Tandem Code Challenge

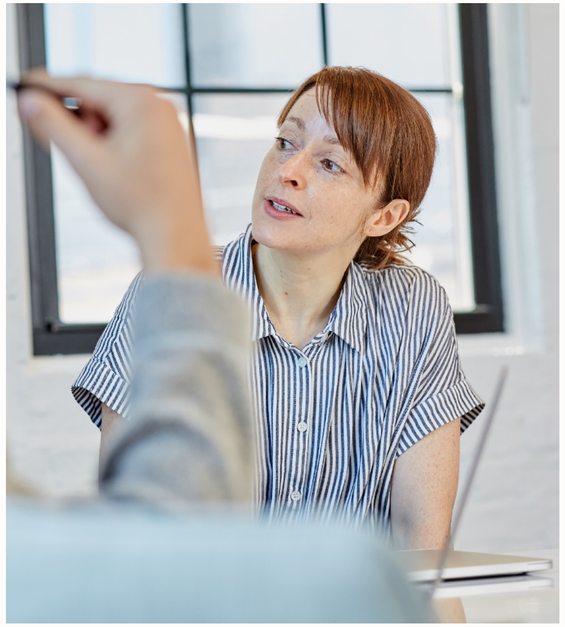
Senior Software Engineer | November 2019

Welcome!

We’re so excited you’re considering joining the Tandem team!

In this document, you will find 2 code challenge options. Please choose one challenge option to complete for your submission.

Each code challenge presents problem statements and requirements for two different types of systems.



About our challenge

There is no strict objective measure for pass/fail. Every code submission will be followed up with a review in person or on the phone where we can discuss and elaborate on your choices, and learn more about your thought process.

While there are defined acceptance criteria for each challenge, it is up to you to fill in the blanks for any requirement that is undefined. Requirements may seem unclear at times, but we are unable to provide additional information while your work is in progress. In the event something is ambiguous, make your own decision and provide the rationale.

Your submission should showcase your capabilities. We encourage you to be creative and play to your strengths. Use any language and platform you're comfortable with and expound on any area which interests you — so long as the core requirements are implemented.

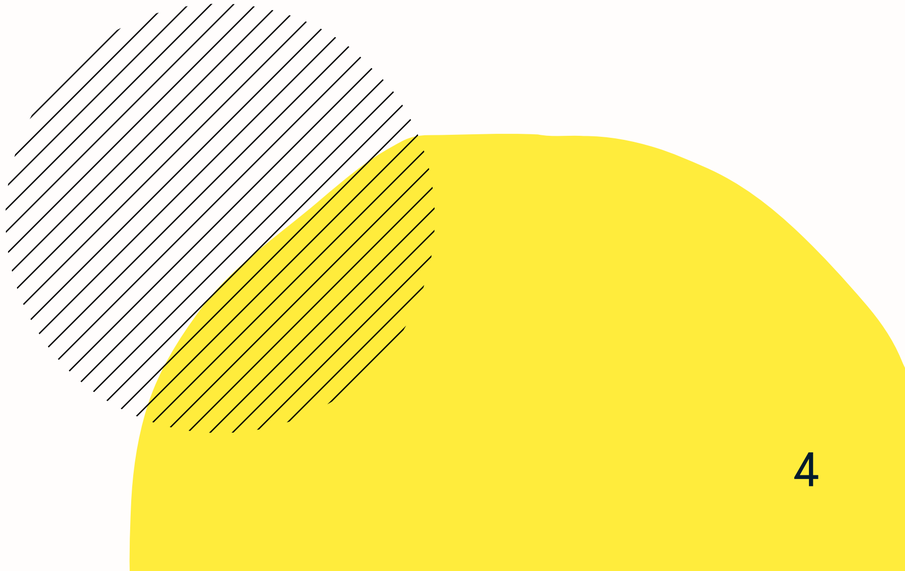
How to submit

We know everyone is busy and we want to provide the time you need to complete your chosen exercise. Please submit your challenge within seven days.

Once you're ready to submit, use the submission link we provided in the email and upload a zip file. Feel free to include your .git or .hg directory in the zip file with your source.

