The Area Project Plan

by

Café-Sur-Cour

2025

AUTOMATION - APP
ESTD 2025

Summary

0. Project Overview

0.1 Project Context

This project is an automation platform inspired by IFTTT, where users can connect different services and create workflows of the form ‘If this happens, then do that’.

.

0.2 Project Team

Noa Roussiere
Eliott Tesnier
Alban Roussee
Matisse Marsac
Albane Merian

1. Stack Explorations

1.1 Potential Stacks

Before starting coding, we first explored the different technologies we could implement for each module, including backend, frontend, and deployment.

Here is a list of the languages and technologies we considered, organized by module:

Backend :

- Node Js
- Nest Js
- Fastify
- Express

FrontEnd :

- Kotlin with JetPackCompose
- Flutter
- React Native
- Next Js
- React

Deployment :

- Dokcer
- Jenkins
- GitHub Action

1.1 Pro's and Cons's

Deployment & CI/CD Considerations:

For some technology choices, we did not need to create a proof of concept to make a decision, notably on the deployment side. For Continuous Integration (CI) and Continuous Deployment/Delivery (CD), we considered using either a Jenkins instance or GitHub Actions workflows.

	Workflow	Jenkins
Pro's	Native to github Easy SetUp	Open Source Easy to use
Con's	None	Needs to be hosted Excessive

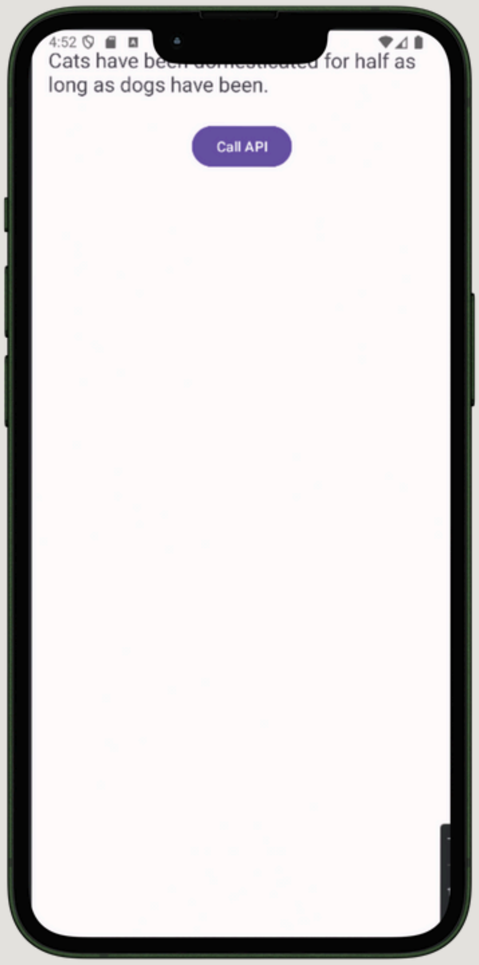
While Jenkins could handle the project, it is overkill given the size of our project and the fact that it will reside in a single repository. Therefore, we opted for GitHub Actions as a simpler and more practical solution.

1.2 Proof Of Consept Mobile

Mobile devlopement is something we haven’t really done yet so to make more informed choices, we realised two POCs using the two most common languages to develop app on android.

Kotlin :

For the first one we used kotlin here is the pro’s and con’s we concluded.



Cats have been domesticated for half as long as dogs have been.

Pros:

- The project structure appears to be quite modular, which can facilitate maintenance and scalability.
- Android Studio provides robust tooling and features that enhance development productivity for Android projects.

Cons:

- Separated XML files for UI layouts can make the codebase harder to navigate and reduce overall readability.
- The use of Java-style classes may feel un-intuitive, especially for developers more familiar with Kotlin's idioms.

