Intern Onboarding Guide – Enterprise Application Project (EAP)

Purpose of this guide

Welcome to the **Enterprise Application Project (EAP)**.

This guide explains:

- how you are expected to work as a team;
- which roles exist and who holds them;
- how planning and delivery are organised;
- what "quality" means in this project;
- which tools and infrastructure are available.

This guide gives you **context and expectations**.
Detailed rules, templates, and agreements are provided in separate documents.

The project mindset

This project is set up as a **professional software development project**.

That means:

- you work in a Scrum team;
- you take responsibility for planning, quality, and delivery;
- problems, defects, and operational issues are part of normal work;
- staff support and coach you, but do not take over your roles.

You are expected to act as a **professional team**, not as students following step-by-step instructions.

Team roles and responsibilities

Product Owner (team role)

One team member acts as the **Product Owner**.

The Product Owner:

- defines and orders the Product Backlog;
- clarifies requirements and acceptance criteria;
- makes scope and priority decisions;
- represents stakeholder interests within the team.

The Product Owner role is held by a **team member**, not by staff.

Staff may act as stakeholders or advisors, but they do **not** make product decisions.

Scrum Master (team role)

One team member acts as the **Scrum Master**.

The Scrum Master:

- facilitates Scrum events;
- ensures Scrum rules and timeboxes are respected;
- helps the team identify and remove impediments;
- supports transparency and collaboration within the team.

The Scrum Master is a **team role**, not a staff role.

Staff members act as **Scrum Coaches**.
They support and challenge the Scrum Master, but they do **not** run Scrum events or take over the Scrum Master's responsibilities.

DevOps responsibility (team responsibility)

One or more team members take responsibility for **DevOps activities**, such as:

- CI/CD pipelines;
- deployments;
- monitoring and logs;
- operational issues.

DevOps is a **team responsibility**.

You are expected to:

- detect problems early;
- investigate delivery and pipeline failures;
- communicate clearly about operational issues;
- work together on recovery or rollback when needed.

Development responsibility (shared responsibility)

All team members contribute to **building, testing, and improving the product**.

Development work includes:

- writing code (frontend and backend);
- creating and maintaining tests;
- code reviews;
- debugging and fixing defects;
- refactoring and technical improvements.

Shared responsibility

All team members contribute to building, testing, improving, and operating the product.

You share responsibility for:

- achieving the Sprint Goal;
- product quality;
- professional behaviour;
- collaboration and communication.

Available tooling and infrastructure

Source control

GitHub (private repositories)

- All code is stored in private GitHub repositories
- Git Flow workflow is mandatory:
  - `main` branch: production-ready code
  - `dev` branch: integration branch
  - `feature/*` branches: individual feature development
- Branch protection rules are active on `main` and `dev`
- Pull requests are required for merging

Project management

Jira (paid license)

- Sprint planning and tracking
- Product Backlog management
- Scrum board for Sprint work
- Access credentials provided by staff

CI/CD

GitHub Actions

- Automated testing on pull requests
- Continuous integration pipeline
- Deployment automation to target environment
- Pipeline configuration is part of your repository

Deployment platform

TransIP VPS (Linux)

- Production-like deployment environment
- Linux-based server
- SSH access for deployment
- Access credentials provided by staff

Communication

Microsoft Teams

- Primary communication tool
- Daily standups (when remote)
- Sprint ceremonies
- Team discussions and decisions

How you plan work

Definition of Ready

Before work is taken into a Sprint, the team agrees that a backlog item is **Ready**.

This means:

- the goal of the item is clear;
- the value for the product or user is understood;
- acceptance criteria are written and understandable;
- the item is small enough to be completed within one Sprint;
- dependencies and open questions are known;
- technical and DevOps implications are considered;
- architectural impact is identified (if applicable).

The Definition of Ready supports **good Sprint Planning**.

It does **not** remove responsibility from the Product Owner or the team.
It helps you make better decisions together.

How you deliver work

Definition of Done

Work is only considered **Done** when the team agrees it meets the Definition of Done.

This includes:

- code that is readable and reviewed;
- automated tests that run as part of the pipeline;
- a successful build;
- deployment in the target environment (if applicable);
- predictable behaviour under failure conditions;
- clear logging and recovery or rollback options;
- updated documentation where needed;
- architectural decisions documented (if applicable).

"Done" means:

- ready to show in a Sprint Demo;
- ready to be used by others.

See the **Definition of Done Template** for the complete checklist.

Architecture and technical decisions

Architecture documentation

Significant architectural and technical decisions must be documented using **Architecture Decision Records (ADRs)**.

When to write an ADR:

- When choosing between multiple viable technical alternatives
- When making decisions that are difficult or expensive to change later
- When deviating from established patterns
- When decisions significantly impact non-functional requirements

ADR process:

1. Propose decision in ADR format
2. Discuss with team
3. Review by at least 2 team members
4. Accept and merge when consensus is reached

See the **ADR Template** for the format and structure.

Architecture governance

Some architectural decisions are **prescribed** by staff and are not negotiable (technology stack, security requirements, infrastructure constraints).

Other architectural decisions are **your responsibility** within those constraints (database schema, API design, component structure, implementation patterns).

Details on which decisions fall into which category will be shared during Sprint 1.

Scrum events

You work in **Sprints of two weeks** and use the Scrum framework.

The Scrum events are:

- Sprint Planning;
- Daily Scrum;
- Sprint Review (Demo);
- Retrospective.

The Scrum Master facilitates these events.
The Product Owner ensures clarity of goals and priorities.

All team members participate actively.

Professional expectations

In this project:

- problems and defects are normal;
- delivery issues may occur;
- not everything will go as planned.

What matters is **how you respond**:

- investigate calmly;
- communicate clearly;
- respect quality and agreed processes;
- make decisions as a team.

This project focuses on **professional behaviour**, not on avoiding mistakes.

First week checklist

In your first week, ensure you have:

- [ ] Access to GitHub repository
- [ ] Access to Jira project

- [ ] SSH access to TransIP VPS
- [ ] Microsoft Teams access
- [ ] Read the Sprint Guide for Interns
- [ ] Read the Definition of Ready template
- [ ] Read the Definition of Done template
- [ ] Read the Product Vision (including GDPR requirements)
- [ ] Understood your team role (Product Owner, Scrum Master, DevOps lead, or Developer)
- [ ] Participated in first Sprint Planning

For Product Owner specifically:

- [ ] Received GDPR requirements from staff/DPO
- [ ] Understand privacy by design principles

Final note

You are trusted with real responsibility.

Use that trust well:

- talk to each other;
- make decisions explicit;
- reflect and improve every Sprint.

Staff are here to **coach and challenge**, not to solve problems for you.

End of Intern Onboarding Guide v1.1