Examen Final Aplicaciones Web 2024-01

Aplicaciones Web (Universidad Peruana de Ciencias Aplicadas)

SI730 - APLICACIONES WEB
EXAMEN FINAL
2024-1

Sección: SI91, SV51, SW51, SW52, SW53, SW55, SW56, WS51, WS52, WS53, WX53, WX54, WX55, WX56

Profesores: Bautista Ubillús, Efraín Ricardo
Cáceres Honores, Francisco José
Mori Paiva, Hugo Allan
Reupo-Musayón Gastulo, Naldo
Sánchez Ponce, Alex Humberto
Tinoco Licas, Juan Carlos
Velásquez Núñez, Ángel Augusto

Duración: 170 minutos

Indicaciones:

1. El examen consta de 1 pregunta, y tendrá **170 minutos** para resolver y entregar.
2. La pregunta es de tipo Proyecto de Software y la entrega de su respuesta es a través de envío de archivo empaquetado **.zip** (único formato válido) con nombre *upc-pre-202401-si730-<sección>-eb-u<código-estudiante>.zip* (por ejemplo, upc-pre-202401-si730-wx59-eb-u201621873.zip), conteniendo el proyecto de software, en la Actividad para el Examen final.
3. Puede utilizar como referencia los materiales publicados en el aula virtual, los sitios referenciados en el enunciado, así como sitios web de documentación de frameworks o tecnologías referenciadas.
4. Cada examen cuenta con un equipo académico, el cual estará conectado **durante el examen**.
5. El alumno debe dedicar los primeros 15 minutos a revisar las preguntas del examen y de presentarse alguna duda enviar un correo al(los) profesor(es)

   Secciones SI91, WX54, WX55: Morales Arévalo, Juan Carlos al correo pcsijumo@upc.edu.pe
   Secciones WS51, WS52, WS53: Bautista Fuentes, Iván Christian al correo pcsiibau@upc.edu.pe
   Secciones WX56, SW51, SW52: Rojas Malasquez, Royer Edelwer al correo pcisrroj@upc.edu.pe
   Secciones SW53, SW55, SW56: Prialé De la Peña, Mónica Rosario al correo pcsimpri@upc.edu.pe
   Secciones SV51, WX53: García Rojas, Fidel Eugenio al correo pcsifgar@upc.edu.pe

6. Los profesores en mención, solo recibirán correos provenientes de las cuentas UPC, de ninguna manera se recibirán correos de cuentas públicas.
7. Ante problemas técnicos, debe de forma obligatoria adjuntar **evidencias** del mismo, como capturas de pantalla, videos, fotos, etc. Siendo **requisito fundamental** que, en cada evidencia se pueda apreciar claramente la **fecha y hora del sistema operativo del computador** donde el alumno está rindiendo el examen.
8. No se recibirán problemas técnicos una vez culminado el examen.

Enunciado:

Caso IRRIOT, inc.

IRRIOT (https://www.irriot.com/), que significa ***IRR****igation* ***I****nternet* ***O****f* ***T****hings*, ofrece una plataforma completa para la automatización del riego inteligente inalámbrico. La oficina principal está ubicada en Estocolmo, Suecia, pero operan a nivel global brindando soporte a clientes de todo el mundo.

IRRIOT desarrolló una solución de riego inalámbrico IoT, que aborda los problemas de movilidad y asequibilidad para los clientes. Ellos utilizan lo último en comunicación por radio de ultra largo alcance para eliminar todo el costoso cableado en el campo. Las estaciones de riego alimentadas por energía solar son respetuosas con el medio ambiente y no requieren mantenimiento. En la columna vertebral tienen un controlador de riego industrial a gran escala, que incluye acceso a un IoT Cloud para el controlador, inteligencia, alarmas y pronóstico del tiempo a través de aplicaciones web y móviles.

La empresa, fundada en 2017, se ha posicionado de forma sólida en el sector de grandes empresas, teniendo entre sus clientes a varias corporaciones listadas en Fortune 500.

Como parte de su estrategia de mercado, desea penetrar en el sector de pequeñas y medianas empresas. Para ello, ha iniciado el proceso de análisis y diseño de un IoT Irrigation System de bajo costo denominado EasyIRRIOT.

Solution Features

EasyIRRIOT se basa en una infraestructura para riego con nodos inteligentes, donde cada nodo es un IoT Device que incluye sensor de temperatura, sensor de humedad, real time clock, display y capacidad de control de hasta cuatro válvulas de riego. Cada nodo se conecta a un Edge Server. El Edge Server se conecta con el IoT Cloud. Los clientes pueden acceder a la información, analíticos de operación y herramientas de control de la solución a través de EasyIRRIOT Web Platform o EasyIRRIOT Mobile App.

