

Portada // TO DO //

Agradecimientos // TO DO //

Resumen // TO DO //

Palabras Clave // TO DO //

Índice general

1	Introducción	1
2	Fundamentos tecnológicos	3
2.1	Lenguajes utilizados	3
2.1.1	Java	3
2.1.2	JavaScript	3
2.1.3	TypeScript	4
2.1.4	HyperText Markup Language	4
2.1.5	Cascading StyleSheets	4
2.2	Frameworks y librerías	5
2.2.1	Spring	5
2.2.2	Java Persistence API	7
2.2.3	JAX-RS	7
2.2.4	Thymeleaf	7
2.2.5	jQuery	8
2.2.6	Bootstrap	8
2.3	Herramientas de desarrollo	9
2.3.1	Eclipse	9
2.3.2	Maven	9
2.3.3	Git	9
2.3.4	Oracle SQL Developer	10
2.4	Servidor de aplicaciones	10
2.5	Sin especificar	10
2.5.1	Ionic	10
2.5.2	AJAX	11
3	Metodología y planificación	13

4	Análisis	15
4.1	Análisis de requerimientos	15
4.1.1	Requerimientos funcionales	15
4.1.2	Requerimientos no funcionales	18
4.2	Modelo de casos de uso	18
4.2.1	Actores del sistema	18
4.2.2	Diagrama de casos de uso	18
4.2.3	Especificación casos de uso	19
5	Diseño	23
6	Implementación	25
7	Pruebas	27
8	Conclusiones	29
9	Trabajo futuro	31
10	Bibliografía	33

Índice de figuras

4.1	Diagrama Casos de Uso Aplicación Móvil.	19
-----	---	----

Índice de tablas

4.1	Caso de Uso: Autenticar Usuario	20
4.2	Caso de Uso: Registrar Usuario	22

Capítulo 1

INTRODUCCIÓN

Capítulo 2

FUNDAMENTOS TECNOLÓGICOS

2.1. Lenguajes utilizados

2.1.1. Java

Java es un lenguaje de programación de propósito general, concurrente, orientado a objetos. Fue diseñado para tener tan pocas dependencias de implementación como fuera posible tal que permitiera a los desarrolladores escribir el programa una vez y ejecutarlo en cualquier dispositivo sin necesidad de recompilarlo.

Fue originalmente desarrollado por James Gosling, de Sun Microsystems, la cual fue adquirida por la compañía Oracle.

Puede ejecutarse en cualquier máquina virtual Java (JVM) sin importar la arquitectura de la computadora subyacente y su sintaxis deriva en gran medida de lenguajes como C y C++, pero con menos utilidades de bajo nivel.

2.1.2. JavaScript

TypeScript es un lenguaje de programación interpretado, dialecto del estándar ECMAScript, orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.

Se utiliza principalmente del lado del cliente, implementado como parte de un navegador web, permitiendo mejoras en la interfaz de usuario y páginas web dinámi-

cas.

JavaScript se diseñó con una sintaxis similar a C, aunque adopta nombres y convenciones del lenguaje de programación Java. Sin embargo, Java y JavaScript tienen semánticas y propósitos diferentes.

2.1.3. TypeScript

TypeScript es un lenguaje de programación libre y de código abierto desarrollado y mantenido por Microsoft.

Este lenguaje es un superset del ya conocido JavaScript y que está pensado para grandes proyectos, los cuáles a través de un compilador de TypeScript se traducen a código JavaScript original.

Un aspecto característico de TypeScript es su sistema de tipos. Permite a los desarrolladores definir variables y funciones tipadas sin perder la esencia de JavaScript gracias a una representación estática de los tipos dinámicos. Definir tipos durante el diseño, nos ayudará a evitar errores en tiempo de ejecución.

2.1.4. HyperText Markup Language

HyperText Markup Language, conocido comúnmente por sus siglas, HTML, es un lenguaje de marcado cuya finalidad es la elaboración de páginas web. Define una estructura básica y un código (denominado código HTML) para la definición de contenido de una página web.

Es un estándar a cargo del World Wide Web Consortium (W3C), organización dedicada a la estandarización de casi todas las tecnologías ligadas a la web.

