Prueba técnica: Aplicación de productividad

Autor: Santiago Sinisterra

Descripción

Desarrollar una aplicación de productividad en la que los usuarios puedan gestionar y filtrar una lista de tareas, así como registrar el tiempo que les toma ejecutarlas. Para esto existe un temporizador, el cual se inicializa con la duración de la tarea en cuestión y cuenta hacia cero, desplegando en todo momento el tiempo restante. El usuario puede ver un histórico de las tareas que ha completado, así como una gráfica que represente su productividad en la última semana.

Requerimientos y reglas de negocio

Requerimientos del Usuario

El usuario puede realizar las siguientes acciones:

- Crear una tarea
- Modificar una tarea (descripción de la tarea y duración)
- Eliminar una tarea
- Reordenar la lista de tareas
- Comenzar con la tarea en curso
- Pausar, detener o reiniciar el temporizador
- Marcar la tarea en curso como finalizada
- Ver el historial de tareas completadas

- Ver una gráfica de su historial de tareas en la última semana
- Filtrar la lista de tareas pendientes según su duración:
    - corto 30 min o menos
    - medio: de 30 min a 1h
    - largo: más de 1h

Reglas de negocio

- La tarea en curso es la tarea que está al inicio de la lista.
- Al marcar la tarea en curso como completada, debe registrarse el tiempo que tomó al usuario completar dicha tarea.
- Al cerrar la aplicación, el temporizador siempre se pausa.
- Se manejan tres duraciones predeterminadas para una tarea (corta: 30 min, media: 45 min, larga: 1h)
- El usuario también puede definir su propia duración en minutos y segundos para una tarea, la cual no puede superar las dos horas.
- Al expirar el tiempo de la tarea en curso, ésta se marca como completada

Especificaciones técnicas

Interfaz de usuario

- Los componentes de la interfaz deben ser desarrollados en React.
- Pueden utilizarse componentes de React que integren el framework de UI de preferencia (material-ui, react-mdl, react-bootstrap), así como para armar la gráfica (react-d3, victory).
- Se recomienda los componentes se estructuren entre contenedores y presentacionales (stateful vs stateless)
- Cuidar la responsividad de la interfaz en múltiples dispositivos

Aplicación

- Los componentes deberían utilizar react hooks.
- Debe existir una función para prellenar la aplicación con 50 tareas aleatorias, completadas consumiendo entre el 80% y el 100% de su duración, distribuidas en la última semana

Código

- Desarrollar el código utilizando la especificación más reciente de JavaScript (ES6)
- Incluir suficientes comentarios en el código
- Manejar control de versiones

Opcionales

- Persistir el estado de la aplicación a través de un API
- Desplegar la aplicación en un servicio PaaS como Heroku
- Incluir pruebas unitarias para la manipulación del estado de la aplicación