Universidad de Costa Rica

Facultad de Ingeniería
Escuela de Ciencias de la Computación e Informática

CI0128 – Proyecto Integrador: Ingeniería de Software y Bases de Datos
Profesoras: Rebeca Obando y Alexandra Martínez

Reporte: Sprint 1

Nombre del Equipo: Ta' Bueno

Miembros:
Emmanuel D. Solís - B97670 (Scrum Master)
emmanuel.solispomares@ucr.ac.cr
Gabriel Zúñiga - B98755
gabriel.zunigaorozco@ucr.ac.cr
Jan Murillo - B95447
jan.murillo@ucr.ac.cr
Kevin Arguedas - B80626
kevin.arguedasmuriel@ucr.ac.cr
Luis D. Chinchilla - B82227
luis.chinchillaotarola@ucr.ac.cr

Índice

UNIVERSIDAD DE COSTA RICA

ECCI Escuela de Ciencias de la Computación e Informática

Curso: CI-0128 I ciclo 2022
Grupo: 01

Proyecto integrador bases de datos e ingeniería de software

Especificación del *sprint* 1

El objetivo de este *sprint* es trabajar en la creación y consulta de las principales entidades del sistema. El equipo debe especificar el durante la ceremonia de planeamiento el *sprint backlog* además de estimar y definir quién es el desarrollador del cada *user story (US)* o tarea. El manejo del *backlog* y el *sprint backlog* se va a hacer a través de la herramienta JIRA. En esta herramienta cada equipo va a manejar los detalles de las *user stories*, incluyendo las descripciones, los criterios de aceptación, la cantidad de puntos, así como cualquier otro detalle necesario para la claridad del desarrollo.

En este primer *sprint* se espera que los estudiantes cumplan con las siguientes ceremonias de Scrum: planeamiento (*planning*), sesiones de revisión diaria (*stand-ups*), sesiones de refinamiento de requerimientos (*refinement o grooming)*, revisión y demo (*review*) con los *stakeholders,* además de la sesión para hacer retrospectiva sobre el proceso (*retrospective*). Las fechas en las que se realizarán estas ceremonias se especifican a continuación:

Ceremonia	Fecha
Sprint planning	5 de mayo 2022 (en clase)
Sprint stand-ups	fechas y horario acordado por el equipo
Sprint grooming para el *sprint* 2	19 de mayo 2022 (en clase)
Sprint demo	26 de mayo 2022 (en clase)
Sprint retrospective	26 de mayo 2022 (fuera del horario de clase)

A. Entregable y requisitos de cada ceremonia

- Ceremonias de revisión diaria (*daily stand-ups*): deben hacerse a la hora convenida por el equipo y usando el medio de comunicación especificado en los entregables del *sprint* 0. Las docentes o el asistente pueden conectarse sin previo aviso a cualquiera de estas reuniones. Si el equipo no está conectado en el horario acordado, se realizará **una penalización de 5 pts** sobre el valor total del proyecto. Similarmente, si alguno de los miembros del equipo no está presente, tendrá esta penalización en su nota individual del proyecto. Como producto de esta ceremonia, el equipo debe entregar lo solicitado en el documento llamado "Artefacto ceremonias scrum", que se encuentra en mediación virtual.

- Ceremonia de planeamiento (*sprint planning*): todas las historias agregadas al *sprint backlog* deben cumplir con el criterio de ***user story ready*** para ser parte del *sprint*. Como producto de esta ceremonia, el equipo debe entregar lo solicitado en el documento llamado "Artefacto ceremonias scrum", que se encuentra en mediación virtual.
- Ceremonia de refinamiento (*sprint grooming*): el equipo debe entregar, como producto de esta ceremonia, lo solicitado en el documento llamado "Artefacto ceremonias scrum", que se encuentra en mediación virtual.
- Ceremonia de revisión y demo (*sprint demo*): el equipo debe entregar, como producto de esta ceremonia, lo solicitado en el documento llamado "Artefacto ceremonias scrum", que se encuentra en mediación virtual.
- Ceremonia de retrospectiva (*sprint retrospective*): esta ceremonia no contará con la presencia de las docentes, al realizarse fuera del horario de clase. Sin embargo, la ceremonia debe ser dirigida y grabada por el *scrum master*. Los entregables serán el enlace al video grabado y lo solicitado en el documento llamado "Artefacto ceremonias scrum", que se encuentra en mediación virtual.