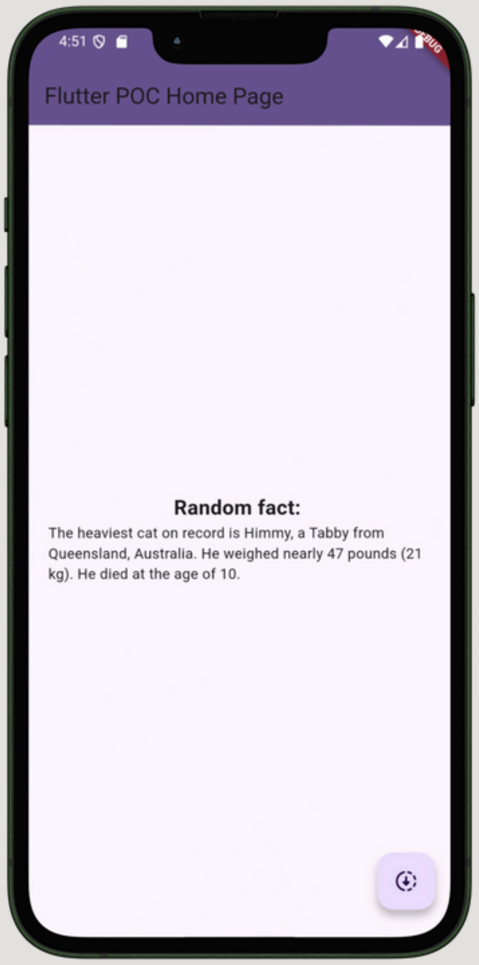
Flutter :

Pros:

- Fast development: Hot reload speeds up UI iteration and testing.
- Rich UI: Built-in widgets for beautiful, customizable interfaces.
- Strong community & Google support: Frequent updates and resources.
- Single codebase: Easier maintenance and feature parity across platforms.

Cons:

- Large app size: Flutter apps tend to be bigger than native apps.
- Limited native APIs: Some platform-specific features require custom plugins.
- Performance: Not always as fast as fully native apps for complex tasks.
- Smaller ecosystem: Fewer third-party libraries compared to native platforms.
- Platform-specific UI: May require extra work to match native look and feel.

