UNIVERSIDAD NACIONAL DE LA PLATA
FACULTAD DE INFORMÁTICA

TALLER DE TECNOLOGÍAS DE PRODUCCIÓN DE SOFTWARE (Opción A)

Carrera: ***Analista Programador Universitario***

Año 2018

Año: **3°**
Carácter: Obligatoria
Regimen de Cursada: Semestral
Correlativas:
Profesores: **Laura A. Fava**
Jorge H. Rosso
Hs. semanales: **6 hs.**

FUNDAMENTACIÓN

Esta asignatura se dicta en el segundo semestre del último año de la carrera Analista Programador Universitario, recibe estudiantes que ya han adquirido como mínimo, conocimientos sobre algoritmos de programas, diferentes paradigmas de programación, entre ellos el paradigma orientado a objetos y han experimentado con diferentes mecanismos para persistir información; esto es, se han apropiado de conocimientos básicos que nos permiten abordar temas más avanzados e integradores.
Asumiendo esta adquisición de conocimientos previos y advirtiendo la proximidad de su graduación, en este espacio nos proponemos introducir a los alumnos en un esquema de producción de software realista de acuerdo a los estándares del arte, usando JAVA y software libre. Para ello además de enseñarles nuevas tecnologías JAVA para el desarrollo de aplicaciones basadas en web, en la última etapa de la cursada, se les propone resolver una problemática social relacionada con desarrollo de software. Esta actividad, les permite a los estudiantes no sólo integrar y aplicar todos los conocimientos adquiridos, sino también, ser parte de un grupo de desarrollo de software y contar con un escenario real con necesidades específicas donde desempeñarse.

OBJETIVOS GENERALES:

Introducir a los alumnos en un esquema de organización de producción de software, utilizando metodologías, prácticas y herramientas actualizadas y acordes con los estándares actuales.
Fomentar la práctica del alumno en esquemas de trabajo similares a los que se utilizan en las empresas de desarrollo de productos de software.
Ofrecer a los alumnos alternativas tecnológicas, siempre en base a herramientas de utilización actual en el mercado laboral.

CONTENIDOS MINIMOS:

- Introducir un ambiente de desarrollo de software estandarizado (con herramientas integradas que den una visión homogénea y estandarizada de las aplicaciones, su interfaz grafica, el acceso a las bases de datos y la interconexión entre aplicaciones), enfocado a un organismo o "clase" de empresa usuaria.
- Practicar como usar el ambiente de desarrollo y las diferencias que tiene que con el ambiente de producción, ilustrando la metodología organizacional del pasaje de desarrollo a producción.

- Practicar con documentación estandarizada mostrando como se pasa de una especificación al código ejecutable.
- Ejemplificar la actividad del tester de aplicaciones. Metodología de trabajo y ambiente de prueba (diferencia con los otros ambientes).
- Plantear el proceso estandarizado de desarrollo de software en una tecnología de uso en el mercado. Rol de la documentación en cada etapa.
- Plantear el desarrollo de una solución a un problema real y que ilustre todas las problemáticas antes descriptas.
- Describir cuales son las principales características de un proceso de desarrollo de software con calidad (introduciendo los principios básicos de CMM o CMMI)

PROGRAMA ANALÍTICO

UNIDAD DIDÁCTICA I: Aplicaciones J2EE

Motivación: A lo largo del curso se trabajará en la construcción de aplicaciones basadas en web, aplicaciones que involucran el uso de la red, del protocolo HTTP y que ameritan un análisis del nuevo –hasta ahora los alumnos solo han trabajado con aplicaciones locales- escenario donde ejecutan. ¿Cuáles son las tecnologías, los mecanismos, los aspectos que se deben tener en cuenta al momento de comenzar a desarrollar aplicaciones basadas en web?

Temas: Arquitectura de las aplicaciones web y componentes del estándar JEE (*Java Enterprise Edition*), en especial **Servlets** y **JavaServer Pages (JSP)**. Introducción a las tecnologías del lado del servidor que posibilitan la construcción de aplicaciones web dinámicas y a las tecnologías del lado del cliente: Java Script y AJAX para implementar interfaces de usuarios gráficas y más eficientes. IDEs (Integration Development Environments) y *Frameworks* para JEE.

