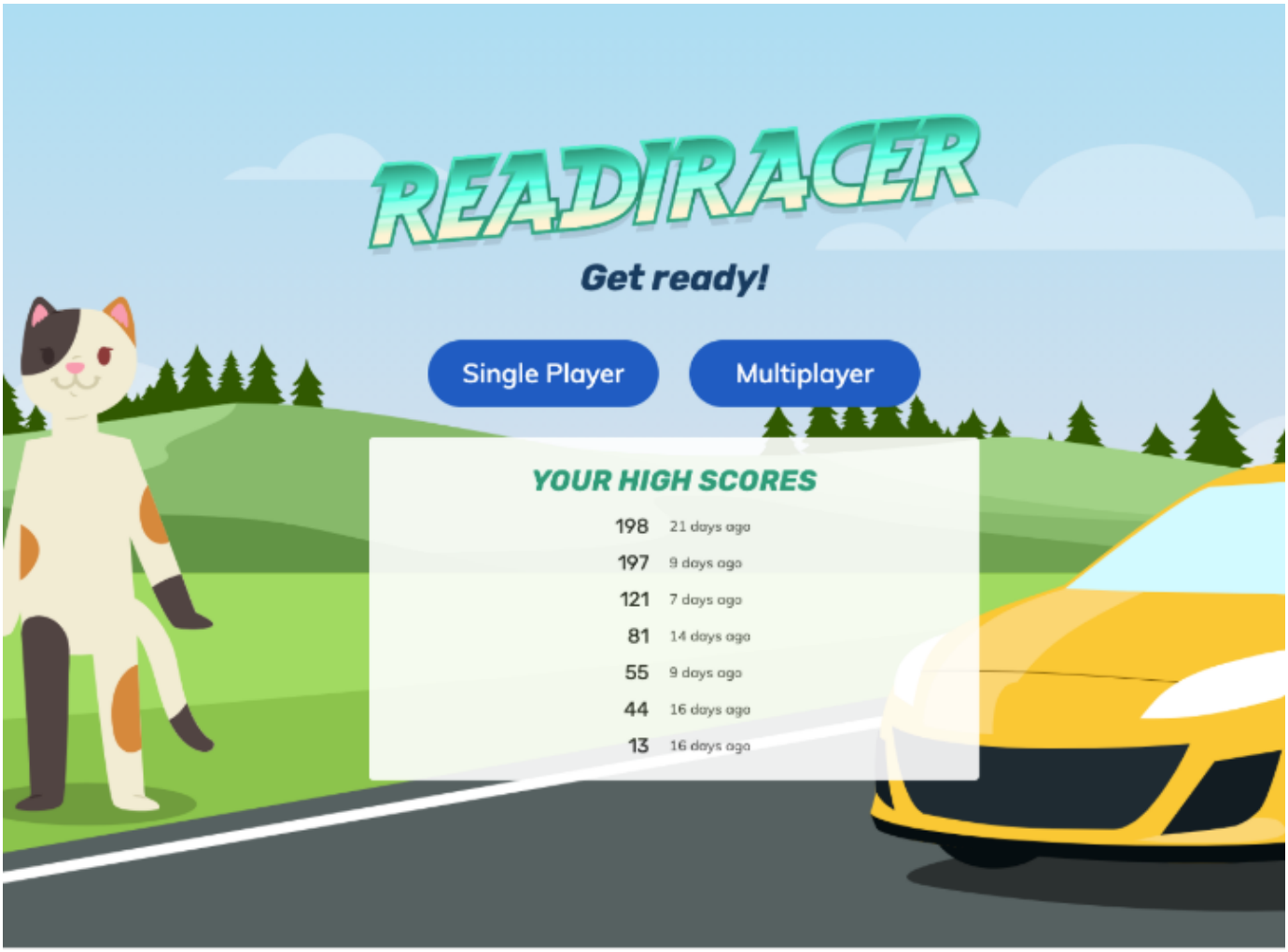
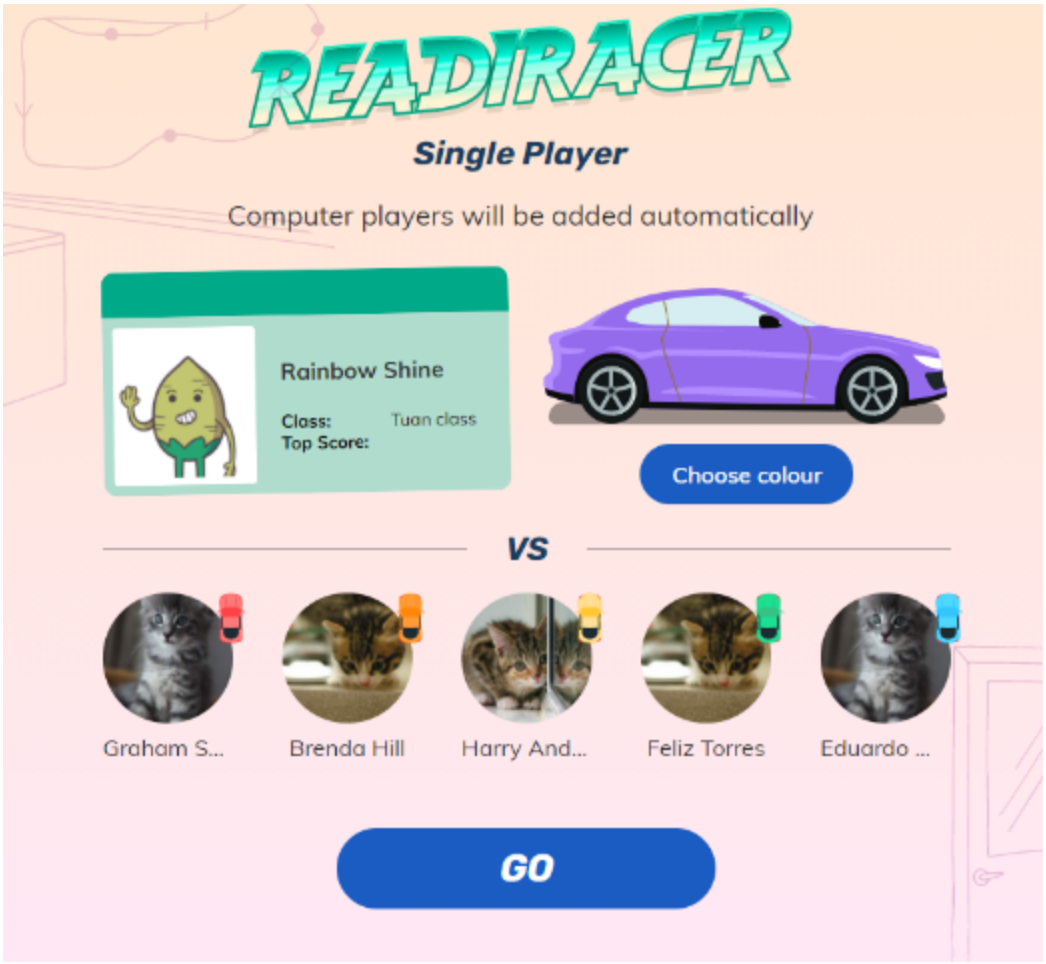
Technical design for ReadiRacer game

