Pontificia Universidad Católica de Chile
Escuela de ingeniería
Departamento de ciencias de la computación
Profesor: Hernán Cabrera

Entrega Final

IIC2513 Tecnologías y aplicaciones WEB

Fecha límite de entrega: LUNES 27 de noviembre 23:30 hrs.

Recuerden que los enunciados sólo dan la línea general de la funcionalidad que deben implementar, pero sin entrar en mayores detalles ni puntos específicos de tal manera que ustedes *demuestren* su capacidad de trabajo en equipo y resolución de problemas.

Fecha límite de entrega, LUNES 27 de noviembre 23:30 hrs.

Indicaciones

El objetivo de esta entrega es **conectar** (si es que ya no lo han hecho) las dos partes de su desarrollo, es decir front con back-end. Eso significa formalizar la comunicación via HTTP entre ambos "extremos", para ello deberán definir sus APIS de conexión así como los end points correspondientes. Junto con ello, tendrán la oportunidad de ***arreglar, mejorar, enmendar y dejar a punto su juego***. Para ello acuerden bien con sus ayudantes los criterios de mejora que deberán incorporar.

Sabemos que no siempre se puede llegar con todo terminado y que hay elementos que han faltado, ya sea por tiempo, dificultad, carga académica o infortunio. Pues esta es la oportunidad de mejorar todos los aspectos pendientes, que han sido señalados por los ayudantes, para terminar su trabajo, de forma integral, entregando un juego funcional.

En esta entrega es indispensable y **obligatorio** que entreguen:

1. **Aspectos mínimos de las entregas anteriores, considerando:**
    a. Un nivel de juego mínimo que incluya reglas mínimas que les permita jugar.
    b. Manejo de usuarios y de inicio de sesión.
    c. Comunicación entre un cliente y un servidor.
2. **Documentación** final actualizada de la aplicación:
    a. Modelo **Entidad-Relación**
    b. Documentación de la **API**
    c. Readme con descripción de las **herramientas usadas** y todos los **pasos para ejecutar** la aplicación
3. Deploy del front de la aplicación en Netlify o similar
4. Deploy del backend de la aplicación en Heroku, Render o similar
5. Uso obligatorio de linter (Eslint es el recomendado) en front y back

6. Convenciones JavaScript, recordar la referencia citada en este curso de estilo en Aribnb https://github.com/airbnb/javascript (camelCase y otras buenas prácticas del lenguaje, respeto sintaxis *linter*)
7. Criterios de calidad del código:

Ítems mínimos (**deben cumplirse todos**):

1. No usar Inline CSS
2. No usar html para estilo ni posicionamiento (sin tablas)
3. Uso apropiado de React
4. **Construir las rutas o URL de links usando ctx.router.url o similar (en lugar de hardcoded)**
5. Validaciones bien definidas en los modelos
6. Control de errores (más comunes)

Ítems importantes que se tendrán en cuenta:

1. Sacar provecho de middlewares
2. **Sin funciones gigantes en los route helpers** ("fat models / skinny controllers")
3. Usar métodos donde corresponda (MVC)

Si un grupo ya tiene todas las funcionalidades pedidas con anterioridad, para el 7 (banda A) se le puede pedir una de las siguientes funcionalidades (siempre y cuando no estén ya implementadas en las entregas anteriores):

1. Capacidad de multipartida. Es decir, varios jugadores interactúan a su vez en varias partidas administradas simultáneamente por el servidor. Además que cada jugador pueda estar participando de más de una partida activa a la vez.
2. Lograr que se guarden todas las partidas del juego y que un usuario pueda ver el historial de sus juegos, tanto del actual como de partidas anteriores para el mismo usuario.
3. Permitir animar en el tablero una partida terminada desde su estado inicial, es decir las jugadas propias y la del resto de los jugadores, paso a paso mostrando cómo se han movido sus piezas, logros, etc.
4. Entregar un panel de control (dashboard) que muestre estadísticas de cada jugador y del resto de los jugadores. Las estadísticas pueden entregar un ranking relativo, cantidad de territorios, poder de sus jugadores, estado de sus héroes, etc. todo dependiendo del tipo de juego que han implementado.
5. Poder elegir el tipo de tablero en el que se desarrollará el juego. Para esto puede disponer de una cantidad de tableros que se puedan seleccionar y luego crear y ejecutar el juego en dicha configuración.

La entrega de todos los archivos se hará en el repositorio de Github creado para su grupo.

NO se aceptarán:

- Entregas por mail (ya sea al profesor o ayudantes)
- Entrega en otro sistema que no sea el que se ha provisto para estos efectos (los repositorios de Github entregados por el coordinador y deploy con Heroku, Render y Netlify o equivalente)

Reemplazo de nota de exámen

Si lo considera conveniente, la nota de esta entrega puede ser usada también como nota de su exámen.
Si no quiere usar la nota de esta entrega como nota de exámen deberá inscribirse para el examen y deberá darlo de forma obligatoria. En este caso, la nota que saque en el examen no será reemplazada.

Salón de la fama

Si un grupo termina el proyecto de forma destacable bajo algunos de estos criterios: juego creativo, completamente funcional, interfaz atractiva, … , optará por pasar al salón de la fama y terminar el proyecto con nota 7. Para poder ser candidato, se debe hacer el deploy de su aplicación (los ayudantes lo probarán y votarán).

Condiciones y restricciones

Aplican todas las restricciones generales entregadas en las entregas anteriores y se pueden sumar y agregar las acordadas con el ayudante.

Recuerden que se mirará en detalle el uso de Github por parte de TODOS los integrantes del grupo. Un integrante que no tenga actividad en github (commits) podrá ser penalizado.

Recomendaciones

1. Si el protocolo es complejo o tienen aspectos de modelo de datos o de usabilidad que los complican, no teman de realizar cambios y ajustes necesarios para entregar con éxito su tarea. Redefinir o rediseñar no está mal en la medida que cumplan con los requerimientos establecidos en las entregas.

2. Implementen todos los puntos de “contacto” (“endpoints”) que requieran y todas las APIs necesarias, no tiene que ser una sola y recomendamos que sean más atomizadas.

3. Aprenderán mucho más si trabajan colaborativamente en su grupo, como equipo en lugar de repartirse el trabajo y realizarlo como unidades independientes. ***RECUERDEN, el integrante que no tenga movimientos en gitHub, será penalizado en su calificación***

4. Si hay problemas con su compañero(a) y no lo pueden resolver, comuníquese con el profesor o con el coordinador.

5. Planifiquen el trabajo para que les permita la colaboración entre los integrantes del equipo

6. Pregunten y consulten, usen foro, colaboren entre ustedes (**NO COPIEN**). Los ayudantes están para apoyarlos

7. **Trabajen con tiempo, no esperen a último momento para comenzar con el desarrollo.**

8. Siempre podrán, justificadamente, cambiar alguna regla, mejorar algún aspecto del juego, etc.

Dudas

Para que todo el curso se vea beneficiado, hagan sus preguntas sobre el material del curso, sobre tecnologías web, y sobre el proyecto a través del **foro del curso** dispuesto para estos efectos. No se responderá ninguna duda de tareas por e-mail.