Desarrollo Web y Mobile - Proyecto de curso

Este es el planteo del proyecto de curso para el curso de *Desarrollo Web y Mobile* para el segundo semestre de 2025. Este documento está sujeto a cambios y correcciones.

Objetivo:

Desarrollar una aplicación web y mobile que permita definir y jugar pencas a torneos varios.

Requerimientos funcionales

La aplicación completa cuenta con tres roles de *usuarios*: *Gambler*, *Manager* y *Admins*.

Historias para todos los roles

- Como *Usuario* ingreso a la aplicación con mi correo electrónico y contraseña para poder usarla.
- Como *Usuario* termino mi sesión antes de cerrar el navegador o la aplicación móvil.

Historias para *Gamblers*

- Como *Gambler* veo los torneos a los que he sido invitado.
- Como *Gambler* veo los partidos dentro de un torneo al que tengo acceso, pasados y futuros.
- Como *Gambler* veo los pronósticos que he hecho en partidos pasados y futuros.
- Como *Gambler* hago o modifico pronósticos sobre partidos futuros en torneos a los que tengo acceso.
- Como *Gambler* veo los pronósticos que han hecho otros usuarios en partidos pasados de torneos a los que tengo acceso.
- Como *Gambler* veo los resultados de los partidos pasados.
- Como *Gambler* veo mi puntaje en un torneo, completo o no.
- Como *Gambler* veo el puntaje de otro usuario en un torneo al que tengo acceso.

Historias para *Managers*

- Como *Manager* defino torneos con nombre, descripción y rango de fechas en que se juegan.
- Como *Manager* defino equipos con nombre, descripción y logotipo (imagen).
- Como *Manager* defino partidos de un torneo, con nombre, equipos y fecha.
- Como *Manager* invito usuarios como seguidores de un torneo.
- Como *Manager* cargo los resultados de un partido, luego de su fecha.
- Como *Manager* puedo hacer todo lo que hace un *Gambler*.

Historias para *Admins*

- Como *Admin* agrego usuarios con su correo electrónico, contraseña y rol; así pueden ingresar a la aplicación.
- Como *Admin* modifico un usuario existente para cambiarle su contraseña o rol.
- Como *Admin* quito usuarios de la aplicación.
- Como *Admin* puedo hacer todo lo que hace un *Manager*.

Requerimientos técnicos

La aplicación tendrá dos *frontends* implementados con Javascript: el *web* con *React*, y el *móvil* con *React Native* (para celulares Android). Si bien se pide una aplicación móvil, la interfaz web debe ser responsiva para soportar pantallas de dispositivos celulares.

Ambas interfaces de usuario (i.e. web y móvil) se comunicarán con una *API REST* provista por la cátedra. Se deberá utilizar JWT para la autenticación de usuarios.

Criterios de Evaluación:

La evaluación del código entregado por los equipos se concentrará en:

- Implementación de las historias de usuario solicitadas.
- Código limpio y bien estructurado, con buenas prácticas de desarrollo.
- Diseño y usabilidad de las interfaces de usuario.

Entregables:

- Repositorio de GitHub con el código fuente de la aplicación web y móvil.
- Documentación del proyecto que incluya:
  - Instrucciones para ejecutar la aplicación.
  - Explicación de las funcionalidades implementadas.
  - Justificación de las decisiones técnicas tomadas.

Plazo de Entrega:

La entrega del proyecto será el 1 de diciembre de 2025. Ese mismo día se estará tomando la defensa del proyecto.

Anexo 1: Data Model

```
interface Entity {
  id: string;
  dateCreated: Date;
  dateModified: null | Date;
}

interface User extends Entity {
  email: string;
  password: string;
  roles: 'gambler' | 'manager' | 'admin';
}

interface Tournament extends Entity {
  name: string;
  description: string;
  beginning: Date;
  ending: Date;
  matches: Match['id'][];
}

interface Match extends Entity {
  title: string;
  date: Date;
  tournament: Tournament['id'];
  teams: Team['id'][];
  score: Record<Team['id'], number> | null;
}

interface Team extends Entity {
  title: string;
  date: Date;
}

interface Invitation extends Entity {
  invitedGambler: User['id'];
  invitingManager: User['id'];
  tournament: Tournament['id'];
  acceptedAt: Date | null;
  revokedAt: Date | null;
}

interface Gamble extends Entity {
  user: User['id'];
  match: Match['id'];
  score: Record<Team['id'], number>;
}
```

Anexo 2: API REST

Tanto los cuerpos de peticiones como los de las respuestas usarán JSON como formato. Todas las rutas excepto `/api/login`, requieren el token de sesión en el *header* `Authorization` (de la forma `Bearer <token>`).

