

Assignment 3

25.06.2023

DSAI-03

Team 47: Managing IU events

Team


Full name	Innomail	Tasks
Ruslan Belkov	r.belkov@innopolis.university	Managing, Backend
Saveliy Lekhtin	s.lekhtin@innopolis.university	Backend
Mikhail Dudinov	m.dudinov@innopolis.university	Parser for sports
Daniil Nikulin	d.nikulin@innopolis.university	UI Design, Frontend
Artem Bulgakov	art.bulgakov@innopolis.university	Backlog, Frontend

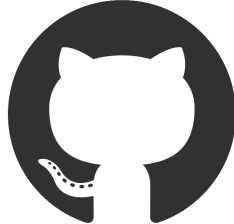
Project board

Product backlog and sprint backlog on one page:

InNoHassle • one-zero-eight

System for storing and managing events in Innopolis

 <https://github.com/orgs/one-zero-eight/projects/4/views/1>



We've learned how to use GitLab to manage project. But some of the features are not available in free version and GitLab has poor designed user interface (you should spend too much time to achieve something), so we preferred GitHub with its Projects and Issues.

Git process

We explored several approaches to organizing Git process. Some of them were too complicated for small teams (such as Git Flow) and they do not facilitate the rapid development of the project.

Also, our project is divided into components which are located in different repositories and have only few responsible people:

- InNoHassle-Events — Saveliy Lekhtin & Ruslan Belkov
- InNoHassle-Website — Artem Bulgakov & Daniil Nikulin
- InNoHassle-Parsers — Ruslan Belkov & Mikhail Dudinov

The complex branching models seemed ineffective, so we decided to follow Trunk-Based Development process. This method has worked well for small teams as well as large corporations (*Google uses a similar approach for their monorepository with 35000+ developers*).

The Trunk-Based Development describes two branching models: Committing straight to the trunk and Short-lived feature branches. The first model is best for small teams. The second model is best for large teams and for situations where an implementation requires special attention from other developers. That's why we decided to commit straight into the main branch and use short-lived branches only for special cases.

It means:

- Every developer implements a feature locally.
- Every developer tests changes locally.
- If a feature is finished and the code is stable enough (tested by the developer locally), a developer pushes changes into main branch.
- If a feature is large or a developer wants other people to check the code, the developer pushes changes into a separate branch.
- After pushing any commits into main branch, other developers responsible for the repository check the code (they may do it during the day).
- Main branch is always considered stable (because developers tested the code locally).
- GitHub's Pull requests mechanism is only used when short-lived branch requires special attention from other developers. For complex features, if there is possibility to meet with the person responsible for the repository, they meet to discuss the solution and check code (similar to pair programming).

We also chose the convention for commit messages — Conventional commits.

Sprint

As the Scrum framework suggests, we have planned to conduct four types of events for the Sprint itself:

1. Sprint Planning
2. Daily Scrum
3. Sprint Review
4. Sprint Retrospective

Sprint Planning

Right after meeting with the DoE representative, Alex Potemkin, we began planning current sprint. We defined its

- Goal — users should be able to choose favorite calendars and view them on one page.
- Backlog — in our case it's epic issues based on the features from the product backlog.
- Increments — subtasks obtained by dividing epics that have reached a **Done** state.

Deadline


Until Wednesday, 28.06.2023 (7 days after sprint planning).

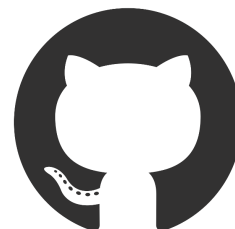
Backlog

Sprint #1 backlog:

InNoHassle • one-zero-eight

System for storing and managing events in Innopolis

 <https://github.com/orgs/one-zero-eight/projects/4/views/2>



Milestone

On GitHub every repository has to have separate milestones:

- [InNoHassle-Events](#)

- InNoHassle-Website
- InNoHassle-Parsers

Daily Scrum

We meet almost every day to inspect the work done and to adjust it if necessary. This allowed us to synchronize our work, and not to perform unnecessary tasks.

Sprint Review

Outcome

As for 25.06.2023, we have implemented authentication on backend side and connection to database to store users data.


Other features are in progress.

Sprint Retrospective

Will be after sprint

Sprint Future

Sprint Review Sprint Retrospective

 https://docs.google.com/document/d/1tbz2UDz0mBaNY7qMMFyYRLf1HJOp_BgoS-ywzXS9kc0/edit#heading=h.gh5ban7xah6a