Please try to exclude any binaries or dependencies that can be built or resolved via a package manager. It's best to use as few external dependencies as possible to keep the setup simple. Your submission must include instructions for how to run your code. You must also list any system dependencies (e.g. Ruby 2.3, Erlang runtime, JDK8, etc).

Please do not post your solution online or share it with other people.



All is fair when we pair

At Tandem, pair programming is an integral part of our development workflow. We believe that pairing results in more effective problem solving and fewer bugs. It also builds a sense of shared responsibility for code and provides the opportunity to learn from each other.

Typically, on Tandem projects, each team has a lead developer who ensures that the developers on the team are rotating pairs frequently. For example, Dev 1 and Dev 2 paired during the last sprint, but in the upcoming sprint, Dev 1 should pair with Dev 3 and Dev 2 should pair with Dev 4.

Currently, a lead developer creates a rotation schedule spreadsheet they have to keep track of and update at the end of each sprint with a new pair rotation. This manual assignment/tracking can be tedious, especially when you are on a long project.

Tandem needs your help in making this process easier and automated for our new project. Following the initial project kick-off meeting, the lead developer needs to create a pairing rotation schedule for the duration of the project.

Goal

Your goal is to create a user interface for a lead developer to enter the number of sprints, the names of all developers, and any paid time off they have requested during the project. Using this information, your application should generate the best pairing rotation possible. By best, we mean that each developer on the team should pair with each of their teammates an equal number of times or as close to equal as possible.

The lead developer wants to see who is pairing, on vacation, and working solo during a sprint (one person will have to work solo if there is an odd number of developers on the team).

Additionally, the lead developer wants to be able to come back to the schedule without having to re-enter all the project and developer details.

Assumptions

- Each sprint has a duration of 1 week.
- Each sprint starts on Monday and ends on Friday (there are no partial sprints).
- Each day of paid time off (PTO) is a full day off.
- All PTO was scheduled in advance of the project starting.

Acceptance Criteria

- The user can view the pair rotation schedule.
- Each developer pairs with every other developer an equal number of times during the project, or as close to equal as possible.
- The user can view the requested vacation time for the developers on the project during each sprint.
- During any given sprint, at most 1 developer is soloing.
- The sprint schedule data is persisted.

Guac is extra

We have an annual potluck at Tandem, and it's always hard to decide what to bring. Many Tandemites love tacos—there are so many places to get good tacos around the Chicago office—so you want to bring some to the event, but you're afraid the team is getting burned out on the same old taco choices.

You also want to be mindful of your team members with dietary restrictions such as those who eat Vegetarian, Vegan, Gluten-free, Dairy-free, Halal, Kosher, and those with nut allergies. For example, if a recipe does not contain any meat, it is vegetarian. If a recipe does not contain any pork, it is halal.

Coming up with new taco recipes that are interesting and dietary-restriction friendly is hard. Create an application that will help you find inspiration for our next potluck.

Goal

Your goal is to create an application that will let you search and filter taco recipes. Leverage an API to find the taco recipes. Use a free API to start, but be sure to build the application in a way that makes swapping out different APIs as easy as possible. In the future, Tandem may decide to extend this application and pay for a more fully-featured API.

Create a persisted data store for ingredients so that you will be able to display recipes you fetch according to their dietary friendliness. You should be able to filter the taco recipes by both ingredients and dietary restrictions.

Assumptions

- Search results can update on form submit. Searching does not need to occur in real-time, as the user types.
- Use your best judgment with regard to the dietary restrictions. Identifying these can be complicated. Please be sure to document your assumptions.

Acceptance Criteria

- User can search by recipe name and see matching results.
- User can filter recipes by ingredients and see recipes that contain those ingredients.
- User can filter recipes by dietary restrictions and see recipes that are safe for the selected restriction.
- The taco recipe data is fetched dynamically from an API.
- Ingredient data is persisted.

Data

Here are some suggested free APIs you could use for your application. However, if there is an API you would rather use, feel free. In your README, please explain why you chose the API you're using.

- https://www.themealdb.com/api.php
- http://www.recipepuppy.com/about/api/
- https://github.com/evz/tacofancy-api


tandem