

**Advanced Algorithms & Advanced Programming**

**TSN Project Report**

Skander Kechaou

Har Aziz Singh

Kacper Janus

Table of Contents

10

[I. Generalities 6](#_Toc199941880)

[II. Needs analysis specifications 8](#_Toc199941881)

[*1. Actors identification & user stories* 8](#_Toc199941882)

[*2. Functional specifications* 9](#_Toc199941918)

[III. Design and architecture 10](#_Toc199941926)

[*1. Introduction* 10](#_Toc199941927)

[*2. Logical Architecture* 11](#_Toc199941928)

[*3. Class Diagram* 13](#_Toc199941949)

[IV. Project Presentation 14](#_Toc199941950)

[*1. Application Development* 14](#_Toc199941951)

[*3. Scenarios* 16](#_Toc199941981)

1. Generalities
2. *Project Overview*

The project involves …

* 1. Purpose:

…

* 1. Objectives:
* …
* …
* …
  1. Scope of the Project:

The platform will include the following functionalities:

* …
* …
* …
  1. Tools and Technologies Used

…

* 1. Expected Outcome

The successful implementation of this platform will:

* …
* …
* …

1. *Methodology Selection*

Choosing the right methodology ensures that the project is completed efficiently.

**Agile** methodology was chosen for various reasons:

* Iterative Development: The project is broken into smaller increments aka sprints, each delivering a functional part of the system.
* Flexibility: Agile relies on iterative cycles, allowing teams to adapt to changes and regularly refine their work (Wrike, n.d.).
* Continuous Feedback: Involves the supervisor’s feedback regularly to ensure the development aligns with the expectations.

**Scrum** is the specific framework chosen since our project has rapidly-changing requirements and in this case allowing for:

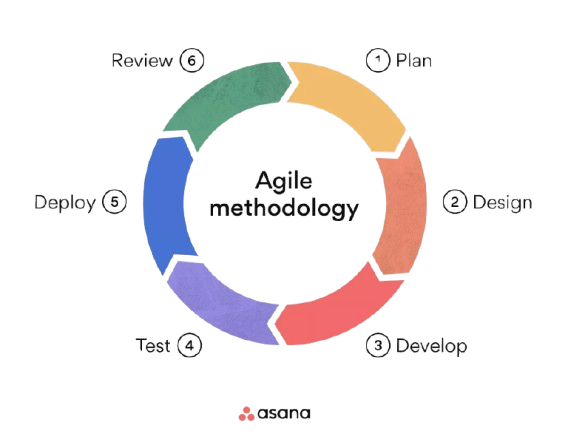
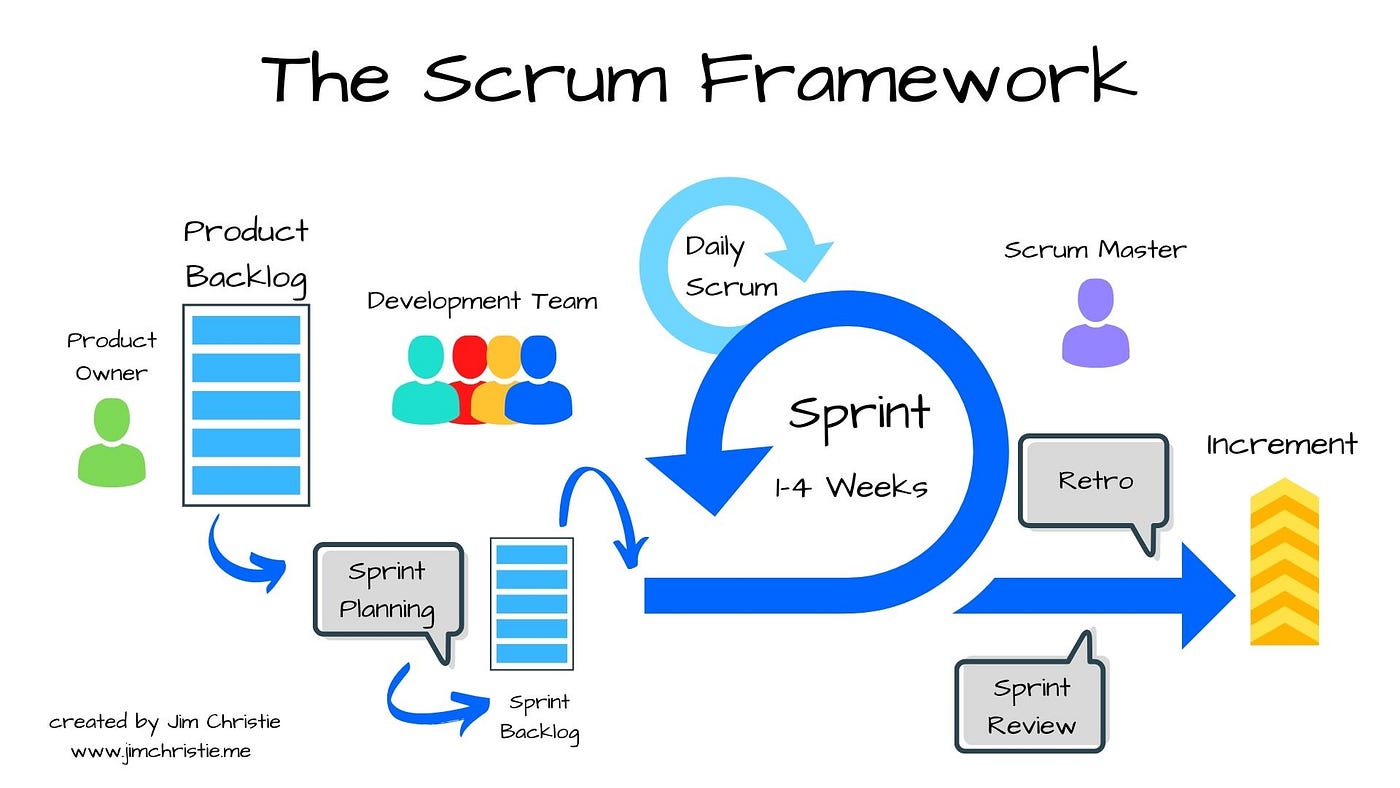
* Sprint planning (using short sprints)
* Daily meetings
* ******Incremented deliverables

