Assignment 2: Use Cases

Project: flotE

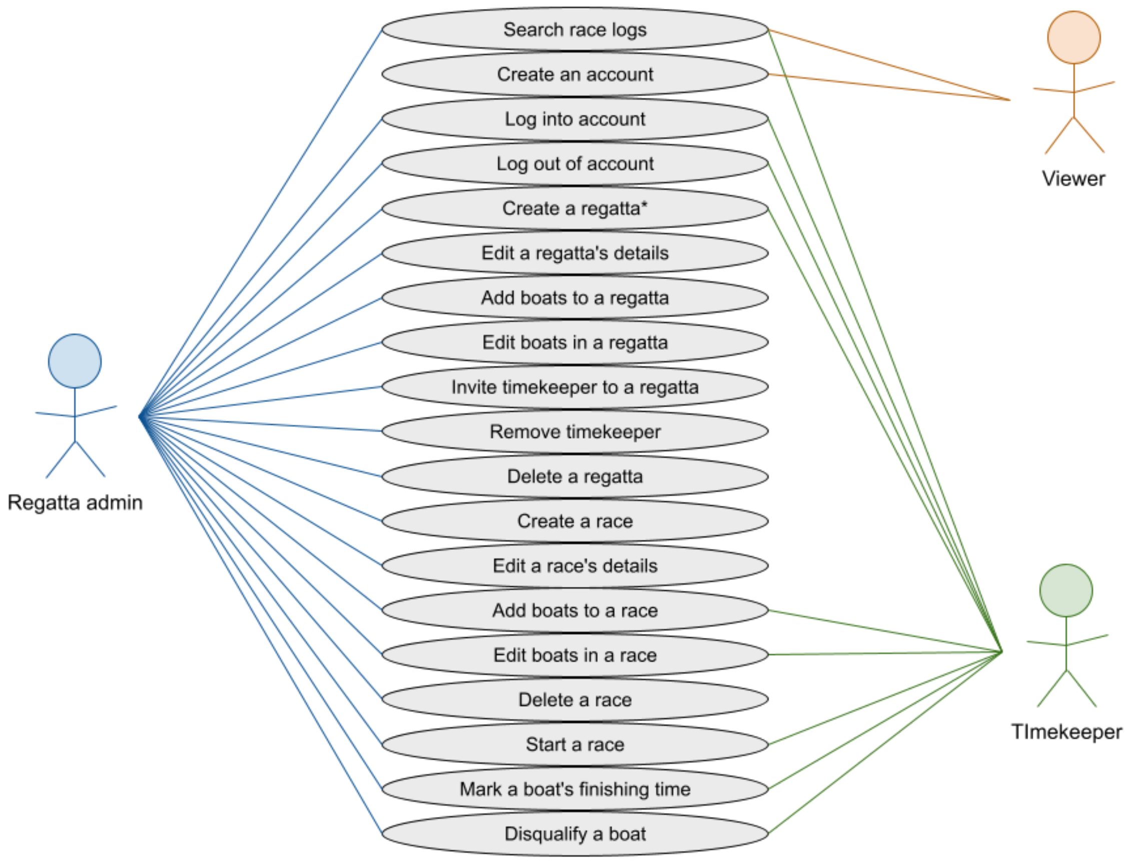
Group 1: Dila Ozersen, Macy Graves, Ashley Bhandari, Austin Henlotter, Jamie Denley, Anushka Trehan (Manager)

Project description: The current method for tracking finishing times in a boat race presents two problems: (1) two people are required for timekeeping, one for keeping time and the other for recording results; and (2) the finish line is not clearly visible from shore, resulting in a gap between when a boat finishes and when viewers receive its finishing time. Our clients proposed flotE to solve these problems. The application has two main features: administration and timekeeping. flotE helps users organize and save regattas, races, and participants, in addition to providing a way to search these stored records. The application also provides timekeeping capabilities: a built-in stopwatch and boat suggestions allow a single user to timekeep a race on their own; and as results are recorded, they are posted to the application in real time (via SMS) for viewers to see.

Use Case Diagram

We have defined three stakeholders: Viewers, regatta admins, and timekeepers.

1. A viewer is an unregistered user. They are allowed to view races and search the application's logs for regattas, races, and participants. They also have the option to create an account if they wish to become a regatta admin or timekeeper.
2. A regatta admin is a registered user who has created a regatta. They are in control of everything about their regatta—details, registered boats, races, and timekeeping.
3. A timekeeper is a registered user who has been invited by a regatta admin to timekeep for that regatta's races. They are allowed to timekeep and change small details about the races they timekeep.



