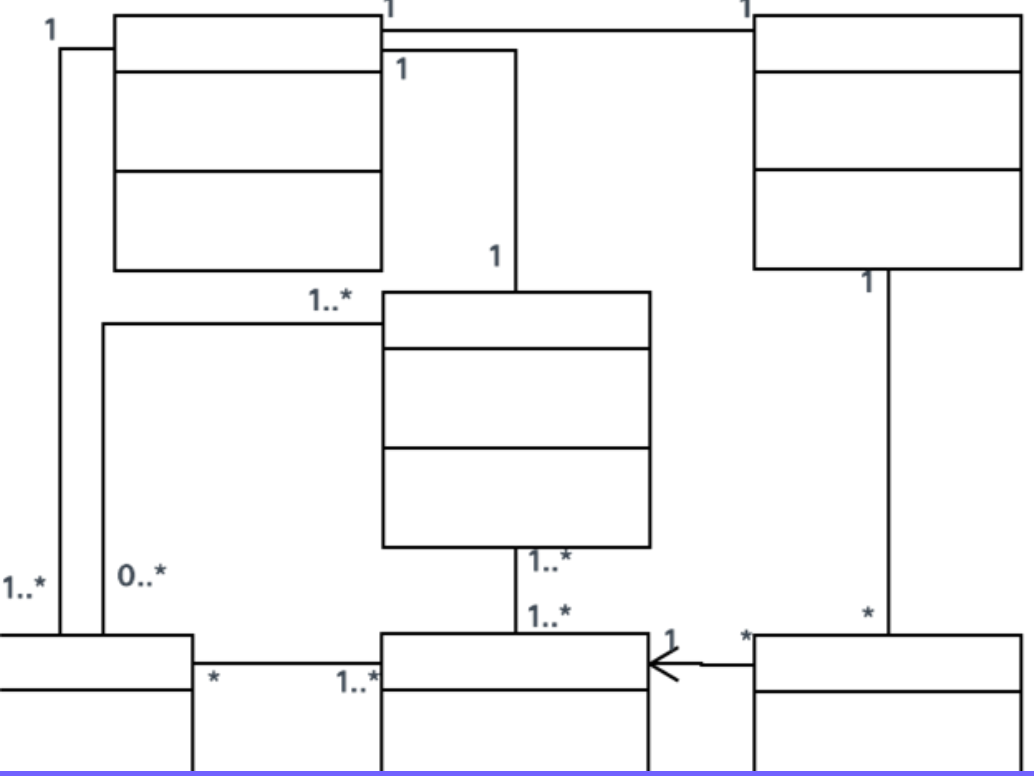

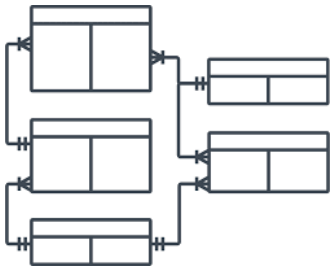

Diseño de Sistemas

GTV – Gestor de Trámites Virtuales

Lectura…

Link al enunciado

Requerimiento 1

Permitir la gestión de trámites por parte de las Entidades

Abstracciones necesarias:

- Clase "***Entidad***" para permitir la gestión de Entidades *("(…) Las entidades que deseen gestionar sus trámites a través de la plataforma serán dadas de alta en el Sistema por un administrador (…)")*

- Clase "***Trámite"*** *("(…) Las entidades podrán especificar los trámites que desean ofrecer a través de la plataforma(...)")*

Requerimiento 2

Permitir la gestión de formularios, por cada trámite, por parte de las entidades

Abstracciones necesarias:

- Clase “***Formulario***”: Los formularios deberían tener una colección de Formularios. Cada formulario debe respetar el concepto de "plantilla", es decir que no debe quedar atado a las respuestas que brinda una persona solicitante.

- Cada formulario debe contener ***Campos/Preguntas***. Por cada campo/pregunta se debe:
  - Especificar si es obligatoria o no -> Sirve para verificar "errores“.
  - Tipo de pregunta (single choice, multiple choice, texto libre, etc.)
  - Posibles respuestas/Opciones.

- **[IMPORTANTE]** No se debe dejar atadas las respuestas que brindan las personas a estos formularios, ya que se estaría rompiendo el concepto de plantilla.

Requerimiento 3

Marcar como “vencidos” aquellos formularios que se encuentren en su fecha de vencimiento y realizar el envío de email correspondiente

Consideraciones necesarias:

- La fecha de vencimiento debe ser calculada y aplicada sobre los formularios (plantillas).
- Este requerimiento sirve para el punto 4). Para poder “*vencer los formularios automáticamente*” es necesario configurar un ***Cron Task*** que verifique, cada cierto tiempo, si el formulario ya entró en su periodo de vencimiento.
- Además, es necesario modelar, en algún atributo, el concepto de "fueRenovado" y la cant. de veces que lo fue

Requerimiento 4

Permitir el alta de solicitudes de trámites por parte de las personas solicitantes.