B. Artefactos a entregar

En este *sprint* se deben **entregar** los siguientes **artefactos**:

1. Código fuente del sistema: debe entregarse el enlace al repositorio github (recuerden que la docente de Ingeniería de Software y el asistente deben tener acceso al mismo, usuarios: **rebeca-ov** y **ChristianRojasRios**). El historial de cambios del repositorio debe tener múltiples *commits* y *pull requests* de cada integrante del equipo. No se va a aceptar un único *pull request* con todo el código fuente del *sprint*.

   Es recomendable crear un *pull request* por cada tarea de un *user story*. Si el cambio es muy pequeño, pueden unir tareas en un solo *pull request*. Sin embargo, no se recomienda hacer *pull request* con muchas líneas de código, debido a que la revisión de código va a ser difícil de manejar.

   La nota del *sprint* dependerá de la contribución que cada miembro del equipo realice al repositorio del proyecto. Para ello, se tomará en cuenta la cantidad y calidad de los *commits* y *pull requests* hechos por cada estudiante.
2. *Sprint backlog*: cada *user story* del *sprint backlog* debe tener: las subtareas requeridas para completarla, los criterios de aceptación, la estimación de la historia, el tiempo estimado en cada subtarea, y el desarrollador encargado de la historia. Toda esta información debe manejarse a través de la herramienta JIRA. Recuerden que para que un *user story* se declare como ***done*** debe cumplir con todos los criterios establecidos por el equipo.
3. Entregable de las ceremonias de Scrum: debe contener todo lo indicado en la sección anterior de este documento ("Entregable y requisitos de cada ceremonia").

4. *Mock up* de las páginas trabajadas en el *sprint*: cada *user story* que requiera una nueva interfaz debe contar con su respectivo *mock up*, el cual debe ser validado por el *product owner* a los *stakeholders*.
5. Diagrama UML: diagrama de clases del diseño planteado para la aplicación.
6. Diseño lógico de la base de datos e implementación: debe entregarse el diseño lógico de la base de datos completa, correspondiente al diagrama ER entregado en el *sprint* 0, o su versión actualizada (en caso de que lo hayan modificado en este *sprint*). Si el equipo ha realizado cambios al diseño conceptual (ER), entonces debe entregar también la versión actualizada del mismo. Recuerden que el diseño conceptual debe responder a los *user stories* del *backlog*. Además, deben entregar el *script* SQL de creación de tablas y sus restricciones, correspondiente a la parte de la base de datos implementada para este *sprint*.
7. Presentación durante el *sprint review*: durante el *sprint* se les proveerá una guía de lo que se espera que presenten, y se les evaluará con base en dicha guía.
8. Autoevaluación y coevaluación: durante el *sprint* se les proveerá el instrumento.

Ponderación de los artefactos:

Artefacto	Porcentaje
1. Código fuente con revisión del buen manejo de GIT: donde se vean múltiples pull request por desarrollador.	30%
2. *Sprint backlog* en JIRA	5%
3. Entregable de las ceremonias Scrum	15%
4. *Mock ups* de la aplicación	10%
5. Diagrama de clases	10%
6. Diseño lógico de la base de datos e implementación	20%
7. Presentación durante el *sprint review*	5%
8. Autoevaluación y coevaluación	5%

1. Código fuente del Git

El enlace del repositorio es el siguiente: `https://github.com/emasp2001/pi_ingeBases/tree/main/Planilla`; dado que este es un repositorio privado la invitación a la profesora había sido enviada, sin embargo esta no fue aceptada dentro del tiempo rango establecido por Github por lo que automaticamente expiró, por lo que se le vuelve a enviar el día 26 de mayo a la 1:00am. El asistente (usuario: *ChristianRojasRios*) ya había aceptado la invitación por lo que él sí se encuentra dentro del repositorio.
Cabe mencionar que se siguió la regla de completar las user stories por medio de diferentes *branch* y *pull requests*.

2. Sprint backlog de Jira

El enlace donde se encuentra el *sprint backlog* es el siguiente: `https://ingesoftg001.atlassian.net/jira/software/projects/TB/boards/5/backlog`.

3. Entregable de las ceremonias Scrum

Se encuentra adjunto en el archivo **bitacora_ceremonias.pdf**.

4. Mock ups de la aplicación

Los Mock Ups de esta aplicación han sido aprobados por el *Project Owner* y se encuentran disponibles en esta carpeta de este entregable bajo el nombre de ***mock_ups.pdf***.