El IoT Device debe soportar las siguientes modalidades de operación

SCHEDULE_DRIVEN
TEMPERATURE_DRIVEN
HUMIDITY_DRIVEN

Pregunta 1 (20 p.).

Usted se integra al backend software developer team, a cargo de la creación de un **RESTful API** que brinde soporte a las operaciones de IRRIOT.

El ecosistema de EasyIRRIOT requiere que el EasyIRRIOT Web Application, EasyIRRIOT Mobile App y el Embedded IoT System cuenten con **Endpoints** en el RESTful API, para el manejo de la información de:

Things, conformadas por los atributos id, serialNumber, model, operationMode, maximumTemperatureThreshold, minimumHumidityThreshold
ThingStates, conformadas por los atributos id, thingSerialNumber, currentOperationMode, currentTemperature, currentHumidity, collectedAt.

Como reglas de negocio, IRRIOT:

- No permite que se registre dos **things** con el mismo valor de **serialNumber**.
- Especifica que **serialNumber** es un identificador único que IRRIOT genera para sus dispositivos, el cual sirve para identificarlos en los inventarios. Internamente es un owned type con un valor UUID que debe generarse al momento de registrarse.
- **EOperationMode** es un enumeration que representa el modo de operación soportado por el dispositivo, teniendo como posibles valores: *ScheduleDriven* (0), *TemperatureDriven* (1) y *HumidityDriven* (2).
- Establece que la información que se desea preservar de cada ***Thing,*** incluye **id** (int, Primary Key, Autogenerado), **serialNumber** (identificador del negocio, Obligatorio, Autogenerado), **model** (string, Obligatorio, no vacío), **operationMode** (EOperationMode, obligatorio), **maximumTemperatureThreshold** (decimal, Obligatorio), **minimumHumidityThreshold** (decimal, Obligatorio).
- Especifica que el valor de **operationMode** para **thing** no se debe solicitar al momento del ingreso, se debe asignar por defecto al momento del registro el valor *ScheduleDriven*.
- Requiere que los valores válidos para **maximumTemperatureThreshold** solo sean decimales entre -40.00 y 85.00.
- Requiere que los valores válidos para **minimumHumidityThreshold** solo sean decimales entre 0.00 y 100.00.
- Establece que la información de cada ***Thing State***, incluye **id** (int, Primary Key, Autogenerado), **thingSerialNumber** (identificador del negocio, Obligatorio), **currentOperationMode** (int, obligatorio), **currentTemperature** (decimal, obligatorio), **currentHumidity** (decimal, obligatorio), **collectedAt** (datetime, obligatorio).
- Especifica que el atributo **thingSerialNumber** debe ser internamente un owned type y el valor UUID contenido debe corresponder con el valor de *serialNumber* para un *Thing* previamente registrado.
- Especifica que un **Thing** puede estar asociado con uno o muchos **Thing States**, pero un **Thing State** solo puede provenir de un **Thing** en particular.
- Especifica que no se puede agregar un **Thing State** con una combinación ya existente de **thingSerialNumber** y **collectedAt**.
- Requiere que el valor de **collectedAt** no sea mayor que la fecha actual.
- Requiere que los valores válidos para **currentOperationMode** solo sean enteros entre 0 y 2.
- Especifica que tanto **Thing** como **Thing State** pertenecen a diferentes bounded contexts, por lo que necesita un Anti-Corruption Layer (ACL) para que el bounded context que lo requiera, acceda a capacidades que exponga el otro bounded context como parte de la implementación de una característica requerida.
- Especifica que tanto **Thing** como **Thing State** son Aggregate Roots en sus respectivos bounded contexts, por lo que se requiere contar con atributos de auditoría para registrar **createdAt** (fecha y hora de creación de registro) y **updatedAt** (fecha y hora de última actualización de registro), de uso interno y poblados de forma automática por el sistema al momento de las operaciones de creación o actualización.

- Requiere que el atributo **operationMode** de **Thing** se mantenga actualizado. Por ello, al momento de registrar un **Thing State**, el valor del atributo *currentOperationMode* incluido debe servir de base para mantener actualizado el **operationMode** del **Thing** correspondiente.