*When a timekeeper creates a regatta, they became a regatta admin for the newly created regatta

Use Cases

Search for regattas, races, and participants	
Description	Search logs by regatta, race, participant, or boat ID. Results are organized into regatta, race, and participant categories. Clicking a result directs the user to the relevant page.
Actors	Primary: All stakeholders (regatta admins, timekeepers, and viewers)
	Secondary: DB
Goal	Search logs for current and past regattas, races, and participants
Preconditions	User is on the landing page
Postconditions	Success: List of matching results is displayed
	Failure: The search fails to execute
Exceptions	DB error
Trigger	User clicks the search bar
Main success scenario	1. User inputs their search query (regatta name / race name / participant name / boat ID) and clicks the "Search" or enter button 2. Application queries DB to find matching results 3. A list of results is displayed
Error scenarios	2a. DB query fails due to an internal error. A "something went wrong" error message is displayed.
Alternative scenarios	3. No matching results. A message indicating this is displayed.

Create a new account	
Description	A new user should be able to create an account so that they can create regattas/races and timekeep.
Actors	Primary: Unregistered user
	Secondary: DB
Goal	Create a new account and log the user in
Preconditions	User logged out and on the landing page
Postconditions	Success: User creates an account
	Failure: Application fails to create the account
Exceptions	Invalid email, invalid password, existing account, internal error
Trigger	User clicks the "Create an account" button
Main success scenario	1. User enters their email and password 2. Application saves credentials to the DB 3. Application confirms successful account creation 4. User is logged in and directed into the application
Error scenarios	1a. Email is invalid. An error message indicating this is displayed. 1b. Password is invalid (weak, contains spaces, etc.). An error message describing password requirements is displayed. 2a. A user with the inputted email already exists in the DB. An error message indicating this is displayed. 2b. Account cannot be created due to an internal error. A "something went wrong" error is displayed.

Log an existing user in	
Description	Log an existing user into their account
Actors	Primary: Existing user
	Secondary: DB
Goal	Log the user in
Preconditions	User is logged out and on the landing page
Postconditions	Success: User is logged in and directed to the application
	Failure: Login attempt fails
Exceptions	Email does not exist in the DB, password is incorrect, internal error
Trigger	User clicks “Login” button
Main success scenario	1. User enters their email and password 2. Application verifies credentials and saves an auth token to local storage 3. Application confirms successful login 4. User is directed into the application
Error scenarios	2a. Email does not exist in the DB. An "account with that email or password does not exist" error message is displayed. 2b. Password is incorrect. An "account with that email or password does not exist" error message is displayed. 2c. User cannot be logged in due to an internal error. A "something went wrong" error message is displayed.

Log a user out of their account	
Description	Log a user out of their account
Actors	Primary: Existing user
Goal	Log the user out
Preconditions	User is logged in
Postconditions	Success: User is logged out and directed to the landing page
	Failure: User cannot be logged out
Exceptions	Logout fails, redirect fails
Trigger	User clicks “Log out” button
Main success scenario	1. Auth token is removed from local storage 2. User is directed to the landing page
Error scenarios	1a. Logout fails. Error message indicating this is displayed. 2a. Redirect fails. User stays on same page.
Alternative scenarios	1. Auth token expires and the user is automatically logged out

Create a regatta	
Description	Create a new regatta with the current user as its admin, to which boats and races can later be added
Actors	Primary: Any registered user
	Secondary: DB
Goal	Create a new regatta and add it to the user's "regatta list"
Preconditions	User is logged in and on the home page
Postconditions	Success: A regatta is created with the user as its admin
	Failure: Application fails to create the regatta
Exceptions	Email does not have an associated account, DB error
Trigger	User presses the “Create a regatta” button
Main success scenario	1. User inputs the regatta name 2. User invites other users to be timekeepers by inputting their emails 3. User clicks the "Save" button 4. Application saves regatta and assigns it to user in the DB 5. Application confirms successful regatta creation and creates a page for the regatta 6. User is directed to the regatta's page
Error scenarios	2a. The inputted email does not have an associated account. An error message indicating this is displayed. 4a. The regatta cannot be saved due to an internal error. A "something went wrong" error message is displayed.