5. Diagrama de clases - UML

El diagrama de clases UML de este proyecto se encuentra adjunto en esta carpeta de entrega bajo el nombre ***diagrama_uml.pdf***.

6. Diseño lógio de la base de datos e implementación

6.1. Modelo Lógico Relacional

Estos se encuentran adjuntos en esta carpeta de entrega bajo el nombre de ***modelo_logico_database.pdf*** para el modelo lógico relacional de tablas, y bajo el nombre ***modelo_entidad_relacion.pdf*** para el modelo de entidad relación.

6.2. Script SQL

Este script se encuentra también adjunto en esta carpeta bajo el nombre ***TaBueno_Query.sql***, sin embargo a modo de **respaldo** se adjunta también ese mismo código en este archivo:

```
------------------------ Oficial Ta' Bueno SQL Query -------------------------
use TaBueno

--------------- Creating Tables --------------
-- Entidad: Usuario
create table Usuario
    (Cedula         char(10)        NOT NULL,
    Contrasena      varchar(50)     NOT NULL,
```

```
    Nombre          varchar(15)     NOT NULL,
    Apellido1       varchar(15)     NOT NULL,
    Apellido2       varchar(15)     NOT NULL,
    Telefono        int             NOT NULL,
    TipoUsuario     tinyint         NOT NULL,
    Provincia       varchar(15)     NOT NULL,
    Canton          varchar(15)     NOT NULL,
    CodigoPostal    char(5)         NOT NULL,
    primary key (Cedula)
);

-- Entidad: Proyecto
create table Proyecto
    (nombre             varchar(200)        not null,
    cedulaEmpleador     char(10)            not null,
    presupuesto         float,
    modalidadPago       varchar(50),
    primary key (nombre, cedulaEmpleador),
    foreign key (cedulaEmpleador) references Usuario
);

-- Entidad: Deducciones Obligatorias
create table DeduccionesObligatorias
    (nombre             varchar(100)        not null,
    porcentaje          float
    primary key (nombre)
);

-- Relacion: TrabajaEn
create table TrabajaEn
    (nombreProyecto         varchar(200)        not null,
    cedulaEmpleador         char(10)            not null,
    cedulaEmpleado          char(10)            not null,
    registroHoras           INT,
    primary key (nombreProyecto, cedulaEmpleador, cedulaEmpleado),
    foreign key (nombreProyecto, cedulaEmpleador) references Proyecto
);

-- Entidad: Beneficios
create table Beneficios
    (nombreBeneficio        varchar(200)        not null,
    cedulaEmpleador         char(10)            not null,
    nombreProyecto          varchar(200)        not null,
    primary key (nombreBeneficio, cedulaEmpleador, nombreProyecto),
    foreign key (nombreProyecto, cedulaEmpleador) references Proyecto
);

-- Entidad: Pago
create table Pago
    (cedulaRecibe           char(10)            not null,
    fecha                   date                not null,
```

```
    nombreProyecto          varchar(200)        not null,
    cedulaEmpleador         char(10)            not null,
    salarioBruto            float,
    tipoPago                varchar(50),
    primary key (cedulaRecibe, fecha),
    foreign key (nombreProyecto, cedulaEmpleador) references Proyecto
);

-- Relacion: IncluyeDeduccionObligatoria
create table IncluyeDeduccionObligatoria
    (cedulaEmpleado                 char(10)            not null,
    fechaPago                       date                not null,
    nombreDeduccionObligatoria      varchar(100)        not null,
    primary key (cedulaEmpleado, fechaPago, nombreDeduccionObligatoria),
    foreign key (cedulaEmpleado, fechaPago) references Pago,
    foreign key (nombreDeduccionObligatoria) references DeduccionesObligatorias
);

-- Entidad: Contrato
create table Contrato
    (nombreProyecto         varchar(200)        not null,
    cedulaEmpleador         char(10)            not null,
    cedulaEmpleado          char(10)            not null,
    fechaInicio             date                not null,
    puesto                  varchar(100)        not null,
    fechaFinalizacion       date,
    jornadaLaboral          varchar(200),
    primary key (nombreProyecto, cedulaEmpleador, cedulaEmpleado, fechaInicio),
    foreign key (nombreProyecto, cedulaEmpleador) references Proyecto,
    foreign key (cedulaEmpleado) references Usuario
);