Durante la etapa de desarrollo, le asignan trabajar en específico sobre dos Endpoints:

/api/v1/things
/api/v1/thing-states/

Incluya como parte del desarrollo la implementación de todas las reglas de negocio, así como las especificaciones a nivel de requests y responses.

Things Endpoint (https://localhost:{port}/api/v1/things)
Debe implementar **solo dos** operaciones en el RESTful API: agregar (POST) un **thing** y consultar (GET) un **thing** por **id**. Los valores de **id** son autogenerados al momento de almacenar la información. Los valores de **serialNumber** son autogenerados al momento de almacenar la información. Al agregar se debe retornar en el response el status 201 (created). Para POST y para GET incluya en el response un objeto con el id generado para el thing y sus demás atributos, incluyendo el serialNumber generado, así como el operationMode con su representación en string. No incluya en el response los atributos de auditoría.

Thing States Endpoint (https://localhost:{port}/api/v1/thing-states)
Debe implementar **solo una** operación sobre los **thing states** en el RESTful API: agregar (POST) un **thing state**. Al agregar se debe retornar en el response el objeto incluyendo el id generado para el mismo y los demás atributos, incluyendo el thingSerialNumber. No incluya en el response los atributos de auditoría.

Technical constraints

1. Elabore la solución con C#, NET 8 y ASP.NET Core Framework.
2. Cree su solución y el proyecto de software con el nombre ***si730ebu<código-estudiante>.API*** (por ejemplo, *si730ebu201621873.API*).
3. Considere que el aggregate **Thing** pertenece al bounded context **inventory** y el aggregate **ThingState** pertenece al bounded context **observability**.
4. Considere el bounded context **shared** con los elementos que correspondan.
5. La información debe ser persistente en una base de datos relacional (MySQL), en un esquema **irriot**.
6. Aplique buenas prácticas de Arquitectura de Software, enfoque de Domain-Driven Design, separación en bounded contexts, layered architecture (Domain, Application, Interfaces, Infrastructure), patrones de strategic y tactical Domain-Driven Design, patrón CQRS, patrón Anti-Corruption Layer (ACL), principios y patrones de diseño, convenciones de nomenclatura en **inglés**, así como buenas prácticas de nomenclatura en C# (entre ellas Upper Camel Case para Namespaces, Clases, Records, Properties, Métodos; Lower Camel Case para atributos privados) y buenas prácticas para nomenclatura de objetos de Base de Datos (entre ellas snake case, tablas en plural, sin mnemónicos, id como nombre de primary key).
7. Utilice minúsculas para los nombres de URL y términos compuestos separados por guión medio (-) para todos los endpoints.
8. Utilice Properties para representar los atributos de las clases Entity.
9. Utilice records en vez de clases para almacenamiento de valores inmutables.
10. Utilice el patrón Assembler para el Object Mapping en la sección Transform de interfaces layer.
11. Para **thing** y **thing state**, incluya properties de auditoría *CreatedAt* y *UpdatedAt* con valores generados de forma automática al momento de la creación. Utilice para ello la dependencia *EntityFrameworkCore.CreatedUpdatedDate*.
12. Utilice Fluent API para implementar las reglas de Object-Relational Mapping.
13. Documente su código en inglés, con **XML documentation comments**, colocando información de propósito para principales objetos de programación, así como propósito, parámetros y valor retornado en clases y

métodos relevantes. Incluya como parte de la documentación sus nombres y apellidos como valor en un tag <remarks>.

14. Incluya documentación de Endpoints con OpenAPI.
15. Considere la gestión de excepciones en la aplicación.
16. Empaquete su solución como un archivo **.zip**. **(único formato válido)** con el nombre ***upc-pre-202401-si730-<sección>-eb-u<código-estudiante>.zip*** (por ejemplo, *upc-pre-202401-si730-wx59-eb-u201621873.zip*).
17. Suba su archivo de solución en la Actividad indicada para el Examen final.