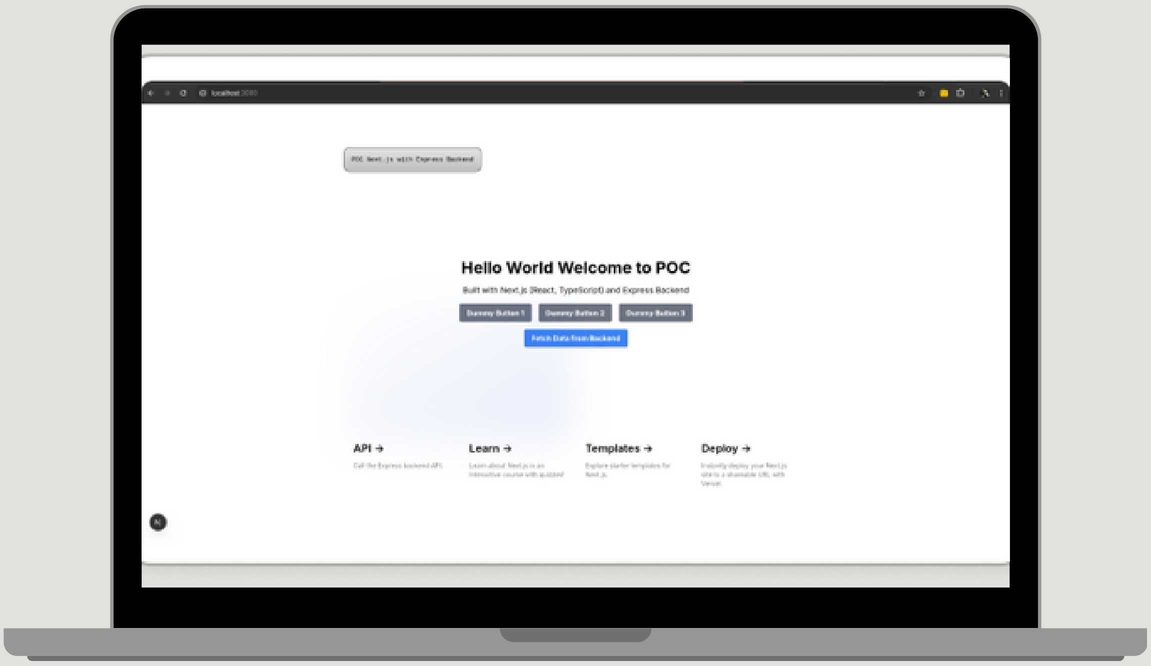


Random fact:

The heaviest cat on record is Himmy, a Tabby from Queensland, Australia. He weighed nearly 47 pounds (21 kg). He died at the age of 10.

1.3 Proof Of Consept WEB

Next and Tailwind


localhost:3000
POC Next.js with Express Backend
Hello World Welcome to POC
Built with Next.js (React, TypeScript) and Express Backend
Dummy Button 1
Dummy Button 2
Dummy Button 3
Fetch Data from Backend
API →
Call the Express backend API.
Learn →
Learn about Next.js in an interactive course with quizzes!
Templates →
Explore starter templates for Next.js.
Deploy →
Instantly deploy your Next.js site to a shareable URL with Vercel.

Pros:
- The framework is intuitive and easy to use, making development straightforward.
- Already familiar to many developers, which speeds up onboarding and productivity.
- Highly modular, allowing for flexible and maintainable code organization.
- Powerful and clean architecture that supports scalable web applications.

Cons:
- JavaScript syntax may not be appealing to everyone and can be less enjoyable for those who prefer other languages.

1.4 Proof Of Consept Backend

Golang

Pros:
The syntax is great (mix between JavaScript and C++) it makes this language easy to take charge.
Language created especially for backend development and easy to containerized with Docker.

Cons:
File architecture is a bit complicated, include loops are forbidden and package separation is not really intuitive.

A lot a packets to pull but not really diversified.

Express :

Pros:
Installs in 2 minutes, no unnecessary crap. Thousands of middlewares for everything you need (auth, logs, CORS...).
Handles all your URL paths without issues. On Node.js, it handles webhooks like a boss. Docs everywhere, tons of tutorials, you never struggle alone

Cons:
Gotta think a bit more than all-in-one frameworks. Total freedom, but it can lead to messy code if you screw around. Basic features, you gotta add them yourself

PassPort JS

Pros:
Passportjs is really flexible multiple external lib

Cons:
There is to much external lib, not a lot of documentation

1.5 Conlusion

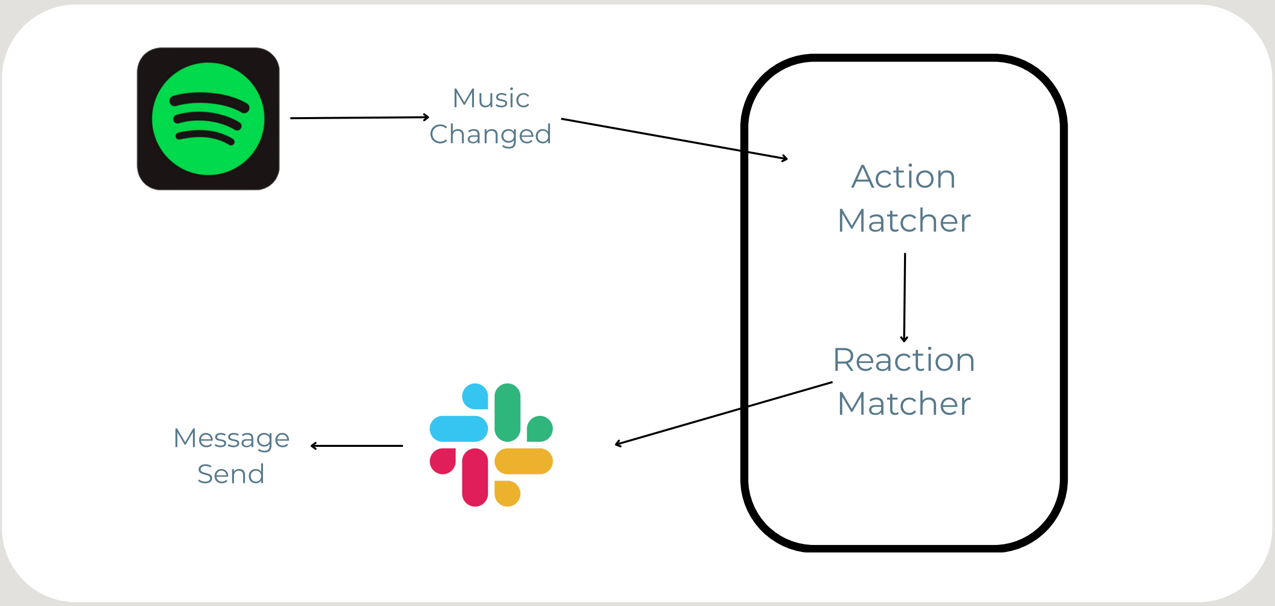
From the PoCs we created, along with insights from the Survivor Piscine, our technology choices became clear:

- Backend: Express with TypeScript
- Web Frontend: Next.js with Tailwind CSS
- Mobile Frontend: Flutter

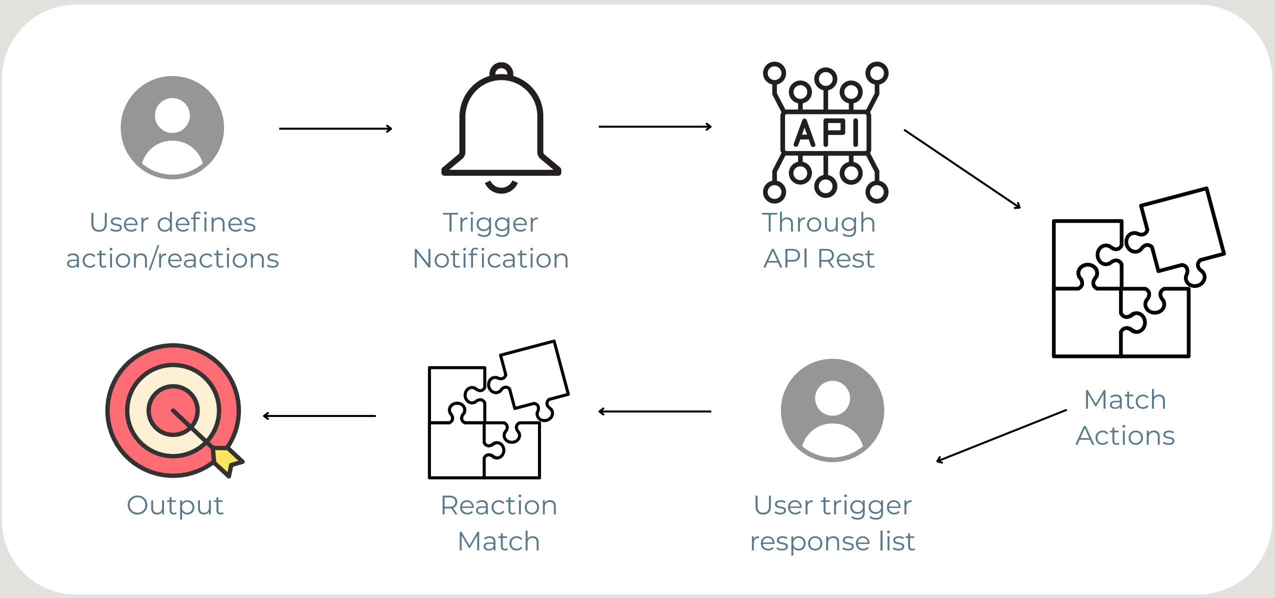
2. Architecture Plan

2.1 Integration Diagram

To visualize the different services and there interactions here is a diagram.