Edit a regatta's details	
Description	A regatta's admin can edit the regatta's name and date(s)
Actors	Primary: Regatta admin
	Secondary: DB
Goal	Change the regatta's name or date(s)
Preconditions	User is logged in, has created a regatta, and is on that regatta's page
Postconditions	Success: The regatta is successfully edited
	Failure: Application fails to edit the regatta information
Exceptions	DB error
Trigger	User clicks the "Edit" button
Main success scenario	1. User inputs the new name and/or date(s) into the provided field 2. User presses "Save" button 3. Application saves the information to the DB 4. Application confirms successful save
Error scenarios	3a. The information is not saved to the DB. A "something went wrong" error message is displayed.
Alternative scenarios	2. User clicks the "Cancel" button. Edits are not saved.

Add boats to a regatta	
Description	A regatta's admin can register a new boats to a regatta
Actors	Primary: Regatta admin
	Secondary: DB
Goal	User adds new boats to the regatta
Preconditions	User is logged in, is a regatta admin, and is on the regatta's page
Postconditions	Success: The user adds new boats to the regatta
	Failure: The user is unable to add boats to the regatta
Exceptions	Boat is already registered, DB error
Trigger	User clicks the “Edit” button
Main success scenario	1. For each boat to be added, user adds its ID and its participants' names to the list 2. User clicks the "Save" button 3. Application saves the boats to the DB 4. Application confirms success and the boats are added to the regatta's list of registered boats
Error scenarios	1a. An inputted boat ID is already registered with the regatta. An error message indicating this is displayed. 3a. The new boat is not saved. A "something went wrong" error is displayed.

Edit boats in a regatta	
Description	A regatta's admin can edit the list of registered boats for a regatta; they can change any boat's ID and associated participants, as well as delete it
Actors	Primary: Regatta admin
	Secondary: DB
Goal	User edits the regatta's list of registered boats
Preconditions	User is logged in, is a regatta admin, and is on the regatta's page
Postconditions	Success: The specified boats are edited or removed
	Failure: Application fails to upload the list
Exceptions	Boat is already registered, DB error
Trigger	User clicks the “Edit” button
Main success scenario	1. For each boat the user wishes to update, they select the "Edit boat" button and input the desired ID and/or participants 2. User clicks the "Save" button 3. Application updates the boat in the DB 4. Application confirms successful update
Error scenarios	1a. The new ID is already associated with another boat in the regatta. An error message indicating this is displayed. 3a. The new boat is not saved. A "something went wrong" error is displayed.
Alternative scenarios	2. User clicks the "Delete boat" button. A popup confirmation is displayed, and the user can click either "Cancel" or "Confirm."

Invite a timekeeper to a regatta	
Description	A regatta's admin can invite other users by email to act as a timekeeper for races in the regatta
Actors	Primary: Regatta admin
	Secondary: DB
Goal	Invite another user to timekeep for a regatta
Preconditions	User is logged in, has created a regatta, and is on that regatta's page
Postconditions	Success: Timekeepers are given access to timekeeping features for the regatta's races
	Failure: Timekeepers cannot access timekeeping features for the regatta's races
Exceptions	User associated with email does not exist, timekeepers don't gain access, users aren't notified of changes
Trigger	User clicks the "Edit timekeepers" button
Main success scenario	1. User inputs one or more emails into the "Invite" field 2. User clicks the "Save" button 3. Application grants users associated with those emails access to the regatta's timekeeping features 4. Application notifies user and invited users of success
Error scenarios	1a. User associated with an inputted email does not exist. An error message indicating this is displayed. 2a. Application fails to grant the invitees access. An error message indicating this is displayed. 3a. Application fails to notify user and invitees of success.
Alternative scenarios	2. User clicks the "Cancel" button. Edits are not saved.

Remove a timekeeper from a regatta	
Description	A regatta's admin can remove a timekeeper, revoking their access to the regatta's timekeeping features
Actors	Primary: Regatta admin
	Secondary: DB
Goal	Remove a timekeeper from a regatta
Preconditions	User is logged in, has created a regatta, and is on that regatta's page
Postconditions	Success: Specified user's access to timekeeping features is revoked
	Failure: Specified user retains access to timekeeping features
Exceptions	DB error
Trigger	User clicks the "Edit timekeepers" button
Main success scenario	1. User clicks the "Delete" button next to a timekeeper's name 2. User clicks the "Save" button 3. Application removes timekeeper's access to timekeeping features 4. Application notifies user of successful removal
Error scenarios	3a. User's permissions are not updated in the DB. A "something went wrong" error is displayed.
Alternative scenarios	2. User clicks the "Cancel" button. Edits are not saved.