Figure - Agile Methodology

Figure - Scrum Framework

1. Needs analysis specifications

*1. Actors identification & user stories*

The identified actors are necessary for the implementation of the system:

* HR department providing resource (employee) information
* Media & Energies department providing project assignment information (admin and users being other employees with access)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | User Story | Task ID | Task | Complexity |
| 1 | As a user I want to … | 1.1 | Establish authentication API endpoint | 1 |
| 1.2 | Create an authentication controller with log in logic | 2 |
| 1.3 | … | 0 |
| 2 | As an admin, I want to … | 2.1 | … | 1 |
| 2.2 | … | 2 |
| 2.3 | … | 0 |
| 2.4 | … | 0 |

*3. Functional specifications*

* 1. Functional specifications:
* …
  + …
  + …
* …
* …
* …

1. Design and architecture

*1. Introduction*

The design and architecture phase translates the functional and non-functional specifications into a structured technical solution. This includes defining the logical components of the system, their interactions, and the core data structures. The objective is to establish a robust, scalable, and maintainable architecture to meet business goals.

*2. Logical Architecture*

2.1. Overview

The system is organized into three primary layers (MVC):

* + - Frontend Layer:
      * …
    - Backend Layer:
      * …
    - Database Layer:
      * …

2.2. Logical Components

* …
* …
* …
* …

2.3. Data Flow

* Frontend Requests:

Users interact with the interface to send requests (e.g., create a new employee or assign a project).

* Backend Processing:

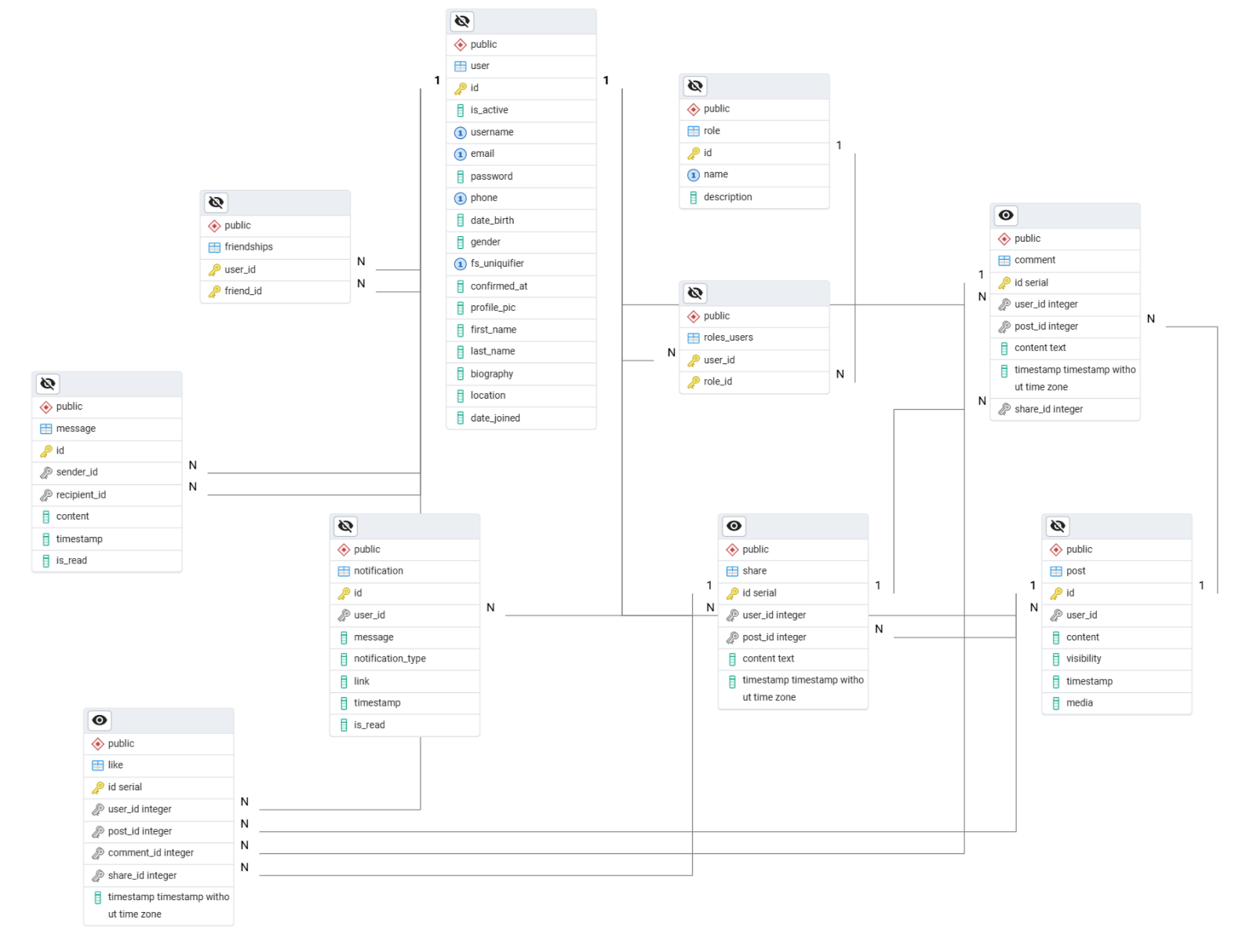
The backend validates and processes these requests, applying business logic.

* Database Transactions:

The backend interacts with the database to retrieve or modify data.

*3. Class Diagram*

The class diagram illustrates the data structure and relationships between various entities within the system.

****

1. Project Presentation

*1. Application Development*

This section provides an overview of the various services that form the backbone of the application, detailing their roles, functionality, and the technologies used in their implementation.

1.1 Backend Service

The backend service serves as the core of the application, handling all server-side logic and database interactions. It is built using Node.js with Express.js and follows a modular architecture for scalability and maintainability.

* + - Key Components:
      * Models: Define the structure of the database entities using Sequelize ORM, facilitating database interactions.
      * Controllers: Manage application logic by processing requests and interacting with models to fetch or manipulate data.
      * Routes: Define the API endpoints and map them to specific controller methods for seamless communication with the frontend.
      * Migrations: Handle database schema evolution using version-controlled scripts to ensure consistency.
      * Seeders: Populate the database with initial data, enabling quick setup and testing of the application.
      * Exposes port 3000 on the container to 3000 on our local machine (localhost)

1.2 Data Enrichment Service (populate\_db.py)

…

* + - Key Components:
      * …
      * …
      * …

1.3 Database Service

…

* + - Features:
      * …
      * ...
      * Exposes PostgreSQL on 5432 port, accessible from other services.

1.4 Frontend Service

…

* + - Features:
      * …
      * …
      * …
      * …

*2. Scenarios*