2.2 DataFlow Diagram



2.3 API

Here we have regrouped a list of potential api's to use for the action reaction element of the app, or even as services.

API	Link	Action
Spotify	https://developer.spotify.com/dashboard/create	No webHook
Github	https://docs.github.com/en/rest?apiVersion=2022-11-28	Action
Discord	https://discord.com/developers/docs/intro	Some Action but not alot and no web hook
Twitter	https://developer.x.com/en/docs/x-api	Web Hook but needs a fee
Nasa	https://api.nasa.gov/	No hooks, but Actions possible
Google	https://developers.google.com/	Action/WebHook
Microsoft	https://learn.microsoft.com/fr-fr/graph/overview?view=graph-rest-1.0	Action/Webhook
Twitch	https://dev.twitch.tv/docs/api/	Action/Webhook
Slack	https://docs.slack.dev/	Action/Webhook

2.4 OAuth Service Choices

For the services we will use with OAuth, we deliberately chose well-known platforms. The reason is that larger providers generally offer better documentation, more stable APIs,
Here are the services we selected and why:

GitHub :
Provides a robust OAuth implementation.
Offers a wide range of API endpoints (repositories, commits, etc.). Supports multiple webhooks, allowing real-time triggers

Google :
As one of the most widely used service providers, Google offers multiple APIs and integrations. APIs span across Gmail, Google Calendar, Google Drive, YouTube, and Google Cloud services.

Microsoft 365 :
Similar to Google's suite, Microsoft 365 provides APIs for Outlook, Teams, OneDrive, and SharePoint. Offers both REST APIs and webhook subscriptions for real-time updates.

Spotify :
While Spotify does not provide webhooks for real-time triggers, it does offer a well-documented Web API. Enables access to music playback, playlists, user libraries, and artist/track data.

Twitch :
The streaming platform offers a comprehensive API for streams, channels, and users. Supports EventSub webhooks, allowing event-driven triggers

Meta :
Through the Meta Graph API, developers can integrate with multiple platforms. Offers APIs and webhooks for Facebook Pages, Instagram accounts, and Messenger bots. Covers a wide range of use cases from social media posting to real-time notifications.

2.5 Action Reaction

From the service choices we maid some of them don't offer, actions or reactions.
Here is the one we defined for now.

Actions :

Github :

- Push on a repositorie
- Pull request on a repositorie
- Merging pull request

Google :

- Mail received
- Event in X time

Microsoft 365 :

- Drive folder changes

Twitch :
- New follower
- New subscriber
- Stream started

Timer :
- Every X time
- Time Elapsed

Meta :
- Instagram message
- Messenger message

Reactions :

Spotify :
- Pause music
- Skip to next track

Google :
- Send email to X
- Create event

Meta :
- Send Instagram Message to X
- Send Messenger message to X

3. Task BreakDown

2.1 Module Separation

To organize our development efficiently, we separated the application into distinct modules.

Backend Modules:

- Authentication
- OAuth 2.0 integration
- Action handling
- Reaction handling

Frontend Modules (Web and Mobile):

- Authentication / Registration
- Home Page
- Action / Reaction management
- Profile managemen

4. Project Scheduling

4.1 Project repartition

Back-End :
Eliott Tesnier
Matisse Marsac
Albane Merian

Mobile :
Alban Roussee

Web :
Noa Roussiere

4.2 Scheduling

PLANNING :

1. Understand problem and context of the project
2. Stack research
3. Creations of Proof Of Concepts
4. Start Documentation
5. Define project architecture
6. Choose Action/Reactions and services
7. Divide the work in task