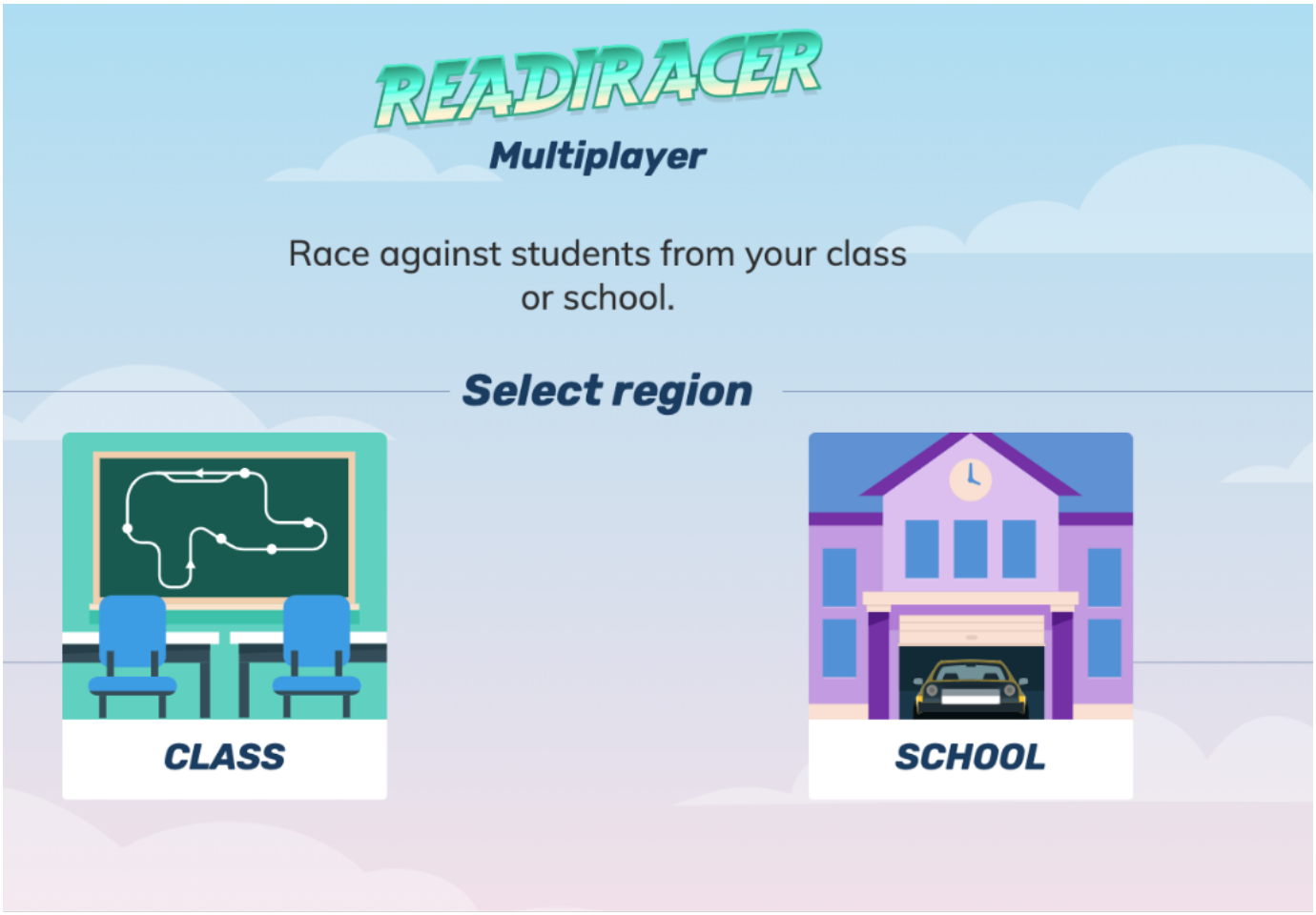
Landing Page UI



Single-Player Lobby UI

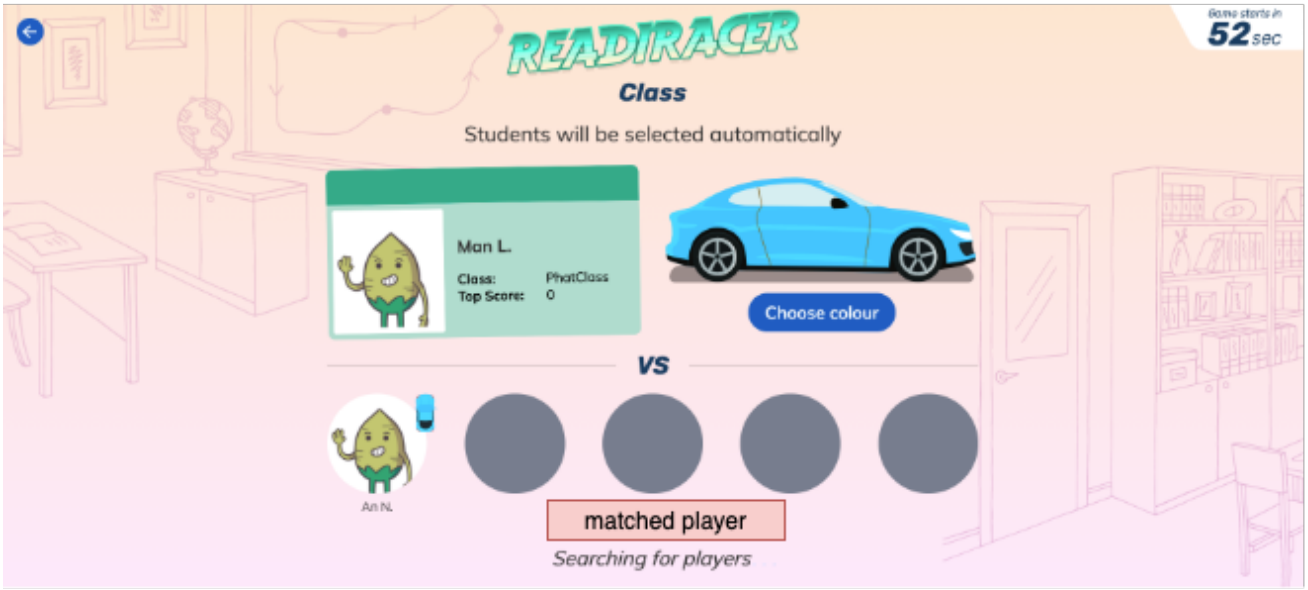


Multi-Player Region UI



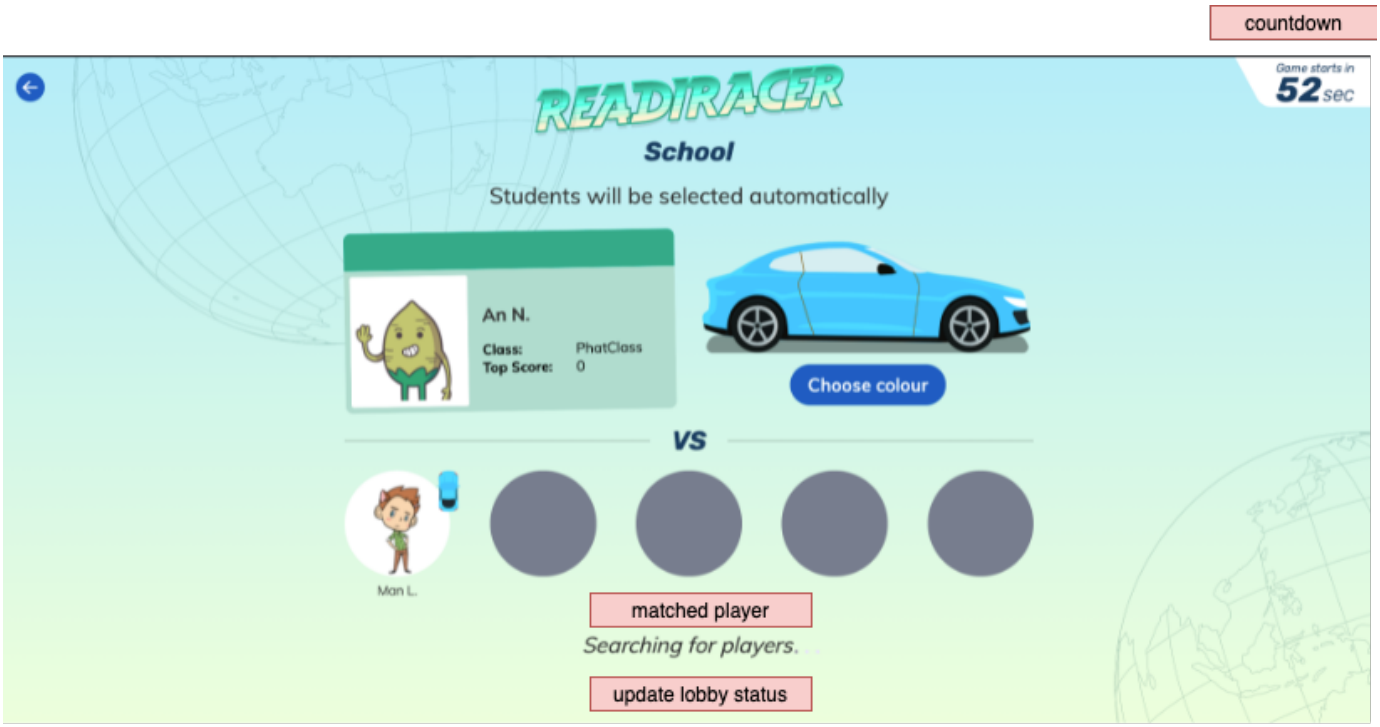
Multi-Player Class Lobby UI

countdown



5. Update lobby status

Multi-Player School Lobby UI



Gateway	Verb	Endpoint	Description	Swagger
1. Get CAR resource or lobby resource in general				
SpellingSync	POST	lobbies/{lobbyId}/resources	User selects a car color.	• QA Swagger
SpellingSync	PUT	lobbies/{lobbyId}/resources	Re-select another color. **OnlinePlayer can select the color that has already been chosen by other GhostPlayer. In that case, the GhostPlayer will switch to the old color that the OnlinePlayer was using.**	• QA Swagger