2.1.5. Cascading StyleSheets

Las hojas de estilo en cascada (o CSS, por sus siglas en inglés.) es un lenguaje de diseño gráfico para definir y crear la presentación de un documento estructurado escrito en un lenguaje de marcado. Especifica como se mostrarán por pantalla, o otro tipo de media, los denominados elementos HTML.

Junto con HTML y JavaScript, CSS es una tecnología usada por muchos sitios web para crear páginas web visualmente atractivas, interfaces de usuario de aplicaciones web y GUIs para muchas aplicaciones móviles.

Su principal objeto es mantener la separación del contenido del documento de su forma de presentación. Con las hojas de estilo se puede prescindir del uso de formatos de estilo dentro de la propia página HTML, de manera que se pueda modificar el estilo de toda una web modificando un único archivo CSS.

2.2. Frameworks y librerías

2.2.1. Spring

Spring es un framework cuya finalidad es facilitar el desarrollo de aplicaciones desarrolladas en Java. Es de código abierto y la primera versión fue elaborada por Rod Johnson. A pesar de que no impone ningún modelo de programación en particular, este framework se ha vuelto popular en la comunidad al ser considerado una alternativa, sustituto, e incluso un complemento al modelo Enterprise JavaBean (EJB)

Spring está compuesto de diversos módulos que se pueden agregar a nuestras aplicaciones, permitiendo a los desarrolladores agregar sólo los módulos que vayan usar. El único módulo necesario para trabajar con Spring es el Spring Core puesto que es el que contiene la DI (Inyección de Dependencias) y la configuración de uso de objetos Java.

Spring MVC

Spring MVC es un framework de aplicaciones web basado en el patrón MVC(model-viewcontroller) y que alberga todas las ventajas del framework de Spring.

- Separación clara de roles. Cada objeto controlador, validador, formulario, de modelo pueden ser realizados por objetos especializados.
- Configuración potente y directa. Capacidad de configuración que permite una fácil referencia a través de contextos, como por ejemplo, desde controladores

web a objetos de negocio.

- Adaptabilidad, flexibilidad y no intrusividad. Definir un controlador usando una de las anotaciones de parámetros (`@RequestParam`, `@PathVariable`,...) para un escenario dado.
- Código de negocio reutilizable. No existe necesidad de duplicación.

Spring MVC es, como otros frameworks MVC, basado en solicitud (request-driven). Están diseñados en torno a un Servlet central que sirve las solicitudes a los controladores y ofrece unas funcionalidades que facilitan el desarrollo de las aplicaciones web. Sin embargo, el `DispatcherServlet` de Spring es más que eso, está completamente integrado con el contenedor Spring IoC y permite hacer uso de las características y funcionalidades de Spring.

Spring Security

Spring Security ofrece exhaustivos servicios de seguridad para las aplicaciones empresariales basadas en Java EE. Las dos principales áreas en las que se enfoca Spring Security son la Autenticación y la Autorización, probablemente, los dos temas más relevantes en la seguridad de las aplicaciones.

- Autenticación es el proceso por el que se determina que uno es el que dice ser.
- Autorización hace referencia al proceso de determinar qué acción o acciones puede realizar en la aplicación.

A nivel de autenticación, Spring Security soporta un amplio rango de modelos de autenticación. La mayoría de estos modelos de autenticación son proporcionados por terceros, o desarrollados por los organismos estándar pertinentes, como Internet Engineering Task Force. A mayores, Spring Security provee su propio conjunto de mecanismos de autenticación y soporta integración de autenticación con diferentes tecnologías

2.2.2. Java Persistence API

Java Persistence API, comúnmente conocida por sus siglas JPA, es la API que describe la gestión de datos relacionales en aplicaciones que utilicen Java. La primera especificación fue lanzada en mayo de 2006 como parte del trabajo del JSR 220.

JPA en sí mismo es solo una especificación, no un producto. Son un conjunto de interfaces que requieren una implementación. Existen implementaciones de JPA de código abierto y comerciales, y cualquier servidor de aplicaciones JAVA EE 5 debe proporcionar soporte para su uso.

