

**INGENIERÍA DE SOFTWARE**

**CURSO: CC65 Programación Concurrente y Distribuida**

**TRABAJO FINAL DEL CURSO**

|  |  |
| --- | --- |
| **PROFESOR:** | Luis Canaval Sánchez |
| **SECCIÓN:** | WX71 |
| **FECHA DE EVALUACIÓN:** | 25/11/2020 |

# 

# Objetivo:

El presente documento define el trabajo final y la rúbrica que permite evaluar el logro del curso CC65 Programación Concurrente y Distribuida

# Logro del curso:

Al finalizar el curso, el estudiante construye aplicaciones concurrentes y distribuidas de alto rendimiento de manera eficaz desde el punto de vista de la ingeniería de software.

**En Ingeniería de Software, el logro contribuye a alcanzar el:**

**EAC - ABET 5:** Trabajo en Equipos Multidisciplinarios: Funcionar eficazmente como miembro o líder de un equipo que participa en actividades apropiadas para la disciplina del programa.

En Ciencias de Computación, el logro contribuye a alcanzar el:

**EAC - ABET 5:** Trabajo en Equipos Multidisciplinarios: La capacidad de funcionar efectivamente en un equipo cuyos miembros juntos proporcionan liderazgo, crean un entorno de colaboración e inclusivo, establecen objetivos, planifican tareas y cumplen objetivos.

# Enunciado

# Deben elegir un problema donde se pueda aplicar uno de los algoritmos desarrollados durante la Tarea Académica 2 y desarrollar un API distribuido para poder acceder a dicho servicio.

# Tareas

1. El algoritmo de machine learning desarrollado para la Tarea 2 constituirá el core del backend a desarrollar.
2. El backend del API debe ser desarrollado íntegramente en GO.
3. El backend debe ser distribuido y funcionar en al menos 5 nodos.
4. El API debe permitir realizar consultas usando JSON para lo cual puede usar Postman como herramienta de prueba.
5. El proyecto se aloja en un Github o similar y debe seguir Git Flow.
6. El video debe demostrar la funcionalidad y tendrá una duración máxima de 2 minutos.
7. El informe debe incluir los siguientes puntos:
   1. Introducción, descripción del problema y motivación
   2. Objetivos
   3. Diseño (arquitectura o componentes, etc.)
   4. Desarrollo
   5. Conclusiones.

# Instrucciones para la entrega del trabajo

La presentación se realiza en la semana 15 durante clase y la entrega consta de un solo Hito a entregar al finalizar la semana 15. El entregable debe ser un solo archivo comprimido conteniendo lo siguiente:

1. Repositorio de código comprimido en zip.
2. Informe en formato pdf o md.
3. Link al repositorio público.
4. Link al video subido a un servicio como youtube como “no listado” o similar.

# 

# INFORME TRABAJO FINAL

**Introducción:**

Debido a la cantidad de personas que día a día pueden haber contraído el virus de Covid-19, es que nuestra plataforma se dio necesaria. En tiempos de pandemia se necesita tener una plataforma que brinde información, y que ofrezca la capacidad predictiva y analítica del estado para la toma de decisiones que en tiempos de pandemia son deficientes. Por lo tanto, nuestra plataforma SanFernando va a ofrecer información confiable y modelos de análisis modernos que nos permiten tener datos exactos y completos.

**Descripción del problema:**

El problema radica en la cantidad de información dispersa obtenida de los posibles contagios acarreados en esta época de pandemia. Debido a la cantidad de información de pacientes con esta enfermedad no se puede tener una visión clara de que medidas tomar para poder mitigar el crecimiento de posibles contagios, es por esto, que mediante el uso de nuestra plataforma se puede tener un análisis más exacto sobre la cantidad de afectados.

**Motivación:**

Nuestra motivación nace del querer conocer más sobre la evolución de esta enfermedad y qué medidas podríamos tomar para poder reducir la cantidad de afectados, además podríamos verificar cómo evoluciona la propagación de esta enfermedad para poder evitar futuros contagios.

**Objetivos:**

* Uno de nuestros principales objetivos es brindar un aplicativo que pueda predecir la cantidad de afectados en ciertas zonas
* Poder tener una visión más clara sobre la cantidad de pacientes afectados por esta pandemia, así como los lugares en los que se propaga.
* Brindar información relevante para poder tomar las medidas necesarias en contra de la propagación del virus Covid-19

**Diseño:**

Para el aplicativo propuesto se utilizó uno de los algoritmos de agrupamiento más usado, el cual es K-means. Este algoritmo de agrupamiento o clustering es un método de redes neuronales de aprendizaje no supervisado, que por medio de este método se logra dar sentido a los datos ingresados y los reúne en grupos similares. Este algoritmo de agrupamiento puede reconocer patrones y estructuras comunes de un conjunto de datos que una persona normalmente no podría detectar.

En términos generales, este algoritmo de agrupamiento puede dividir los datos ingresados en grupos con características en común, con lo cual lo hace una herramienta muy útil para poder reconocer patrones comunes en los datos ingresados y poder separarlos acorde a las necesidades.

Se trabajará usando la metodología Scrum y las historias de usuario serán realizadas en el aplicativo Trello.

Interfaz de usuario gráfica

Descripción generada automáticamente

**Desarrollo:**

Para el desarrollo de SanFernando, inicialmente se procedió a la identificación de los elementos del dominio y los conceptos descriptivos de sus propiedades, definiendo además las relaciones que se establecen entre ellos, en base a las cuales se organiza el conocimiento. Se determinó, básicamente, que el problema que se quiere resolver está basado en cómo contribuir con el proceso de seguimiento al paciente en observación por COVID-19.

**Conclusiones:**

En conclusión, por medio de nuestro aplicativo se puede hacer un seguimiento a los pacientes afectados por el Covid-19, con lo cual hace que la propagación de este virus se vea medrado. Además, nuestro aplicativo brinda una rápida solución mediante los datos procesados, logrando la toma de decisiones a un nivel más veloz.

**Anexos:**

Trello:

<https://trello.com/invite/b/NuR3S88l/bfbd04007f0603d006fe0ceefd125d17/sprintpcd>

Github:

<https://github.com/marcRosas/PCD_TF.git>

**Bibliografía:**