SpellingSync	GET	lobbies/{lobbyId}/resources	Get lobby's resources. Optional query params (will returns all available values if not used): • array[string]: resourceTypes = CAR, SLOT,… • int: userProfileId	• QA Swagger
2. Get lobby SLOT resource				
SpellingSync	GET	lobbies/{lobbyId}/resources/SLOT	Get SLOT resource with avatar and fullName. Optional query params: • int: userProfileId	• QA Swagger
3. Get students and invitations				
SpellingSync	POST	lobbies/{lobbyId}/invitations	Send an invitation to another user. **The invitation will automatically accepted if the player is invited as GhostPlayer, otherwise FE will need to wait for a response (via notification).**	• QA Swagger

SpellingSync	PUT	lobbies/{lobbyId}/invitations/{invitationId}	Accept/deny an invitation. `public enum InvitationStatus { Pending = 1, Accepted = 2, Denied = 3, Expired = 4, }`	
SpellingSync	GET	lobbies/{lobbyId}/students	Get all students in the same classroom including status, name and **invitations**.	• QA Swagger
4. Get user info, saved preference				
Spelling	GET	/profile/activity-types/{activityType}	Get user's saved preference for an activity type. For Readiracer activity type is 11.	• QA Swagger
Spelling	PUT	/profile/activity-types/{activityType}	Save user preference for an activity type. For Readiracer activity type is 11.	• QA Swagger