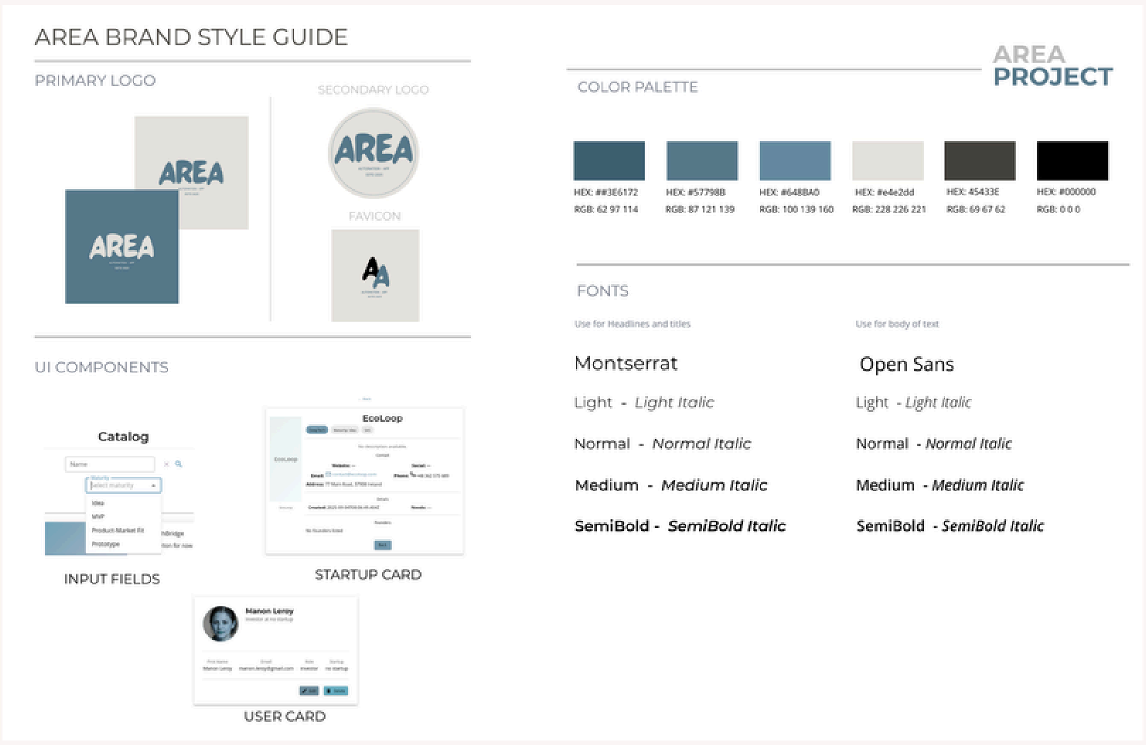
Minimum Viable Product :

1. Implement and Initialize the DataBase
2. Create the template for the Action & Reaction
3. Implement the basic authentifaction of a user
4. Create one basic OAuth
5. Implement the basic of the Web and mobile

5. Brand Definition

5.1 Visual Branding

To start our project more easly and cleanly, so we thought it best to create a brand chart and style guide. Regrouping the chosen fonts and color palette and the suposedly ui components we could use.



AREA BRAND STYLE GUIDE

PRIMARY LOGO

SECONDARY LOGO

FAVICON

UI COMPONENTS

INPUT FIELDS

STARTUP CARD

USER CARD

AREA PROJECT

COLOR PALETTE

HEX: ##3E6172	HEX: #57798B	HEX: #6488A0	HEX: #e4e2dd	HEX: 45433E	HEX: #000000
RGB: 62 97 114	RGB: 87 121 139	RGB: 100 139 160	RGB: 228 226 221	RGB: 69 67 62	RGB: 0 0 0

FONTS

Use for Headlines and titles	Use for body of text
Montserrat	Open Sans
Light - *Light Italic*	Light - *Light Italic*
Normal - *Normal Italic*	Normal - *Normal Italic*
Medium - *Medium Italic*	Medium - *Medium Italic*
SemiBold - *SemiBold Italic*	SemiBold - *SemiBold Italic*