# Багтрекер

Реализовать упрощённую версию таск-трекера наподобие jira или redmine.

В трекер можно добавлять задачи. Каждая задача содержит следующие поля для заполнения (обязательные отмечены *****):

- Номер (генерируется автоматически)*
- Тип (bug или task)*
- Приоритет (critical/high/medium/low)
- Статус*
- Заголовок*
- Описание
- Исполнитель
- Создатель*
- Дата и время создания (генерируется автоматически)*
- Дата и время последнего изменения (генерируется автоматически)*

Также в каждой задаче можно указать, какие задачи она блокирует.

Реализовать простую регистрацию пользователей (логин + пароль). Каждый может иметь одну из 4 возможных ролей (менеджер, тимлид, разработчик, тест-инженер). Менеджер должен иметь возможность управлять пользователями (менять логин и роль).
Статус задачи может иметь следующие значения:

- To do
- In progress
- Code review
- Dev test
- Testing
- Done
- Wontfix

Переход задачи между статусами от To do до Done возможен только в указанном порядке. Переход задачи в статусы To do и Wontfix возможен из любого статуса.

Комбинация полей статус + исполнитель имеет следующие ограничения:

- Менеджер никогда не может быть назначен исполнителем задачи.
- Тимлид может быть исполнителем при любом статусе
- При любом статусе кроме in progress исполнитель может быть не указан
- Статусы in progress, code review, dev test не могут иметь исполнителем тест-инженера
- Статус testing не может иметь исполнителем разработчика.

(При конфликте ограничений приоритетнее то, которое указано выше в списке).

Реализовать REST-эндпоинты для:

- Регистрации пользователей, смены пароля
- Управления пользователями для менеджера
- Просмотра задачи. Должны отображаться все поля, ссылки на подзадачи, а также блокирующие и блокируемые задачи.
- CRUD-операций над задачами (удаление только для менеджера), просмотра списка задач
- Изменения статуса задачи с обновлением исполнителя
- Поиска задач по тексту (в заголовке и описании) или номеру. Сначала показывать недавно обновленные.

Подготовить для демонстрации работы примитивный UI или http-запросы в idea/postman/swagger. Для реализации можно использовать любой Python Фреймворк и PostgreSQL.

## Будет плюсом:

- Юнит-тесты
- Swagger-UI
- Dockerfile и docker-compose.

## Бонусные задачи:

- Добавить в фильтр при поиске поля тип, статус, создатель и исполнитель
- Предусмотреть произвольную сортировку в результатах поиска и списке задач
- Сохранять и показывать историю изменений задачи
- Добавить возможность создавать подзадачи с неограниченной вложенностью

## Время выполнения работы:

5-8 дней