Spelling	GET	/users/profile	Get user's profile data (name, highest score, class,…) Optional query params: • int: classId (leave as empty to return all the classrooms that the user is currently in)	• QA Swagger
5. Create & update lobby				

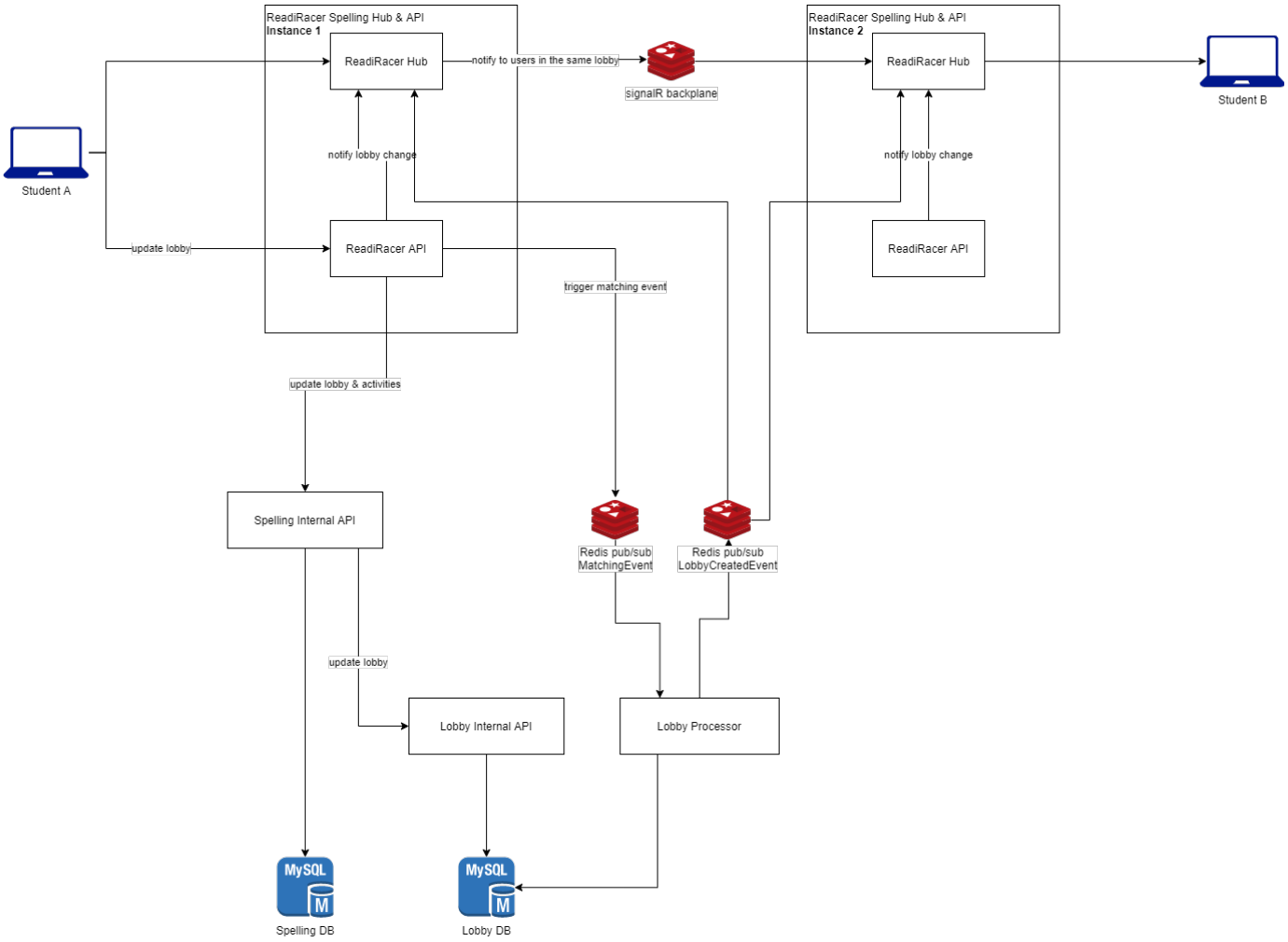
SpellingSync	POST	lobbies/	Create new lobby room `public enum LobbyType { /// <summary> /// Lobby for students in the same class. /// </summary> Class = 1, /// <summary> /// Lobby for students in the same school. /// </summary> AutoMatchingSchool = 2, /// <summary> /// Lobby for all students around the world. /// </summary> AutoMatchingWorld = 3, CustomMode = 4 }`	• QA Swagger
SpellingSync	GET	lobbies/{lobbyId}	Get lobby data	• QA Swagger