Objetivos Mínimos: Comprender el protocolo HTTP y el lenguaje HTML, las arquitecturas de aplicaciones basadas en web, más precisamente aplicaciones JEE, así como también los entorno de desarrollo y los *frameworks* utilizados para su implementación.

UNIDAD DIDÁCTICA II: Componentes JEE

Motivación: Aquí comienza el aprendizaje de la plataforma empresarial de JAVA o JEE. Se analizará en profundidad el estándar JEE 5 -*Java Enterprise Edition* 5-. Nos preguntaremos: ¿cuáles son las componentes del estándar JEE?, ¿para qué sirve cada una de ellas?, ¿por qué los Servlets son una parte vital del estándar Java EE?, ¿cómo integrar las componentes para desarrollar aplicaciones con arquitecturas modulares, extensibles y reusables?

Temas:

Servlets. Características generales. Servicios que proveen los Contenedores Web. Ciclo de vida de los Servlets: init(), service() y destroy(). El Archivo descriptor de la aplicación: web.xml, *deploy* de

Servlets. Parámetros de inicialización de servlets y de contexto. Redireccionamiento del requerimiento: sendRedirect() y delegación del requerimiento y de la respuesta: forward() e include().

Filtros. Características generales. Ciclo de vida de los Filtros: init(), doFilter() y destroy(). Configuración. El objeto FilterChain. Las clases wrappers. Programando requerimientos y respuestas customizados.

JavaServer Pages (JSP). ¿qué son las páginas JSP? Ciclo de vida de una JSP: fase de traducción, fase de compilación y fase de ejecución. Elementos para construcción de JavaServer Pages: Elementos de scripting, directivas, acciones estándares, el lenguaje de expresiones (JSP EL) y Java Standard Tag Library (JSTL).

Sesiones. Métodos tradicionales y el objeto HTTPSession. ¿Cómo mantener el estado con el protocolo HTTP? Mecanismos de intercambio de ID para manejar sesiones: cookies y URL *rewriting.* Lectura y escritura de datos de las sesiones. Sesiones en ambientes multi-servidores.

Contextos**:** aplicación, sesión, requerimiento.

Objetivos Mínimos: Conocer las diferentes componentes estándares provistas por la plataforma empresarial JAVA. Saber implementarlas y usarlas en la capa de software correspondiente.

UNIDAD DIDÁCTICA III: Persitencia de datos

Motivación: Por muchos años, la persistencia ha sido un tema de debate en la comunidad de Java. ¿Es la persistencia un problema que ha sido resuelto por las Bases de Datos y extensiones tales como *Store Procedures* o es un problema más general que debe ser resuelto por un modelo de componentes java, como *EntityBeans* de EJB (*Enterprise Java Beans*)? Podríamos manejar código de la manera más primitiva CRUD (create, read, update, delete) con SQL y JDBC? Será una nueva propuesta llamada ORM (*Object-Relational Mapping*) la solución definitiva? El debate continúa.

Temas: ¿Qué es la persistencia de datos? Mecanismos para persistir: Serialización, Conexión a Base de Datos usando JDBC (Java DataBase Connectivity) y ORM. Uso del patrón DAO (Data Access Object) para encapsular el acceso a datos.
Serialización. Su uso, ventajas y desventajas.
Java DataBase Connectivity. Tipos de Drivers JDBC. La API (Application Programming Interface) JDBC. Establecimiento de una Conexión. Ejecución de Sentencias SQL. Las clases Statement, PreparedStatement y CallableStatement. Pool de Conexiones. La interface DataSource.
Object-Relational Mapping. JPA Java Peristence API. Comprendiendo los Estándares. Hibernate y EJB 3.0.

Objetivos Mínimos: Reconocer los diferentes mecanismos para persistir datos y poder seleccionar el adecuado para cada situación.

