# Assignment 1. Deciding what to build

| Member | Role | Tasks |
|---|---|---|
| Alexandr Cherkasov | Backend | • Study the structure of the project;<br>• Audit Django models and REST v1;<br>• Document database schema;<br>• Prepare test fixtures. |
| Almir Avkhadiev | Backend, QA | • Provision and configure a staging VM;<br>• Deploy current version;<br>• Write deployment documentation;<br>• Review Django and PostgreSQL frameworks;<br>• Prepare weekly report |
| Anton Bugaev | Frontend | • Study the structure of the project repository;<br>• Consider how to adapt existing design to InnoNoHassle identity;<br>• Consider how to rewrite the frontend code part. |
| Dmitriy Ezovskikh | Frontend | • Study the structure of the project repository;<br>• Review React.js basics;<br>• Prepare a weekly report. |
| Timur Daminov | Team-Lead, Frontend | • Assign tasks to the team members;<br>• Prepare interview script;<br>• Held a meeting with the customer;<br>• Provid and build the VM. |

June 17, 2025

# Contents

# 1. Interview Script

## 1.1. Draft Questions

- What specific problem should the new frontend solve?
- Are there any tasks that are not included in the current release?
- Who will use the system (students, trainers, administrators, guests)?
- What permissions and scenarios for each role?
- What functionality can be removed?
- Is there an approved InNoHassle design system (colors, components, fonts)?
- Do we need to repeat the current appearance 1:1 or a redesign is fine?
- What models and entities are already in Django?
- What needs to be changed for the new REST version (filtering, pagination, serialization, versioning)?
- What type of authentication: JWT, session, cookie?
- Is it necessary to change the database? Do we need to backup/migrate existing data before the release?
- Deployment environment (Docker, k8s, bare-metal)?
- CI/CD (GitHub Actions, GitLab CI, ...)?
- Is E2E (Cypress/Playwright) required?
- Code review and linting standards (eslint, pre-commit)?
- Is documentation required for admins/developers?

## 1.2. Improvements Based on the Mom Test

- “What functionality can be removed?” changed to “What functionality do users rarely use or complain about?”. Instead of asking about the customer opinion, we are asking about real product problems that users complain about.
- “What needs to be changed for the new REST version (filtering, pagination, serialization, versioning)?” changed to “What specific issues have you faced with the current REST version?”. Again, we are asking about actual problems and efficiently avoiding opinions. Moreover, we are encouraging customer to give an open-ended answer.

## 2. Interview Notes

- The university sports portal frontend must be merged into the InNoHassle website.
- The colour palette must match InNoHassle, yet the overall layout may stay close to the current sports site.
- Legacy jQuery frontend is hard to maintain; a React rewrite is expected.
- The Django backend remains unchanged.
- A REST-like v2 API will be introduced, while v1 remains operational for compatibility.
- Development VM with Docker Compose is required; production will mimic it.
- Customer emphasised ease of onboarding new student developers and automated tests.

## 3. Interview Recording Link

https://drive.google.com/file/d/1UUZtRHp5qLYIcdUJ0dhH6MgEgEA8YShs/view?usp=sharing

## 4. Research Board Link

https://www.figma.com/design/PyaIBlw0FzgBOWt5v3SIlR/SportSite?node-id=0-1&t=QlW4deLPqXzDcA8K-1

## 5. Research of existing solutions

The main requirements from the customer are to migrate the functionality of the sports registration website into the association university microservices platform InnoNoHassle. Registration via the SwipeCalendar API is already implemented and actively used.

Based on the interview, it became clear that preserving the existing logic, improving maintainability, and adapting the interface to match the InnoNoHassle design system are key priorities. Therefore, we are not building the system from scratch but modernizing the proven solution using React and REST v2.

For this reason, analyzing five existing alternatives is not necessary: we already have a working system and clear requirements — our task is to adapt and improve, not to choose between different options.

| Product | Features | Notes |
|---|---|---|
| SwipeCalendar API | Scheduling, booking | Already integrated and actively used for sports section registration. It is necessary to recreate a similar functionality with minor changes. |
| InnoNoHassle | The association of university microservices | Requires adapting the frontend of the existing website to match the platform's style and layout. |

Table 1: Qualitative analysis.

# 6. Summary Report

## 6.1. Key Learnings from Interview

- Merge the sports site into InNoHassle with a unified brand.
- Replace jQuery with React while keeping the Django backend.
- Maintain REST v1 for legacy clients; build versioned REST v2 for the new UI.
- Provide a Docker-based development and test environment.

## 6.2. MVP Feature Roadmap

- **MVP v1:** Refactor and stabilise current functionality in React; achieve parity with the legacy UI.
- **MVP v2:** Integrate with InNoHassle SSO, design system and main navigation.
- **MVP v3:** Public release on the InNoHassle domain, full QA, and deprecate the old site.

## 6.3. Questions to be clarified

No questions to clarify for now.

## 6.4. Use of AI/LLMs

No AI or LLM were used during this week.

## 6.5. Next Steps

1. Finish exploring the repository structure and code base — Alexandr, Anton.
2. Finalise the task list in the issue tracker — Timur.
3. Draft a migration plan for REST v2 endpoints — Almir.
4. Prepare a proof-of-concept React page calling a real API — Dmitriy.
5. Validate the VM deployment script on a fresh host — Almir.
6. Schedule a design-system hand-off meeting with the customer — Timur.