

Assignment 5

09.07.2023

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	Frontman
Daniil Nikulin	d.nikulin@innopolis.university	UI Design, Frontend
Artem Bulgakov	art.bulgakov@innopolis.university	Backlog, Frontend

Sprint Planning

Sprint Goal

The goal of this sprint was

- improve the website's user experience (UX/UI);
- enhance the API for further development;
- explore the testing environment.

Deadline


09.07.2023 (7 days after sprint planning).

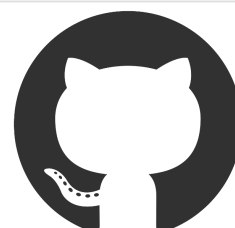
Backlog

Sprint #3 backlog:

InNoHassle • one-zero-eight

System for storing and managing events in Innopolis

 <https://github.com/orgs/one-zero-eight/projects/4/views/1?filterQuery=milestone%3A%22Sprint+3%22>



Acceptance Criteria

Export schedule to the calendar app

Given: The user is on the schedule page that includes a description and a link to the .ics file. The user has access to the Internet and has a calendar app that supports importing calendars by URL.

When: The user copies the provided link and imports the schedule by pasting the link in their calendar app.

Then: The calendar app successfully imports the schedule and displays it to the user. The schedule is accurately synchronized with the user's calendar, including all events, dates, and descriptions. The system responds to the calendar app with the actual .ics file, ensuring seamless integration and future updates to the schedule.

Authenticate through Innopolis SSO

Given: The user is on the main web page and has access to the Internet. The user possesses valid Innopolis SSO credentials.

When: The user presses the "Sign In" button and logs in via the Innopolis SSO.

Then: The system gets status about user authentication from Innopolis SSO. If the user successfully passed logging in stage, the user is granted access to the system's features and resources.

View schedule using a popup

Given: The user is logged in and currently viewing the schedule page.

When: The user clicks on a specific schedule item.

Then: A responsive and visually appealing popup window opens, providing an intuitive and comprehensive display of the selected schedule details. The popup window includes events and view modes. The user can easily navigate or close the popup window to return to the schedule page.

View schedules on the Dashboard page

Given: The user is logged in and currently on the dashboard page.

When: The user navigates to the "Your Schedules" section of the dashboard.

Then: The dashboard page presents a well-structured and easily scannable list of schedules, prominently displaying key information for each schedule. The user can

quickly identify and access their desired schedule by visually scanning the list, improving their overall experience and efficiency.

Add schedule to favorites

Given: The user is logged in and currently on the schedule page.

When: The user clicks on the "Add to Favorites" button associated with a specific schedule item.

Then: The selected schedule item is instantly added to the user's favorites list, ensuring easy access to frequently used or important schedules. The "Add to Favorites" button visually indicates that the schedule item has been successfully added, providing feedback to the user. The user can manage their favorites list, enabling them to prioritize and personalize their schedule viewing experience.

Hide schedule from view on the Dashboard page

Given: The user is logged in and currently on the dashboard page.


When: The user chooses to hide a specific schedule from their view using an available option or toggle.

Then: The selected schedule is immediately hidden from the dashboard page, removing it from the list of visible schedules. This customization feature allows the user to tailor their dashboard view to focus only on the schedules they need or prefer. The hidden schedule remains accessible and can be easily unhidden or managed through appropriate settings or filters, providing flexibility and control to the user.

CI/CD Implementation


Actions · one-zero-eight/InNoHassle-Events


API of InNoHassle Events. Contribute to one-zero-eight/InNoHassle-Events development by creating an account on GitHub.


 <https://github.com/one-zero-eight/InNoHassle-Events/actions>


one-zero-eight/
InNoHassle-Events


API of InNoHassle Events




 3
Contributors

 11
Issues

 2
Stars

 0
Forks



For our CI/CD pipeline, we have used GitHub Actions. The pipeline consists of two stages: pre-commit and testing.

GitHub Actions Workflow definitions:

<https://github.com/one-zero-eight/InNoHassle-Events/tree/main/.github/workflows>

Pre-commit Stage

In the pre-commit stage, we have integrated the following tools:

- **Ruff**: A linter that helps ensure consistent code style and formatting.
- **Black**: A code formatter that automatically formats the code according to the Python style guide.
- **Pre-commit**: A tool to run all above tools. Developers also install pre-commit locally to ensure that changes pass Ruff and Black.


These tools run on every push, ensuring that the code adheres to the defined coding standards.

Testing Stage

In the testing stage, we have integrated the following tools:

- **Pytest**: A testing framework for Python. We have written unit tests using Pytest to verify the functionality of our code.
- **Coverage**: A tool for measuring code coverage. We have generated coverage reports to track the percentage of code covered by our unit tests.