Delete a regatta	
Description	A regatta's admin can delete the regatta, deleting all of its races and all relevant records from participants' profiles along with it
Actors	Primary: Regatta admin
	Secondary: DB
Goal	User deletes a regatta that they admin
Preconditions	User is logged in, has created a regatta, and is on that regatta's page
Postconditions	Success: The regatta is successfully deleted from the DB
	Failure: The regatta is not deleted from the DB
Exceptions	DB error
Trigger	The user clicks the “Delete regatta” button
Main success scenario	1. A popup confirming that the user wants to delete the regatta and all its races is displayed 2. User clicks the "Confirm" button 3. Application removes the regatta and its races from the DB, and updates its participants' records 4. Application confirms successful deletion of the regatta
Error scenarios	3a. The regatta is not removed from the DB properly. A “something went wrong” error message is displayed.
Alternative scenarios	2. The user does not want to delete the regatta and clicks the "Cancel" button. The regatta is not deleted.

Create a race	
Description	Create a new race, assign it to a regatta, and add boats/participants
Actors	Primary: Regatta admin
	Secondary: DB
Goal	Create a race, assign it to a regatta, and select participating boats from the regatta's list of registered boats
Preconditions	User is logged in and on a regatta's page
Postconditions	Success: Race is created and added to the regatta
	Failure: Application fails to create the race
Exceptions	Repeated IDs in list, DB error
Trigger	User clicks the "Add a race" button
Main success scenario	1. User inputs the race name and (optionally) scheduled date 2. User optionally adds boats the race by entering their IDs; suggestions from the regatta's list of registered boats are displayed 3. User clicks the "Save" button 4. Application saves the race and adds it to each participant's record in the DB 5. Application confirms successful save and creates a page for the race 6. User is directed to the race's page
Error scenarios	2a. A boat with the inputted ID has already been added to the list. An error message indicating this is displayed. 4a. Race cannot be saved due to an internal error. A "something went wrong" error message is displayed.

Edit a race's details	
Description	A regatta's admin can change the race's name and scheduled time
Actors	Primary: Regatta admin
	Secondary: DB
Goal	Rename a race
Preconditions	User is logged in, has created a regatta, and is the page of a race that belongs to the regatta
Postconditions	Success: The race's details are successfully changed
	Failure: Application fails to edit the race
Exceptions	Invalid input, DB error
Trigger	User clicks the "Edit" button
Main success scenario	1. User types a new name and/or date in 2. User clicks the "Save" button 3. Application saves details to the DB 4. Application confirms successful save
Error scenarios	1a. Invalid input. An error message indicating name requirements is displayed. 3a. Details are not saved to the DB. A "something went wrong" error is displayed.
Alternative scenarios	2. User clicks the "Cancel" button. Edits are not saved.

Add boats to a race	
Description	A regatta's admin or the race's timekeeper can add a boats from the regatta's list of registered boats to the race
Actors	Primary: Regatta admin, timekeeper
	Secondary: DB
Goal	User adds registered boats to the race
Preconditions	User is logged in, is a regatta admin or timekeeper for an existing race, and is on the race's page
Postconditions	Success: User successfully adds boats to the race
	Failure: Application fails to add boats
Exceptions	DB error
Trigger	User clicks the “Edit” button
Main success scenario	1. User adds boats the race by entering their IDs; suggestions from the regatta's list of registered boats are displayed 2. The user clicks the "Save" button 3. Application updates the race in the DB 4. Application confirms successful save
Error scenarios	3a. The application fails to update the information properly. A "something went wrong" error message is displayed.
Alternative scenarios	2. User clicks the "Cancel" button. Edits are not saved.

Edit boats in a race	
Description	A regatta's admin or the race's timekeeper can edit a boat's finishing time; switch it to DNS/DNF; or remove it from the race (but not the regatta)
Actors	Primary: Regatta admin, timekeeper
	Secondary: DB
Goal	User updates the race's boat list
Preconditions	User is logged in, is a regatta admin or timekeeper for an existing race, and is on the race's page or race's timekeeping page
Postconditions	Success: User successfully updates the race's boat list
	Failure: Application fails to update the race's boat list
Exceptions	DB error
Trigger	User clicks the "Edit" button
Main success scenario	1. For each boat the user wishes to update, they select the "Edit boat" button and input the desired finish time ID and/or participants 2. User clicks the "Save" button 3. Application updates the boats in the DB. 4. Application confirms successful update
Error scenarios	3a. The boats are not updated in the DB. A "something went wrong" error message is displayed.
Alternative scenarios	2. User clicks the "Remove boat" button. A popup confirmation is displayed, and the user can click either "Cancel" or "Confirm."