UNIDAD DIDÁCTICA IV: Spring

Motivación: Si bien es posible escribir aplicaciones web codificando directamente con las APIs de JAVA estándares, es una buena práctica usar *frameworks* para hacerlo. El uso de *frameworks* no sólo acelera el proceso de desarrollo ofreciendo una capa de abstracción a las APIs, sino también, ayudan a desarrollar aplicaciones modulares, extensibles y reusables, basadas en su gran mayoría en algún patrón de diseño aprobado por la comunidad de desarrolladores. En esta etapa final del curso se enseña **Spring Core y Spring MVC,** *frameworks open source* de amplio uso en todos los ámbitos de desarrollo, maduros y con una comunidad de usuarios muy activa. Demostraremos que estos *frameworks* promueven el uso de buenas prácticas de desarrollo y analizaremos su integración con AJAX y con los frameworks para persistencia vistos en la unidad anterior.

Temas:
Evolución del framework Spring Core. Arquitectura del Framework. Inversión de Control (IoC) e Inyección de Dependencias (DI). Configuración de la aplicación por XML y por anotaciones.
Introducción a Spring MVC. Procesamiento de un requerimiento. Configuración y contextos de la aplicación. Internacionalización. Validación.
Implementación de web services mediante REST(Representational State Transfer). Principios de RESTful: identificación de recursos mediante URIs, uso de una interfaz uniforme y acotada. Implementando REST con Spring MVC.

Objetivos Mínimos: Comprender la arquitectura de los *frameworks* Spring Core y Spring MVC, y utilizarlos para el desarrollo de aplicaciones web, vinculando los conceptos propuestos por los *frameworks* con conocimientos adquiridos a lo largo del taller, ya sea para la definición de la vista de la aplicación como para las otras capas intermedias hasta la comunicación con los datos del sistema.

UNIDAD DIDÁCTICA V: Angular

Motivación: En los últimos años la arquitectura está teniendo un cambio desde aplicaciones multi-page a aplicaciones singel-page (SPA). AngularJS es un framework *open source*, que usa una arquitectura MVC (Model-View- Controller) para crear aplicaciones SPA. Permite utilizar los mismos documentos HTML como templates para las vistas de la aplicación y además maneja el *data binding* del modelo de datos con la vista, dando dinamismo.

Temas:
Características de AngularJS. Arquitectura de AngujarJS. Componentes principales: Módulos, Controladores. Inyección de dependencias y Scopes. Data Binding. Servicios, Filtros, Directivas. Autenticación mediante tokens.

Objetivos Mínimos: Comprender las diferentes arquitecturas de aplicaciones web y las caarcteríticas principales de Angular. Implementar una aplicacion Angular que consulte servicios REST provistos por Spring.

METODOLOGÍA DE ENSEÑANZA

La materia consta de un encuentro presencial semanal con una duración de 6 horas donde se integra el desarrollo conceptual que se utilizará en la producción práctica. Se organiza en una instancia inicial de explicación teórica donde se articulan los contenidos previos y se introducen y trabajan nuevos temas del programa que luego serán aplicados en la resolución de problemas prácticos.
La ***instancia teórica*** comienza con un repaso del tema previo, donde se indaga a los alumnos sobre las dificultades que han tenido en la aplicación de los conceptos teóricos en la práctica. Después del cierre del tema anterior se plantea el nuevo tema, se lo desarrolla y se intenta motivar a los estudiantes planteándoles los desafíos de la instancia práctica que la sucede.
La ***instancia práctica*** comienza con una breve explicación de la propuesta o trabajo práctico del día, donde se selecciona uno de los ejercicios para que sea entregado en forma individual por cada estudiante. Si bien sólo se pide la entrega de uno de los ejercicios, la disponibilidad de máquinas y la buena relación docente/alumno permite realizar un seguimiento continuo de todos los ejercicios planteados en los trabajos prácticos.
Además de las producciones individuales semanales, se propone también una actividad grupal integradora en la etapa final de la cursada. Este trabajo final es un prototipo avanzado de un Sistema Informático destinado a una Entidad Pública con necesidades en el área de informática que es resuelto por todos los grupos –de no más de tres alumnos- que componen la clase. Esta actividad tiene como objetivo, además de aplicar el conocimiento adquirido, funcionar como estímulo para implicar a los alumnos en sus aprendizajes y para comprometerlos con actividades de extensión universitaria.