The testing stage runs automatically on every push and provides us with feedback on the quality and coverage of our code. We have implemented unit tests (only for API) that cover critical functionalities of our codebase, ensuring its correctness and reliability. The feedback is provided in the form of comment to commit on GitHub by bot.



github-actions bot commented on b77eae2 2 minutes ago

...





Coverage 74%


▼ Coverage Report

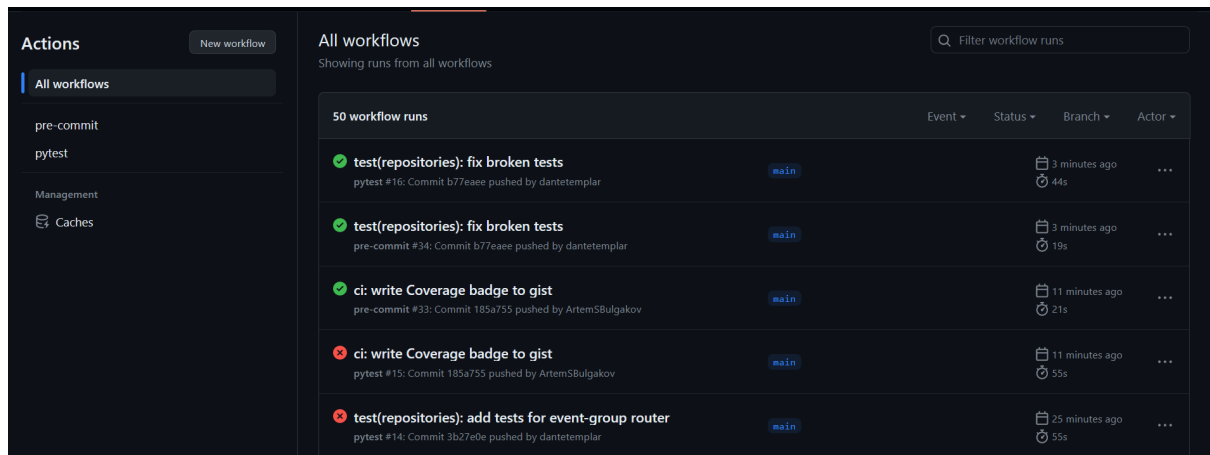
File	Stmts	Miss	Cover	Missing
src				
dev.py	3	3	0%	1-4
exceptions.py	22	6	73%	7, 16, 24, 32, 40, 52
main.py	60	1	98%	113
src/app				
dependencies.py	39	5	87%	15-17, 35, 43
src/app/auth				
common.py	28	17	39%	12-21, 25-32, 36-44, 49-50
dependencies.py	16	6	62%	33-38, 42
dev.py	28	16	43%	18-47
innopolis.py	36	18	50%	27-57
jwt.py	45	29	36%	25-27, 31-35, 39-43, 47-51, 55-63, 67-74
src/app/event_groups				
routes.py	20	10	50%	22-26, 43-48, 63-64
src/app/users				
routes.py	34	20	41%	35-37, 56-61, 79-81, 101-106, 110-120
src/repositories/event_groups				
abc.py	31	9	71%	7, 15, 19, 23, 27, 31, 35, 39, 43
repository.py	69	17	75%	24-31, 45, 48-75
src/repositories/users				
abc.py	31	9	71%	7, 15, 19, 23, 27, 31, 35, 39, 43
repository.py	69	17	75%	24-31, 45, 48-75

Showing 1 changed file with 12 additions and 91 deletions.

json_repository.py	67	21	69%	50, 60, 74-76, 80-88, 92-100
sql_repository.py	68	15	78%	102-120, 123-136
src/schemas				
event_groups.py	43	3	93%	22, 40, 69
users.py	27	1	96%	32
src/storages/sql				
storage.py	34	4	88%	10, 14, 18, 30
src/storages/sql/models				
event_groups.py	27	1	96%	9
users.py	17	1	94%	9
TOTAL	817	212	74%	

Tests	Skipped	Failures	Errors	Time
12	0 	0 	0 	2.784s 





Sprint Review

What considered as *Increment*:

- interface of the site is examined for non-obvious problems, these problems have been fixed;
- the API has been improved to meet the needs of the website;
- (partially) repositories for the tagging system is prepared.

Sprint Retrospective

What went well

- The integration of pre-commit tools (Ruff and Black) helped maintain consistent code style and formatting throughout the project.
- The implementation of unit tests using Pytest improved the overall code quality and provided confidence in the functionality.
- The CI/CD pipeline ensured that the code was tested and validated on every push, reducing the risk of introducing bugs.

Challenges encountered

- We faced unexpected behavior while using the Python pytest library in asynchronous mode. This required additional troubleshooting and adjustments in our testing approach.
- Applying Test Driven Development (TDD) to the development of the website's user interface (UI) was challenging. It is difficult to anticipate and write tests for UI-related aspects that focus on user experience and visual appeal.