El objetivo de esta API es no perder las ventajas de la orientación a objetos al interactuar con una base de datos y permitir usar objetos regulares, comúnmente conocidos como POJOs.

2.2.3. JAX-RS

JAX-RS es la API de Java para la elaboración de servicios web RESTful que brinda soporte en la creación de servicios web de acuerdo con el patrón arquitectónico REST. Desde la versión 1.1, JAX-RS es una parte oficial de Java EE 6. Una característica notable de ser parte oficial de Java EE es que no es necesaria ninguna configuración para comenzar a utilizar JAX-RS.

Esta API utiliza anotaciones, introducidas en Java SE 5, para simplificar el desarrollo y la implementación de clientes y recursos web.

De la misma forma que sucedía con JPA, JAX-RS no es más que una especificación, necesita un producto que la implemente. Jersey y RESTEasy son implementaciones de JAX-RS,

2.2.4. Thymeleaf

Thymeleaf es una librería Java que implementa un motor de plantillas válidas para entornos web como independientes. Es un software de código abierto creado originalmente por un ingeniero de software español llamado Daniel Fernández. No está hecho ni respaldado por ningún software de ninguna compañía y se ofrece al público de manera totalmente gratuita, tanto en formato binario como en código

fuentes, bajo licencia Apache.

Su objetivo principal es la creación de plantillas de una manera elegante y con un código bien formateado.

Thymeleaf ofrece una buena integración con Spring MVC a través de su dialecto SpringStandard, pero esta integración con Spring es completamente opcional y el dialecto estándar está destinado a usarse sin Spring.

2.2.5. jQuery

jQuery es una librería multiplataforma de JavaScript, creada inicialmente por John Resig. Es un software libre y de código abierto, y posee un doble licenciamiento bajo la licencia MIT y la licencia Pública General de GNU, permitiendo su uso en proyectos libres y privados.

Su objetivo es la realización de funcionalidades basadas en JavaScript de forma rápida y sencilla. Permite realizar recorridos y manipulaciones de documentos HTML, manejar eventos, animaciones y usar AJAX mucho más simple con una API fácil de usar que funciona en multitud de navegadores.

2.2.6. Bootstrap

Bootstrap es un kit de herramientas de código abierto para desarrollo web junto a HTML, CSS y JS.

Originalmente creado por un diseñador y desarrollador de Twitter, Bootstrap es uno de los frameworks front-end y proyectos de código abierto más populares en el mundo. Antes de ser un framework de código abierto, Bootstrap era conocido como Twitter Blueprint.

Bootstrap incluye plantillas de diseño basadas en HTML y CSS para tipografías, formularios, botones, tablas,... así como complementos de JavaScript opcionales. También brinda la capacidad de crear fácilmente diseños receptivos.

2.3. Herramientas de desarrollo

2.3.1. Eclipse

Eclipse es una plataforma software compuesto por un conjunto de herramientas de programación de código abierto multiplataforma.

Fue desarrollado originalmente por IBM como el sucesor de su familiar de herramientas VisualAge. Ahora, está siendo desarrollado por la Fundación Eclipse, una organización independiente, sin ánimo de lucro, que fomenta una comunidad de código abierto y un conjunto de productos complementarios.

2.3.2. Maven

Maven es una herramienta de software para la gestión y construcción de proyectos Java creada por Jason van Zyl en 2002 que tiene un modelo de configuración de construcción basado en formato XML.

Maven utiliza un Project Object Model, conocido como POM, para describir el proyecto software a construir, sus dependencias de otros módulos y componentes externos, y el orden de construcción de los elementos.

Una de sus características clave son su disponibilidad para usarse en la red puesto que el motor incluido en su núcleo puede dinámicamente descargar plugins de un repositorio.

2.3.3. Git

Git es un sistema de control de versiones distribuido de código abierto y gratuito diseñado para manejar todo, desde proyectos pequeños a muy grandes, con velocidad y eficiencia.

Originalmente fue diseñado como un motor de sistema de control de versiones de bajo nivel sobre el cual otros podían codificar interfaces frontales. Desde entonces hasta ahora, el núcleo del proyecto Git se ha vuelto un sistema de control de versiones completo, utilizable en forma directa.