Materiales Didácticos

Nuestra facultad nos provee soporte para la creación de cursos a través de la plataforma MOODLE (http://moodle.org/). Esta plataforma nos permite disponer de un lugar centralizado para compartir el material utilizado para todas las actividades, en especial, las clases teóricas con antelación a su exposición y los trabajos prácticos antes del comienzo de los mimos.
Para el desarrollo de las clases teóricas se utiliza un cañón óptico para proyectar las diapositivas digitales previamente subidas a MOODLE. De manera similar, los enunciados de los trabajos prácticos también están disponibles a dicha plataforma antes de la práctica.
Para la instancia práctica se hace uso de una sala de computadoras adecuadamente equipada, que por el número de alumnos de la materia permite que cada uno pueda disponer de una men forma individual.

EVALUACIÓN

Todas las actividades de clase son, potencialmente, actividades de evaluación, a partir de las cuales se puede revelar información útil para cumplir con las funciones básicas de la evaluación: retroalimentar la acción didáctica y acreditar los aprendizajes. Por este motivo, se realizan diferentes evaluaciones a saber a lo largo de la secuencia de enseñanza:

***Evaluación diagnóstica inicial*:** El objetivo de esta evaluación es obtener las características de los estudiantes en el momento de partida, de manera de poder planificar efectivamente el proceso de enseñanza-aprendizaje.
***Evaluación diagnóstica continua o formativa*:** Tiene como propósito facilitar el aprendizaje de los alumnos, no simplemente medir cuanto han aprendido. Para ello, al finalizar algunas unidades didácticas claves se les pide a los estudiantes que entreguen un trabajo individual relacionado con los temas de la unidad y con una complejidad levemente superior a los ejercicios desempeñados en las clases prácticas. Esta evaluación debe ser desarrollada en forma individual y en el laboratorio donde se desarrollan los trabajos prácticos.
Estos trabajos motivan el aprendizaje de los estudiantes y les promueven la auto evaluación, mientras que a los docentes nos permiten diagnosticar y remediar dificultades en el proceso de aprendizaje.
Evaluación sumativa: tiene como objetivo hacer un balance de lo aprendido a lo largo del proceso y es realizada con dos propósitos:

a. **acreditar si el estudiante alcanzó o no los objetivos mínimos**, esto es la aprobación de la cursada. Para aprobar la cursada un estudiante debe haber aprobado al menos un 80% de las evaluaciones continuas o ejercicios especiales que se realizan en el laboratorio donde se desarrolla la práctica y de esta manera accede a una *evaluación reducida* o debe rendir una *evaluación integradora* con temas de todo el semestre.
b. **calificar al estudiante**, esto es la aprobación del final y la definición de una nota. Los alumnos tienen dos alternativas: integrar algunos de los trabajos realizados en las prácticas y completarlo para lograr un módulo que forma parte de un *sistema integral* o rendir un *examen final* tradicional.

BIBLIOGRAFÍA OBLIGATORIA

- Servlets and JavaServer Pages – The J2EE Technology Web Tier, Jayson Falkner, Kevin Jones. *Addison-Wesley*.
- Head First Servlets & JSP, Bryan Basham, Kathy Sierra, Bert Bates. O'Reilly
- Java Persistence with Hibernate, Christian Bauer, Gavin King. *Manning*, 2007.
- Stuts2 in Action, Donald Brown, Chad Michael Davis.

BIBLIOGRAFÍA COMPLEMENTARIA

- J2EE Tutorial, Stephanie Bodoff, Dale Green, Kim Haase, Eric Jendrock, Mónica Pawlan, Beth Stearns. *Addison-Wesley*.
- Practical Apache Struts2 Web 2.0 Projects, Ian Roughley
- Database Programming with JDBC and Java 2nd edition, George Reese, *O'Reilly, 2002*.
- Jakarta-Struts LIVE, Rick Hightower. *SourceBeat*, 2004.

