# MeetTheTeam Project



## Introduction

As part of our evaluation process, we'd like you to implement a Symfony 4.4 project that involves a user login flow, and a CRUD, following the requirements laid out in this document.

The goal of this project is to evaluate your knowledge of Symfony's Best Practices, as well as your level of familiarity with components in Symfony's ecosystem.

From a business perspective, this app will allow registered users to log into the app, add new colleagues they've met in a new team they've joined, and write small notes for each colleague they'd like to remember.
You can imagine using this app once you join a new remote team, to help you meet your colleagues and write small summaries on what each person's role & interests are.

## Project Timeline & Deliverables

Once this brief is shared with you, we'll set a due date of 7 calendar days for you to submit your Symfony app implementation.

We understand that sometimes things come up in people's schedules. In case you need more time, don't hesitate to reach out telling us why and we'll adapt to your needs.

Please share your implementation by pushing it to gitlab.com or bitbucket.com as a private repository and sharing that repository with rodyhaddad@coddict.com.

## Evaluation Criteria

For transparency, we'll lay out the criteria we'll be evaluating once we have the Symfony app:

1. Does the implemented functionality follow all the requirements mentioned in this document?
2. Is the app a functioning Symfony 4.4 project?
3. Does the written code follow good Coding Standards (PSR-1 & PSR-12), and is easily readable?
4. Are you following Symfony's Best Practices, and not re-inventing the wheel unnecessarily? Specifically, we care about your usage of Symfony:
    a. Controllers, Commands, Services (DI), Forms, Validations, Annotations, Yaml Configuration, Testing
5. Does the app have any sort of unit/integration tests developed, covering at least part of the code?
6. Are the git commits messages descriptive?

With this in mind, here are the things we **won't** be evaluating:

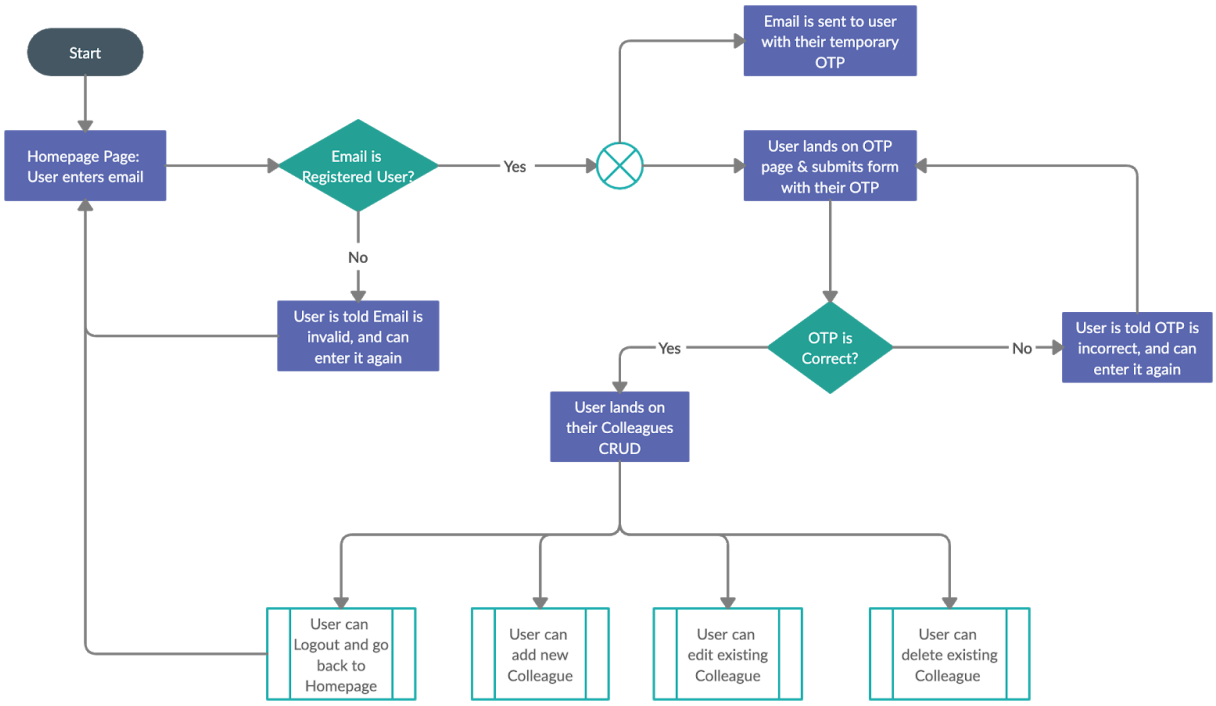
1. Your test coverage doesn't need to be extensive: As long as you can show knowledge in writing some automated tests, that's what matters. We're not looking for full coverage, just enough to show proficiency.
2. The overall frontend design: As long as it's usable, that's what matters. We suggest using Bootstrap 4 to simplify your life, but this isn't a requirement