Git es fácil de aprender y ofrece un rendimiento increíblemente rápido. Su principal objetivo es llevar el registro de cambios en archivos y coordinar el trabajo que varias personas realizan sobre archivos compartidos.

2.3.4. Oracle SQL Developer

Oracle SQL Developer es un entorno de desarrollo integrado y gratuito que simplifica el desarrollo y la administración de las bases de datos de Oracle, tanto de implementaciones tradicionales como en la nube.

Este software admite productos de Oracle y una grana variedad de complementos de terceros que los usuarios pueden implementar para conectarse a bases de datos que no sean de Oracle.

SQL Developer ofrece un desarrollo completo de extremo a extremo de sus aplicaciones PL / SQL, una hoja de trabajo para ejecutar consultas y scripts, una consola DBA para administración, una interfaz de informes y una solución completa de modelado de datos.

2.4. Servidor de aplicaciones

2.5. Sin especificar

2.5.1. Ionic

Ionic es un completo SDK de código abierto diseñado para el desarrollo de aplicaciones móviles híbridas. La versión original fue lanzada en 2013 y construida sobre AngularJS y Apache Cordova. Las versiones más recientes, conocidas como Ionic 3, están construidas sobre Angular.

Proviene de herramientas y servicios para desarrollar aplicaciones móviles híbridas usando tecnologías web (CSS, HTML,...).

Ionic proviene de toda la funcionalidad que se puede encontrar en los SDK de desarrollo móvil. Las aplicaciones construidas se pueden personalizar en función del

uso final, Android o iOS, por ejemplo.

Además del SDK, Ionic también brinda servicios que los desarrolladores pueden usar para habilitar funciones, como la tecnología push, test A/B, construcciones automáticas, etc...

Ionic también proporciona una poderosa interfaz de línea de comandos, lo que permite poder comenzar a desarrollar un proyecto con un simple comando. También permite a los desarrolladores agregar complementos de Cordova.

2.5.2. AJAX

Capítulo 3

METODOLOGÍA Y PLANIFICACIÓN

Capítulo 4

ANÁLISIS

En este apartado se explicará, detalladamente, el análisis realizado.

4.1. Análisis de requerimientos

4.1.1. Requerimientos funcionales

A continuación se muestran los requerimientos funcionales del sistema, clasificados en distintas áreas.

Acceso a la aplicación

- El sistema ofrecerá la posibilidad de que un usuario se registre en la aplicación.
- El sistema ofrecerá la posibilidad de que el usuario se identifique en el sistema.
Los usuarios deben ingresar al sistema con nombre de usuario y contraseña.

Aplicación Móvil

- El sistema ofrecerá la posibilidad de crear nuevas rutas.
- El usuario podrá consultar las rutas, propias y de otros usuarios, según ciertos criterios de búsqueda.

- El sistema permitirá a los usuarios autorizados eliminar las rutas propias que deseen.
- El sistema permitirá a los usuarios autorizados a consultar los detalles de las rutas.
- Para las rutas, el sistema permitirá:
 - Establecer las fechas de inicio y fin.
 - Consultar, asignar y desasignar a la ruta, los eventos disponibles en esas fechas.
 - Consultar el itinerario por días.
 - Modificar la hora de comienzo establecida para cada día.
 - Consultar, añadir y eliminar lugares de interés a cada día de la ruta.
 - Editar el modo de viaje a realizar entre diferentes lugares.
 - Mostrar el itinerario, por días y en total, en el mapa.
 - Habilitar y deshabilitar el sistema de geolocalización para conocer la ruta hecha en tiempo real.
 - Consultar y comparar, el itinerario definido con el obtenido a tiempo real.
 - Editar los permisos de la ruta.

Aplicación Web

- El usuario podrá consultar las rutas existentes, propias y de otros usuarios, según ciertos criterios de búsqueda.
- El sistema permitirá a los usuarios autorizados a consultar los detalles de las rutas.
- El sistema permitirá a los usuarios autorizados marcar las rutas propias como privadas, con el fin de no compartirlas con los demás usuarios.
- El sistema solo ofrecerá la posibilidad de consulta sobre los detalles de una ruta, permitiendo ver el itinerario, si tiene datos en tiempo real guardados, etc...
- El sistema permitirá a los usuarios autorizados eliminar las rutas propias que deseen.