SpellingSync	PATCH	lobbies/{lobbyId}	Update lobby status `public enum LobbyStatus { Created = 1, Matching = 2, Progressing = 3, Completed = 4, Canceled = 5 }` **Change status to Progress will trigger an create activities of all users in the lobby.**	• QA Swagger

Playing game UI

Gateway	Single player Endpoint	Gateway	Multi players Endpoint	Description
1. **Start & progress game**				
Spelling	POST /classes/{classId}/activity-types/{activityType}/activities/data	SpellingSync	PATCH /lobbies/{lobbyId}	Click GO to start game, update lobby/activity status. • QA Swagger • https://mathletics.atlassian.net/browse/LIT-4633 • https://mathletics.atlassian.net/browse/LIT-4734
Spelling	GET /activities/{activityId}	SpellingSync	GET /lobbies/{lobbyId}/activities	Get activity data. • QA Swagger
Spelling	PUT /activities/{activityId}/ai-players	SpellingSync	GET /lobbies/{lobbyId}/players	Get other player's pre-generated steps. • QA Swagger • https://mathletics.atlassian.net/browse/LIT-4289
Spelling	POST /activities/{activityId}/attempts	SpellingSync	POST /lobbies/{lobbyId}/activities/{activityId}/attempts	Submit answer. • QA Swagger • https://mathletics.atlassian.net/browse/LIT-4712
Spelling	PUT /activities/{activityId}	SpellingSync	PATCH /lobbies/{lobbyId}/activities/{activityId}	Progresses an activity (finish, game over,…). • QA Swagger

Component Diagram



The communications among components (FE Client, Hub, API, and Processor), assume that Student A & B are in the same lobby room or will have a lobby room created for 2 students

- Student A sends an HTTP request that can trigger a Web socket notification to student B
  - Student A ReadiRacer API
  - ReadiRacer API Spelling API and/or Lobby API to update data
  - ReadiRacer API ReadiRacer Hub instance 1
  - ReadiRacer Hub instance 1 ReadiRacer hub instance 2 via Redis backplane
  - ReadiRacer hub instance 2 Student B
- Spelling Hub & Processor communicate via **Redis pub/sub**
  - Student A registers for auto-matching via WS and an event will be triggered to Processor
  - The Processor creates a new lobby room and sends another event to hubs to notify connected clients

What data is communicated between FE and BE ?

- We send object-based data. For example Lobby, Lobby resource, lobby activity
- Pros
  - Less events to process
  - Does not care what is updated, just make sure to track the modified date
- Cons
  - Send a lot more data. For example:
    - instead of sending LobbyStatusUpdated with status info
    - we send the whole object
    - FE need to check what's changed using react-hook, which is extremely easy
- We send the same data two times using signalR and polling service independently