## Overall Behaviour & User Flow

From a bird's eye view, the app will allow users to:

- Land on a login form with an "email" input to identify themselves.
- Once they're identified, they'll receive an email with an OTP, that they'll need to enter to authenticate.
- Once they're authenticated, they'll see a list of their colleagues and can:
    - Add new colleagues
    - Edit an existing colleague
    - Delete an existing colleague
- They can also log out & go back to the login form

The User Flow looks as follows:




Link: https://app.creately.com/diagram/EY2RjmE9Tg5/view

The rest of the document describes the above functionalities with user stories & acceptance criteria.

## User Personas & Glossary

Glossary:

| | |
|---|---|
| **OTP** | One-Time Password. It consists of case-insensitive alphanumeric characters, generated at random & sent to Identified Users by email |
| **Registered User** | A user that has been registered by an Administrator, using their email & full name. |

User Personas:

| | |
|---|---|
| **Administrator** | A server user that has the ability to run “bin/console” commands on the app |
| **Anonymous User** | A web user that hasn’t been identified yet |
| **Identified User** | A web user that has entered an email that matched a Registered User, but didn’t enter their OTP yet |
| **Authenticated User** | A web user that has entered an email that matched a Registered User, as well as the correct OTP, and is now logged in |

# General Technical Criteria

These are Technical Criteria that apply to all user stories:

- Forms:
  - All forms should use Symfony's Form component, and have CSRF enabled
- Validation:
  - All validation rules should be done using Symfony's Validation component
  - If required, you should use custom Validation Constraints
- Configuration:
  - Routes/Entities/Validation configurations should be done using annotations.
  - For other configuration, use YAML and not XML/PHP
- Third-party:
  - You should use the recommended third-party packages, like Doctrine, Twig, Mailer, etc., that you can find promoted in the symfony.com documentation
  - You can use VichUploaderBundle to simplify the image upload
  - To simplify the CRUD functionality, you can use EasyAdmin 3.x or Symfony Maker's CRUD generator. You'll find that both will need a bit of tweaking though.

# User Stories: Administrator

**Intro:** As the app won't have a "registration" process, *Administrators* are responsible for adding new Registered Users using a custom Symfony command, accessed through "bin/console".

> As an Administrator, I can register new users from the CLI, so they can log into the app & use their Colleagues CRUD.

Acceptance Criteria:

- The command must be accessed through "bin/console *${command-name}*"
- The command must require the Administrator to enter the user's Full Name & Email
- The Administrator can enter these details as command options
- In case the Administrator omits the command options, the command must ask for these values interactively
- The command must validate that the Full Name is not blank and that the Email is a well-formatted email. In case the validation fails, the command must re-prompt the user to enter these values properly
- Once the user creation is complete, the command must output the resulting User Id

Technical Criteria:

- The users should be saved in a MySQL Or SQLite Database, using Doctrine ORM & Doctrine Entities
- The validation must be done using Symfony's Validation system
- Symfony's Console component should be used to register & run the command using "bin/console"

# User Stories: Anonymous User

**Intro:** An anonymous user is any web visitor that is yet to be identified. They can only see the login form, and can't access the Colleagues CRUD

> As an Anonymous User, I can access the app's homepage, so I can enter my email and identify myself

Acceptance Criteria:

- The homepage should return a well-formatted HTML page
- The homepage should have a simple HTML form with an "Email" input and a submit button

> As an Anonymous User, I can submit my email on the homepage form, so that I can identify myself

Acceptance Criteria:

- If the submitted email matches a Registered User's email, this user should become an Identified User

> As an Anonymous User, I should see a validation error, if I submit an email that's not properly formatted

*No special criteria apart from the general criteria mentioned earlier in the document.*

> As an Anonymous User, I should see a validation error, if I submit an email that's not linked to a Registered User