Administración Web

- El sistema solo permitirá acceso a usuarios con permisos de administración.
- El sistema permitirá a los usuarios con dichos permisos, dar de alto nuevos usuarios.
- El sistema permitirá las altas, bajas, modificaciones y consultas de las entidades del sistema.
 - El sistema ofrecerá la posibilidad de crear, eliminar, modificar y consultar datos de usuarios.
 - El sistema ofrecerá la posibilidad de crear, eliminar, modificar y consultar datos de rutas.
 - El sistema ofrecerá la posibilidad de crear, eliminar, modificar y consultar datos de lugares.
 - El sistema ofrecerá la posibilidad de crear, eliminar, modificar y consultar datos de categorías.
 - El sistema ofrecerá la posibilidad de crear, eliminar, modificar y consultar datos de eventos.

Seguridad

- El sistema ofrecerá la posibilidad de que el usuario modifique sus datos de acceso al sistema.
- El sistema solo permitirá acciones correctamente autenticadas, exceptuando las de acceso a la aplicación.
- Los usuarios de la aplicación solo podrán modificar los datos a los que el usuario esté autorizado. Un usuario no podrá modificar la información de los recursos de los que no es propietario.
- Los intercambios de datos que realice el sistema a través de internet, serán mediante el uso del protocolo encriptado https.

4.1.2. Requerimientos no funcionales

Rendimiento

Disponibilidad

Estabilidad

Escalabilidad

4.2. Modelo de casos de uso

4.2.1. Actores del sistema

4.2.2. Diagrama de casos de uso

Tras conocer los requerimientos funcionales del sistema, se ha optado por la elaboración de dos diagramas de casos de uso. El primero con las necesidades del cliente móvil y el segundo con las de la parte web.

Para el cliente móvil, presentamos el siguiente diagrama.