Abstracciones necesarias:

- Clase “***SolicitudTramite***”(similar al concepto de "*Cuestionario Contestado*“) Esta solicitud debe tener como atributos, mínimamente:
  - Persona solicitante.
  - Trámite.
  - Formularios contestados
- Clase “***FormularioContestado***”: considerando que un trámite podría llegar a tener uno o varios formularios asociados. Cada Formulario Contestado debe tener como atributos, mínimamente:
  - El formulario al que hace referencia
  - Una colección de "***Respuestas***". Es importante que la respuesta considere todos los tipos de preguntas, es decir que debe contener una colección de opciones seleccionadas, un atributo "valorContestado" (o similar, para el caso de las preguntas libres), etc.

Permitir el alta de solicitudes de trámites por parte de las personas solicitantes

```
>> SolicitudesDeTramitesController

          crearSolicitudDeTramite(datos) {
                    SolicitudDeTramite nuevaSolicitud = new SolicitudDeTramite();
                    Tramite tramiteBuscado = this.repoTramite(datos.tramiteId);

                    nuevaSolicitud.setTramite(tramiteBuscado);

                    for(datos.formularios as formulario) {
                              FormularioContestado formContestado = new FormularioContestado();
                              Formulario formularioAContestar = tramiteBuscado.formularioDeId(formulario.id);

                              formContestado.setFormulario(formularioAContestar);

                              for(formulario.respuestas as respuesta) {
                                        Respuesta unaRespuesta = new Respuesta();

                                        Campo campoAContestar = formularioAContestar.campoDeId(respuesta.campoId);

                                        unaRespuesta.setCampo(campoAContestar);
                                        campoAContestar.generarRespuesta(unaRespuesta, respuesta); //DELEGAMOS AL TIPO, POR SI
HAY OPCIONES Y RTA LIBRE

                                        formContestado.agregarRespuesta(unaRespuesta);
                              }

                              nuevaSolicitud.agregarFormularioContestado(formContestado);
                    }

                    this.repoSolicitudes.agregar(nuevaSolicitud);
          }
```

Requerimiento 5

Permitir el seguimiento (cambio de estados) de las solicitudes de trámites por parte de las entidades.

Abstracciones necesarias:

- "***Posible estado***": para poder representar a los posibles estados "genéricos" por los cuales pueden pasar todas las solicitudes de trámites. Puede ser un enumerado o una clase concreta (varias instancias). No se debería utilizar un State pues no existe un comportamiento específico en la solicitud que varíe de acuerdo al estado, ni tampoco existe transiciones definidas.

- "***Posible estado del trámite***": para representar a los posibles estados por los cuales puede pasar una solicitud para UN trámite de UNA entidad en particular. Esta abstracción es necesaria por el requerimiento de las configuraciones de acciones.

- "***Estado***": para representar al estado concreto de la solicitud. El mismo debe tener, mínimamente: un posible estado del trámite; una fecha/hora.

Permitir el seguimiento (cambio de estados) de las solicitudes de trámites por parte de las entidades.

```
>> CambioDeEstadoSolicitudController

        cambiarEstadoSolicitud(datos) {
                SolicitudDeTramite solicitudBuscada = this.repoSolicitudes.find(datos.solicitudTramiteId);
                PosibleEstadoTramite posibleEstado = this.repoPosiblesEstados.find(datos.posibleEstadoId);

                EstadoTramite nuevoEstadoTramite = new EstadoTramite();
                nuevoEstadoTramite.setPosibleEstado(posibleEstado);
                nuevoEstadoTramite.setFechaHora(LocalDateTime.now);

                solicitudBuscada.cambiarEstado(nuevoEstadoTramite);

                nuevoEstadoTramite.ejecutarAcciones();
        }
```

Requerimiento 6

Permitir que las entidades configuren las acciones a ejecutar en cada estado.

Consideraciones:

- Para lograr este requerimiento se debe modelar el "***Posible estado del trámite***", mencionado en el requerimiento anterior. Cada posible estado del trámite debe tener una colección de "***acciones"***.

Abstracciones necesarias y otras consideraciones:

- "***Acción***": podría ser una interface con un método "ejecutar" o similar, inspirado en el patrón Command.

- Cada acción requiere de distintas configuraciones para poder funcionar:
  - *Acciones stateless*: deberíamos enviarles, sí o sí, por parámetro en el método "ejecutar", la configuración deseada.
  - *Acciones Stateful*: las configuraciones están guardadas dentro de los objetos que representan a las acciones, lo que los convierte en objetos NO reutilizables.

Permitir que las entidades configuren las acciones a ejecutar en cada estado.

```
>> PosibleEstadoTramiteController

        configurarAccion(datos) {
                PosibleEstadoTramite posibleEstado = this.repoPosiblesEstados.find(datos.posibleEstadoId);

                Accion unaAccion = FactoryAcciones.crearAccion(datos.accion, datos.configuracion);

                posibleEstado.agregarAccion(unaAccion);


                this.repoPosiblesEstados.merge(posibleEstado);
        }
```

Gracias