Technical Criteria:

- The validation must use Symfony's Validation component, using a custom Validation Constraint

> As an Anonymous User, if I open any other app URL, I will receive a 403 Forbidden page, with a link to the homepage

Technical Criteria:

- This should use Symfony's Custom Error Pages
- The page should explain to the user that they don't have access to this page, with a link to the homepage

# User Stories: Identified User

**Intro:** Identified Users are users who entered a correct email of a registered user, but still need to authenticate themselves first before accessing the Colleagues CRUD

> As an Identified User, as soon as I identify myself, I should receive an email with an OTP to authenticate myself

Acceptance Criteria:

- The email must be well-formatted, with HTML & Text versions
- The email should contain a randomly generated OTP that will allow the user to authenticate

Technical Criteria:

- The OTP should expire 20 minutes after it was sent
- The OTP should be saved either in Redis (recommended) or in MySQL/SQLite

> As an Identified User, I should land on a page that has an “OTP” input, so that I can submit it & authenticate myself

Acceptance Criteria:

- The user should be redirected to this page as soon as they successfully identify themselves
- It should just have an “OTP” input and a submit button
- If they submit the correct OTP, they should become an Authenticated User, and be redirected to their Colleagues CRUD
- If they submit the wrong OTP, they should receive a validation error

Technical Criteria:

- The validation must use Symfony’s Validation component, using a custom Validation Constraint

# User Stories: Authenticated User

**Intro:** Authenticated Users are users who authenticated themselves using a registered email and a proper OTP. They can access their Colleagues CRUD, and log out if need be.

> As an Authenticated User, as soon as I authenticate myself, I should land on my Colleagues listing page

Acceptance Criteria:

- This page should list all the existing Colleagues associated with this Authenticated User. Other Registered Users can have their own Colleagues that shouldn't be displayed here.
- It should have buttons to add a new Colleague, edit an existing Colleague, or delete an existing Colleague
- It should have a link to log out and become an Anonymous User again
- The columns to display in the Colleagues listing are:
    - Picture (as an <img />)
    - Name
    - Role
    - Notes (truncated with an ellipsis)
    - Actions (Edit & Delete)
- If any of the details are `null`, they should be displayed as "N/a"

> As an Authenticated User, I want to add a new Colleague, to more easily remember who they are

Acceptance Criteria:

- The form should contain the following fields:
    - Name
        - Regular Text Input
        - Required, Not Blank
    - Picture
        - File Upload Input
        - Not required
    - Role
        - Regular Text Input
        - Not required
    - Notes
        - TextArea
        - Not required
- The Picture field should only support files that are Image formats
- If there are any validation errors when submitting the form, they should be properly displayed to the user

- Once the form is submitted, the created Colleague must be associated with this specific Authenticated User, so no other Registered User can see this entry

> As an Authenticated User, I want to edit an existing Colleague, to more easily remember who they are

Acceptance Criteria:

- The form should contain the same fields & validation behaviour as the previous “add” User Story
- An Authenticated User should be denied access to edit any Colleague associated with other Registered Users

> As an Authenticated User, I want to delete an existing Colleague, to clean up unnecessary entries

Acceptance Criteria:

- When the “Delete” button is clicked, the user must reconfirm their action through a popup with a CSRF-protected form, so they wouldn’t delete it by mistake
- After deletion, the listing page is re-displayed, with the deleted entry now gone
- An Authenticated User should be denied access to delete any Colleague associated with other Registered Users

> As an Authenticated User, I want to logout, so I can become an Anonymous User again

Acceptance Criteria:

- Once the Authenticated User logs out, they should be redirected to the app homepage, and no longer have access to their Team CRUD

# Other Consideration: Authentication Mechanism

Since the app's authentication mechanism doesn't use the traditional "Username/Password" schema, making it work with Symfony can be a bit hard, depending on how deep you've dived into Symfony's Security internal. To not complicate the app more than it should be, you can:

- Avoid using Symfony's Security system completely, and instead, save the user's status in their HTTP session.
  This means you wouldn't need to configure UserProviders, Firewalls, Roles, Access Controls, etc.
- If you still want to use Symfony's Security system, we suggest using Symfony's Custom Guards (docs)

As long as the Acceptance Criteria of all the stories are respected, we won't evaluate the way you'll implement the Authentication Mechanism, to keep things simple.