NO forma parte del alcance del proyecto:

1. Soporte de CORS.
2. Security.
3. Testing.

Rúbrica de calificación

Criterio de Calificación	Sobresaliente (S)	Esperado (E)	Necesita Mejorar (M)	Insuficiente (I)	Calificación
C01. Building y ejecución	Al abrir el proyecto y ordenar la ejecución, ésta se inicia sin problemas. El API es accesible en la ruta indicada.	La aplicación no llega a iniciar y ejecutarse, sin embargo el proceso de building llega a concluir.	Al cargar el proyecto el proceso de building presenta errores y no llega a concluir.	No elabora solución	
	2.0 puntos	**1.0 punto**	**0.5 puntos**	**0 puntos**	
C02. Things Endpoint	El RESTful API expone el endpoint especificado en el enunciado. Se evidencia la funcionalidad de las operaciones solicitadas, proporcionando en cada caso los valores esperados y respondiendo adecuadamente ante las excepciones. Se evidencia la persistencia de los objetos solicitados, cumpliendo con la estructura según enunciado, en una tabla nombrada según especificaciones, en base de datos relacional según enunciado en el esquema indicado. Se incluye documentación del Endpoint con OpenAPI.	El RESTful API expone el endpoint especificado en el enunciado. Se evidencia parcialmente la funcionalidad de las operaciones solicitadas, proporcionando en algunos casos los valores esperados y respondiendo de forma parcialmente adecuada ante las excepciones, o se evidencia parcialmente la persistencia en base de datos relacional, o se evidencia parcialmente la persistencia de los objetos solicitados con estructura según enunciado, en una tabla *nombrada según especificaciones* en base de datos relacional según enunciado en el esquema *indicado*.	La aplicación implementa y expone el endpoint especificado en el enunciado, pero no cumple con la ruta especificada o no se puede registrar elementos.	La aplicación no implementa o expone el endpoint solicitado.	
	4.0 puntos	**3.0 puntos**	**1.5 puntos**	**0 puntos**	
C03. Thing States Endpoint	El RESTful API expone el endpoint especificado en el enunciado. Se evidencia la funcionalidad de las operaciones solicitadas, proporcionando en cada caso los valores esperados y respondiendo adecuadamente ante las excepciones. Se evidencia la persistencia de los objetos solicitados, cumpliendo con la estructura según enunciado, en una tabla nombrada según especificaciones, en base de datos relacional según enunciado en el esquema indicado. Se incluye documentación del Endpoint con OpenAPI.	El RESTful API expone el endpoint especificado en el enunciado. Se evidencia parcialmente la funcionalidad de las operaciones solicitadas, proporcionando en algunos casos los valores esperados y respondiendo de forma parcialmente adecuada ante las excepciones, o se evidencia parcialmente la persistencia en base de datos relacional, o se evidencia parcialmente la persistencia de los objetos solicitados con estructura según enunciado, en una tabla *nombrada según especificaciones* en base de datos relacional según enunciado en el esquema *indicado*.	La aplicación implementa y expone el endpoint especificado en el enunciado, pero no cumple con la ruta especificada o no se puede registrar elementos.	La aplicación no implementa o expone el endpoint solicitado.	
	4.0 puntos	**3.0 puntos**	**1.5 puntos**	**0 puntos**	
C04. Business Rules	El desarrollo incluye la implementación de reglas de negocio, cubriendo de forma completa las condiciones y escenarios establecidos, siendo éstas ejecutables, con adecuado manejo de excepciones, implementando éstas en las capas más adecuadas, aplicando convenciones y buenas prácticas.	El desarrollo incluye la implementación de la mayoría de reglas de negocio, cubriendo de forma parcial las condiciones y escenarios establecidos, siendo éstas ejecutables, implementándolas en las capas adecuadas en la mayoría de casos, ó aplicando parcialmente convenciones y buenas prácticas.	El desarrollo incluye la implementación de algunas de las reglas de negocio, incumpliendo la mayoría de condiciones y escenarios establecidos, siendo éstas ejecutables, implementándolas en las capas adecuadas en muy pocos casos, ó con poca evidencia de aplicar convenciones y buenas prácticas.	No implementa reglas de negocio, o no cubre escenarios más allá de operaciones CRUD básicas, o éstas no son ejecutables.	
	4.0 puntos	**3.0 puntos**	**1.5 puntos**	**0 puntos**	
C05. Code Organization	El desarrollador organiza el código y los elementos de backend de la solución, aplicando buenas prácticas de C#, .NET Framework Core, ASP.NET Core y Domain-Driven Design, agrupando los elementos de la solución según convenciones, manteniendo organización de paquetes y carpetas recomendadas por el fabricante y buenas prácticas de la industria de software. Cumple de forma completa con los technical constraints sobre organización.	El desarrollador aplica en la mayoría de casos para el backend convenciones, recomendaciones y buenas prácticas de C#, .NET Framework Core, ASP.NET Core y Domain-Driven Design. Cumple con la mayoría de technical constraints sobre organización.	El desarrollador en algunos casos para el backend convenciones, recomendaciones y buenas prácticas de C#, .NET Framework Core, ASP.NET Core y Domain-Driven Design. Cumple de con solo algunos de los technical constraints sobre organización.	No se evidencia un criterio de organización para los elementos de la solución.	
	2.0 punto	**1.0 puntos**	**0.5 puntos**		
C06. Code Quality	Utiliza para el backend el lenguaje de programación C# La codificación tiene un estilo claro, indentando los bloques de código según los estándares de programación correspondientes al lenguaje, aplicando una lógica consistente en los métodos, condicionales sin escenarios no contemplados, uso adecuado de reutilización de código para evitar redundancia. Aplica patrones de arquitectura, principios y patrones de diseño. Distribuye el código en los niveles correspondientes, asignando lógica de persistencia, lógica de negocio, lógica de control, lógica de mapping y transferencia a las interfaces y clases que corresponden. Cumple de forma completa con los technical constraints.	Utiliza para el backend el lenguaje de programación C#. La codificación es funcional, aplica en la mayoría de casos los estándares de indentación de bloques de código, ó existen algunas ineficiencias en la codificación: redundancia ó inconsistencias en la lógica de programación. Aplica parcialmente patrones de arquitectura y patrones de diseño, o existe en algunas partes una distribución de la lógica en los niveles incorrectos. Cumple con la mayoría de los technical constraints.	Utiliza para el backend el lenguaje de programación C#. La codificación es funcional, pero sólo aplica algunos de los estándares de indentación de bloques de código, ó existen muchas ineficiencias en la codificación: redundancia ó inconsistencias en la lógica de programación. Aplica algunos patrones de arquitectura y patrones de diseño, o existe en muchos casos una distribución de la lógica en los niveles incorrectos. Cumple de con solo algunos de los technical constraints.	No utiliza el lenguaje de programación C# para el backend, ó la codificación es funcional pero no se evidencia aplicación de estándares ó criterios de eficiencia en la codificación, con ausencia de comentarios, ó no aplica patrones de arquitectura ni patrones de diseño, o la codificación no es funcional.	
	3.0 puntos	**2.0 punto**	**1.0 puntos**	**0 puntos**	
C07. Naming Standards	El desarrollador aplica en todos los nombres de objetos de programación como namespaces, componentes, interfaces, clases, objetos, variables, constantes y métodos la nomenclatura en inglés y la nomenclatura estándar para identificadores de clases, objetos, miembros de programación, así como los recursos. Cumple de forma completa con los technical constraints sobre nomenclatura.	El desarrollador aplica en la mayoría de casos la nomenclatura en inglés y la nomenclatura estándar para identificadores de clases, objetos, miembros de programación, así como los recursos. Cumple con la mayoría de technical constraints sobre nomenclatura.	El desarrollador aplica en muy pocos casos la nomenclatura en inglés y la nomenclatura estándar para identificadores de clases, objetos, miembros de programación, así como los recursos. Cumple con solo algunos de los technical constraints sobre nomenclatura.	El desarrollador no aplica nomenclatura en inglés para los objetos de programación ó recursos.	
	1.0 puntos	**0.5 punto**	**0.25 puntos**	**0 puntos**	
Total	**20 puntos**	**13.5 puntos**	**6.75 puntos**	**0 puntos**	