Todos los usuarios

- `POST /api/login`: Ingreso de un usuario a la aplicación, retornando un *token* de sesión.
  - Cuerpo `{ email: string, password: string }`
  - Respuesta OK 200 `{ token: string }`
- `POST /api/logout`: Despedida de un usuario, dando de baja su sesión en la aplicación.
  - Cuerpo `{}`
  - Respuesta OK 204 No Content
- `GET /api/me`: Retorna la información del usuario actual de la aplicación.
  - Respuesta OK 200 `{ username: string }`

Gamblers

- `GET /api/me/tournaments/`: Listar torneos a los que el usuario actual tiene acceso.
  - Respuesta OK 200 `Tournament[]`
- `GET /api/me/tournaments/:id/matches`: Listar partidos de un torneo al que el usuario actual tiene acceso.
  - Respuesta OK 200 `Match[]`
- `GET /api/me/tournaments/:id/gambles`: Listar pronósticos realizados en un torneo.
  - Respuesta OK 200 `Gamble[]`
- `GET /api/me/gambles`: Listar todos los pronósticos realizados por el usuario actual.
  - Respuesta OK 200 `Gamble[]`
- `POST /api/me/gambles`: Crear o actualizar un pronóstico.
  - Cuerpo `{ match: Match['id'], score: Record<Team['id'], number> }`
  - Respuesta OK 200 `Gamble`
- `DELETE /api/me/gambles/:id`: Eliminar un pronóstico.
  - Respuesta OK 204 No Content
- `GET /api/me/invitations`: Listar invitaciones pendientes del usuario actual.
  - Respuesta OK 200 `Invitation[]`
- `POST /api/me/invitations/:id/accept`: Aceptar una invitación a un torneo.
  - Respuesta OK 200 `Invitation`
- `POST /api/me/invitations/:id/reject`: Rechazar una invitación a un torneo.
  - Respuesta OK 200 `Invitation`

Managers

- `GET /api/tournaments`: Listar todos los torneos.
  - Respuesta OK 200 `Tournament[]`
- `POST /api/tournaments`: Crear un torneo.
  - Cuerpo `{ name: string, description: string, beginning: Date, ending: Date }`
  - Respuesta OK 201 `Tournament`
- `DELETE /api/tournaments/:id`: Eliminar un torneo.
  - Respuesta OK 204 No Content
- `GET /api/teams`: Listar todos los equipos.
  - Respuesta OK 200 `Team[]`
- `POST /api/teams`: Crear un equipo.
  - Cuerpo `{ title: string, description: string, logoUrl: string }`
  - Respuesta OK 201 `Team`
- `DELETE /api/teams/:id`: Eliminar un equipo.
  - Respuesta OK 204 No Content
- `GET /api/matches`: Listar todos los partidos.
  - Respuesta OK 200 `Match[]`

- `POST /api/matches`: Crear un partido.
  - Cuerpo `{ title: string, date: Date, tournament: Tournament['id'], teams: Team['id'][] }`
  - Respuesta OK 201 `Match`
- `DELETE /api/matches/:id`: Eliminar un partido.
  - Respuesta OK 204 No Content
- `GET /api/invitations`: Listar invitaciones de un torneo.
  - Respuesta OK 200 `Invitation[]`
- `POST /api/invitations`: Invitar un usuario a un torneo.
  - Cuerpo `{ invitedGamblerEmail: string }`
  - Respuesta OK 201 `Invitation`
- `DELETE /api/invitations/:id`: Revocar una invitación.
  - Respuesta OK 204 No Content
- `POST /api/invitations/:id/revoke`: Revocar una invitación.
  - Cuerpo `{}`
  - Respuesta OK 200 `Invitation`
- `POST /api/matches/:id/result`: Cargar el resultado de un partido.
  - Cuerpo `{ score: Record<Team['id'], number> }`
  - Respuesta OK 200 `Match`
- `DELETE /api/matches/:id/result`: Eliminar el resultado de un partido.
  - Respuesta OK 204 No Content

Admins

- `GET /api/users`: Listar todos los usuarios.
  - Respuesta OK 200 `User[]`
- `POST /api/users`: Crear un usuario.
  - Cuerpo `{ email: string, password: string, roles: 'gambler' | 'manager' | 'admin' }`
  - Respuesta OK 201 `User`
- `DELETE /api/users/:id`: Eliminar un usuario.
  - Respuesta OK 204 No Content

Error comunes

- 400: Bad Request `{ error: string }`
- 401: Unauthorized
- 403: Forbidden
- 404: Not Found
- 500: Internal Server Error `{ error: string }`

Fin