Delete a race	
Description	A regatta's admin can delete a race, removing it from the DB and regatta
Actors	Primary: Regatta admin
	Secondary: DB
Goal	User deletes the race from both the DB and the regatta.
Preconditions	User is logged in, is a regatta admin or timekeeper for an existing race, and is on the race's page
Postconditions	Success: User successfully deletes the race
	Failure: User fails to delete the race.
Exceptions	DB error
Trigger	The user pressed the “Delete race” button
Main success scenario	1. A popup confirming that the user wants to delete the race is displayed 2. User clicks the "Confirm" button 3. Application removes the race from the DB. 4. Application confirms successful deletion of the race
Error scenarios	3a. Race is not removed from the DB. A "something went wrong" error message is displayed.
Alternative scenarios	1. The race is in progress. The message confirms that the user wants to delete the race while it's in progress. 2. The user clicks the "Cancel" button. The race is not deleted. 3. The race is in progress. Application ends the race and timekeepers are kicked out.

Start a race	
Description	A timekeeper starts the race, beginning the stopwatch
Actors	Primary: Regatta admin, timekeeper
	Secondary: DB
Goal	Start a race and record its start time
Preconditions	User is logged in and on a race's page.
Postconditions	Success: A list of boat ID's and their finishing times is logged
	Failure: Boat ID's and their finishing times are not logged
Exceptions	No registered boats, race already started, stopwatch fails, DB error
Trigger	The user clicks the “Start race” button
Main success scenario	1. User is directed to the race's timekeeping page 2. A stopwatch begins 3. The race's start time is saved to the DB
Error scenarios	1a. No boats are registered for the race. An error message indicating this is displayed. 1b. Another user has already started the race. An error message indicating this is displayed. 2a. The stopwatch does not begin. A "something went wrong" error message is displayed. 3a. The start time is not saved to the DB. A "something went wrong" error message is displayed.

Mark when a boat finishes the race	
Description	A timekeepers marks when a boat finishes the race, saving the timestamp and boat's ID
Actors	Primary: Regatta admin, timekeeper
	Secondary: DB, SMS API
Goal	Save the current timestamp and boat's ID
Preconditions	User is logged in, has started a race, and is on the timekeeping page
Postconditions	Success: The boat's finishing time and ID are saved
	Failure: The boat's finishing time and ID are not saved
Exceptions	SMS request fails
Trigger	User clicks the "Mark" button
Main success scenario	1. A row is added to the "Finishing times" list with the current timestamp and an empty "ID" field 2. User fills out the ID field; selectable suggestions from the race's list of registered boats appear as they type 3. User clicks the "Save" button 4. The ID and timestamp are sent to the server via SMS and added to the race's page
Error scenarios	4a. Info cannot be sent due to an internal error. Application periodically reattempts request until it is sent.
Alternative scenarios	2. The inputted ID does not exist in the list. User manually finishes typing it in. 3. If all IDs have been accounted for, the stopwatch stops running and a "race completed" message appears to the user and on the race's page

Disqualify a boat from a race	
Description	A boat should be disqualified if it did not start or finish the race
Actors	Primary: Regatta admin, timekeeper
	Secondary: DB, SMS API
Goal	Disqualify the boat from the race and indicate whether it is a DNS or DNF
Preconditions	User is logged in, has started a race, and is on the timekeeping page
Postconditions	Success: The boat is disqualified from the race
	Failure: The boat remains in the running
Exceptions	Specified boat ID is not in the race, SMS request fails
Trigger	User clicks the "Add DNS/DNF" button
Main success scenario	1. User inputs the boat ID and selects whether it was a DNS or DNF 2. User clicks the "Save" button 3. The ID and result are sent to the server via SMS and added to the race's page
Error scenarios	1a. Specified ID is not registered for the race. An error message indicating this is displayed. 3a. Info cannot be sent due to an internal error. Application periodically reattempts request until it is sent.