-- Entidad: EstadoBeneficio
create table EstadoBeneficio
    (nombreBeneficio        varchar(200)        not null,
    cedulaEmpleador         char(10)            not null,
    nombreProyecto          varchar(200)        not null,
    fechaInicio             date,
    fechaFinalizacion       date,
    primary key (nombreBeneficio, cedulaEmpleador, nombreProyecto, fechaInicio),
    foreign key (nombreBeneficio, cedulaEmpleador, nombreProyecto) references Beneficios
);

-- Relacion: ApareceEn
create table ApareceEn
    (nombreBeneficio        varchar(200)        not null,
    cedulaEmpleador         char(10)            not null,
    nombreProyecto          varchar(200)        not null,
    fechaInicio             date,
    cedulaRecibePago        char(10),
    fechaPago               date,
```

```
    primary key (nombreBeneficio, cedulaEmpleador, nombreProyecto, fechaInicio,
                 cedulaRecibePago, fechaPago),
    foreign key (nombreBeneficio, cedulaEmpleador, nombreProyecto, fechaInicio)
    references EstadoBeneficio
);

-- Entidad: DeduccionesVoluntarias
create table DeduccionesVoluntarias
    (nombre                 varchar(100)        not null,
    primary key (nombre)
);

-- Entidad: EstadoDeduccionVoluntaria
create table EstadoDeduccionVoluntaria
    (nombreDeduccion        varchar(100)        not null,
    fechaInicial            date                not null,
    fechaFinalizacion       date,
    monto                   float               not null,
    nombreProyecto          varchar(200)        not null,
    cedulaEmpleador         char(10)            not null,
    primary key (nombreDeduccion, fechaInicial),
    foreign key (nombreProyecto, cedulaEmpleador) references Proyecto
);

-- Relacion: IncluyeDeduccionVoluntaria
create table IncluyeDeduccionVoluntaria
    (nombreDeduccion        varchar(100)        not null,
    fechaInicialDeduccion   date                not null,
    fechaPago               date                not null,
    cedulaRecibePago        char(10)            not null,
    primary key (nombreDeduccion, fechaInicialDeduccion, fechaPago, cedulaRecibePago),
    foreign key (nombreDeduccion, fechaInicialDeduccion) references EstadoDeduccionVoluntaria,
    foreign key (cedulaRecibePago, fechaPago) references Pago
);


--------------- Inserting Values --------------
-- Usuarios
insert into Usuario values ('1234567890', 'password', 'Diego', 'Chinchilla', 'Otarola',
                            '76343352', 0, 'San Jose', 'Hatillo', '10101');
insert into Usuario values ('0987654321', 'password', 'Jan', 'Murillo', 'Barquero',
                            '23213433', 1, 'Alajuela', 'Alajuela', '20201');
insert into Usuario values ('2365123132', 'password', 'Gabriel', 'Zuniga', 'Orozco',
                            '35435344', 1, 'Heredia', 'Heredia', '30301');

-- Proyectos
insert into Proyecto values ('Sistema de Planilla CR', '1234567890', '0987654321', 0)

-- Deducciones Obligatorias
insert into DeduccionesObligatorias values ('Caja Costarricense de Seguro Social', 0.10);
insert into DeduccionesObligatorias values ('Impuesto al Valor Agregado', 0.05);
```

```
Insert into DeduccionesObligatorias Values ('Enfermedad y Maternidad', 0.055)
Insert into DeduccionesObligatorias Values ('Invalidez, Vejez y Muerte', 0.0384)
Insert into DeduccionesObligatorias Values ('Aporte Trabajador', 0.1)
Insert into DeduccionesObligatorias Values ('Bajo 863000', 0)
Insert into DeduccionesObligatorias Values ('Hasta 1267000', 0.1)
Insert into DeduccionesObligatorias Values ('Hasta 2223000', 0.15)
Insert into DeduccionesObligatorias Values ('Hasta 4445000', 0.20)
Insert into DeduccionesObligatorias Values ('Sobre 4445000', 0.25)

-- Beneficios
insert into Beneficios values ('Plan Dental', '1234567890', 'Sistema de Planilla CR')

--------------- Consulting Values --------------
select * from Usuario
select * from Proyecto
select * from Beneficios
select * from DeduccionesObligatorias
```

7. Presentación durante el *sprint review*

Se realizó la presentación de este código el día 26 de Mayo del 2022.

8. Autoevaluación y coevaluación

Cada miembro del equipo debió ser responsable de adjuntar su propia autoevaluación y coevaluación.