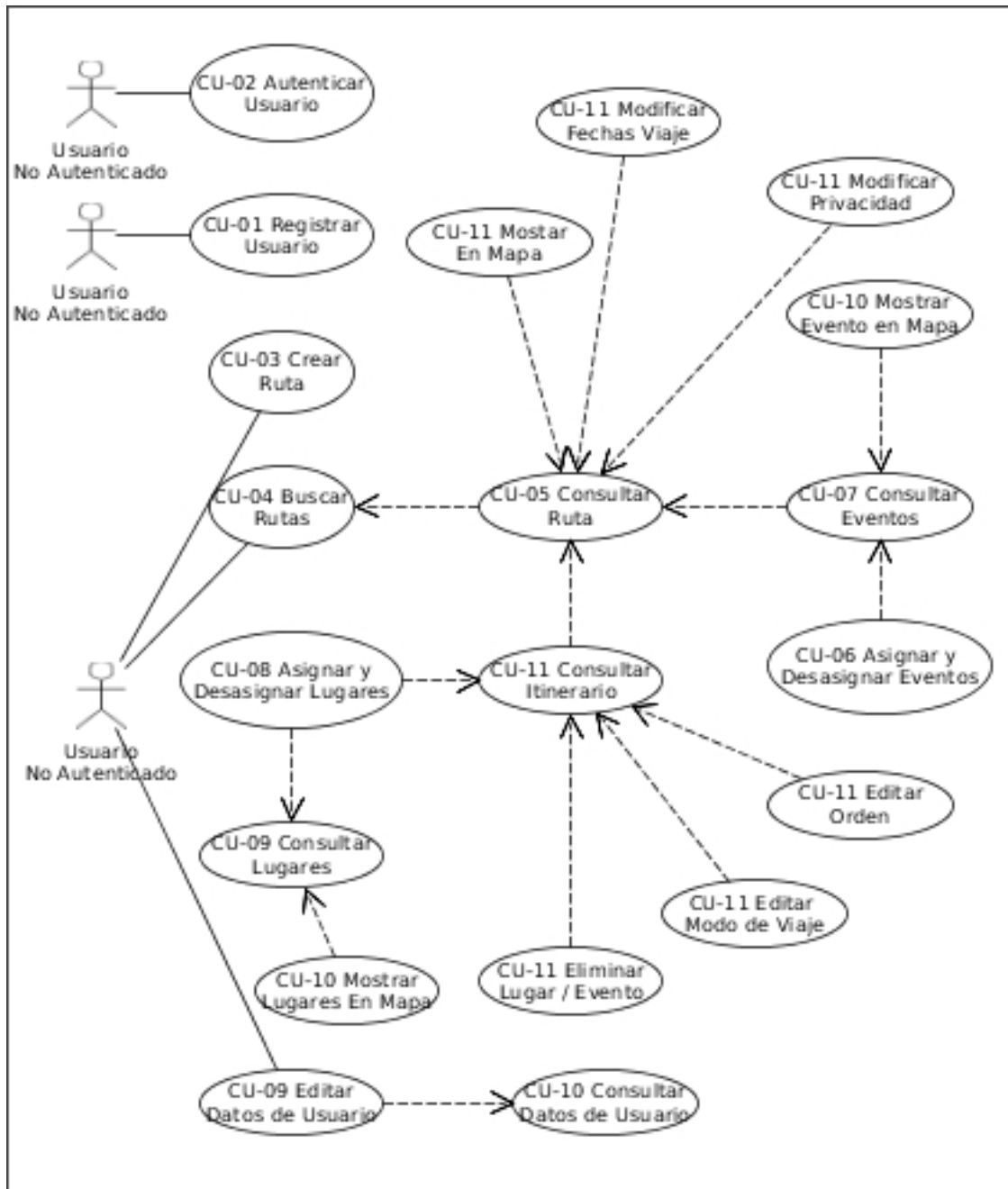


Figura 4.1: Diagrama Casos de Uso Aplicación Móvil.

4.2.3. Especificación casos de uso

Autenticar Usuario

CU<01> Autenticar Usuario	
Descripción	El usuario se identifica introduciendo las credenciales de acceso en el sistema
Actores	Usuario Administrador Moderador
Precondiciones	N/A
Secuencia Normal	<ol style="list-style-type: none">1. El usuario introduce sus credenciales en la ventana de login.2. El usuario pulsa el botón de <i>Acceder</i>.3. El sistema valida las credenciales.4. El usuario accede a la aplicación.
Excepciones	<p>3a. Los datos introducidos no son correctos.</p> <ol style="list-style-type: none">1. El usuario vuelve a introducir las credenciales.
Postcondiciones	El usuario queda autenticado en el sistema

Tabla 4.1: Caso de Uso: Autenticar Usuario

Registrar Usuario

CU<02> Registrar Usuario	
Descripción	El usuario introduce los datos para darse de alta en la aplicación.
Actores	Usuario
Precondiciones	N/A
Secuencia Normal	<ol style="list-style-type: none">1. El usuario selecciona la opción de registrarse.2. El sistema muestra un formulario indicando los campos necesarios para realizar el registro.3. El usuario rellena los campos y pulsa el botón de <i>Registrarse</i>.4. El sistema valida los datos introducidos por el usuario.5. El usuario accede a la aplicación.
Excepciones	<p>3a. El usuario pulsa el botón de <i>Cancelar</i>.</p> <ol style="list-style-type: none">1. El sistema cancela el registro y redirige al usuario a la pantalla de login.2. Implementa CU-<01> Autenticar Usuario <p>3a. Los datos introducidos por el usuario no son válidos.</p> <ol style="list-style-type: none">1. El sistema muestra un mensaje de error e invita al usuario a volver a introducir los datos. (Regreso al paso 2)

Continúa en la siguiente página

Continúa de la página anterior

Postcondiciones	El usuario queda registrado y autenticado en el sistema
------------------------	---

Tabla 4.2: Caso de Uso: Registrar Usuario

Capítulo 5

DISEÑO

Capítulo 6

IMPLEMENTACIÓN

Capítulo 7

PRUEBAS

Capítulo 8

CONCLUSIONES

Capítulo 9

TRABAJO FUTURO

Capítulo 10

BIBLIOGRAFÍA