CRONOGRAMA DE CLASES Y EVALUACIONES

Clase/Encuentro	Actividades teóricas y prácticas Contenidos	Evaluaciones previstas
Aplicaciones Web	Evolución de las aplicaciones web. Tecnologías Java: La plataforma empresarial Java (J2EE), Servidores J2EE, IDEs para desarrollo de aplicaciones Java. *Frameworks* para desarrollo de aplicaciones Java. Práctica sobre Servidores HTTP, protocolo HTTP, aplicaciones Web.	
Servlets	Características generales de los Servlets. Ciclo de vida de los Servlets. Servicios que proveen los Contenedores Web. Manejo de requerimientos y respuestas HTTP. Práctica de Servlets.	
Filtros	Características generales de los Filtros. Ciclo de vida de los Filtros. El objeto FilterChain Clases Wrappers. Requerimientos y respuestas customizados. Práctica sobre Filtros	
JavaServer Pages	Definición de páginas JavaServer Pages (JSP). Ciclo de vida de una JSP. Elementos para construcción de JavaServer Pages: elementos de scripting, directivas, acciones estándares. El lenguaje de expresiones (JSP EL). Práctica sobre JSP.	
MAVEN	Conceptos básico. Gestión de aplicaciones Java SE/EE usando MAVEN. Archetypes. Creación de proyectos con Maven. Funcionamiento. Ciclo de Vida, Fases, Plugins Dependencias, repositorios.	**ENTREGABLE 1** Prototipo **05/10/2018**
JavaServer Pages y Sesiones	Cómo mantener el estado con el protocolo HTTP? Mecanismos de intercambio de ID para manejar sesiones: cookies y URL *rewriting.* El objeto **HTTPSession:** ligar y eliminar elementos. Invalidar sesión. Soporte de sesiones en servlets y JSP: Sesiones en ambientes multi-servidores. Práctica sobre JSP y Sesiones.	**ENTREGABLE 2** Páginas Servlet, JSP y Sesiones **12/10/2018**
Tecnologías del lado del cliente	JavaScript, Librería JQuery, JSON Bootstrap	

JDBC	Persistencia. Tipos de persistencia con JAVA: Serialización, JDBC & SQL, Mapeo desde el modelo de objetos al modelo relacional. El estándar EJB 3.0 & Hibernate. Definición de capas de acceso a datos (DAOs)	**ENTREGABLE 3** Modelo de Datos **26/10/2018**
Hibernate	Hibernate. Arquitectura, Módulos. Ciclo de vida de objetos persistentes Mapeo nativo (usando XML) Mapeo con Anotaciones	
JPA	Estándares: EJB 3.0/JPA Motores JPA Mapeos JPA-QL Estructura de una aplicación usando JPA	
Spring Core	Evolución del framework Spring Core. Arquitectura del Framework. Inversión de Control (IoC) e Inyección de Dependencias (DI). Configuración de la aplicación por XML y por anotaciones.	
Spring MVC	Introducción a Spring MVC. Procesamiento de un requerimiento. Configuración y contextos de la aplicación. Internacionalización. Validación.	**ENTREGABLE 4** Hibernate y JPA **16/11/2018**
RESTful y Spring MVC	Implementación de web services mediante REST(Representational State Transfer). Principios de RESTful: identificación de recursos mediante URIs, uso de una interfaz uniforme y acotada. Implementando REST con Spring MVC.	
Angular	Implementación de una aplicación AngularJS que consuma los servicios REST implementados. Implementar la autenticación con Json Web Token(JWT)	**ENTREGABLE 5** Avances de Funcionalidad JPA + Spring CURSADA **14/12/2018**
Promoción de Final (1 instancia)		**ENTREGABLE 6** Avances de Funcionalidad Promoción con 5 **15/02/2019**
Promoción de Final (2 instancia)		**ENTREGABLE 7** Aprobado mayor a 5 (Sistema completo) **31/5/2019**

Contacto de la cátedra (mail, página, plataforma virtual de gestión de cursos):

- e_mail: lfava@info.unlp.edu.ar
- Plataforma virtual: https://catedras.info.unlp.edu.ar/

Firmas del/los profesores responsables