Lima, 3 de Julio del 2024

Anexos

Anexo A. Referencias

Comprimir y descomprimir archivos: https://support.microsoft.com/es-es/windows/comprimir-y-descomprimir-archivos-8d28fa72-f2f9-712f-67df-f80cf89fd4e5

REST API Tutorial: https://restfulapi.net/

Swashbuckle.AspNetCore - OpenAPI Library for ASP.NET Core:
https://github.com/domaindrivendev/Swashbuckle.AspNetCore

MySQL Connector/NET for Entity Framework – Entity Framework Core Support:
https://dev.mysql.com/doc/connector-net/en/connector-net-entityframework-core.html

Properties
https://learn.microsoft.com/en-us/dotnet/csharp/programming-guide/classes-and-structs/properties

Entity Properties
https://learn.microsoft.com/en-us/ef/core/modeling/entity-properties?tabs=fluent-api%2Cwithout-nrt

EntityFrameworkCore.CreatedUpdatedDate
https://www.nuget.org/packages/EntityFrameworkCore.CreatedUpdatedDate

What is a UUID (or GUID)?
https://pddivine.medium.com/what-is-a-uuid-or-guid-16d0ead25008

Guid struct
https://learn.microsoft.com/en-us/dotnet/api/system.guid?view=net-8.0