Тестовое задание на должность php backend-разработчика

Необходимо на языке php реализовать приложение по принципам **RESTful** API. Для СУБД можно использовать SQLite, MySQL или PostgreSQL.

В тестовом задании представлены обязательные и дополнительные пункты. Чтобы тестовое задание считалось выполненным, необходимо реализовать обязательные пункты. Дополнительные пункты можно делать по желанию в любом количестве, их выполнение будет говорить о вашем профессиональном уровне. В соответствии со сложностью выполнения заданий нами будут оценены ваши компетенции и выдвинуто соответствующее зарплатное предложение.

Необходимо представить списочно, какие, из перечисленных в тестовом задании пунктов были вами реализованы.

• Обязательные пункты (уровень 1):

1. Составить инструкцию по **установке** и **настройке** приложения.
2. Составить справку по всем методам в API (url, список параметров в запросе и ответе) или создать коллекцию запросов в программе Postman и экспортировать её в json-формате.
3. Реализовать метод для регистрации нового пользователя.
4. Реализовать метод для аутентификации пользователя.
5. Все методы, кроме регистрации и аутентификации, должны работать только для авторизованных пользователей.
6. Реализовать CRUD-действия (создание/чтение/редактирование/удаление) для одной модели и получение всех записей, относящихся к данной модели.
7. Реализовать валидацию полей при создании и редактировании записи.
8. Реализовать сортировку, пагинацию и фильтры для данной модели.

• Дополнительные пункты (уровень 2):

1. Сделать возможность получения в ответе только тех атрибутов, которые указаны в запросе (например, ?fields=id,name).
2. Создать вторую модель и добавить связи "one-to-many" и "many-to-many" (методы в соответствующих моделях).
3. Реализовать загрузку и привязку изображений для какой-либо записи.
4. Использовать сторонний фреймворк (например, Yii, Laravel, Lumen, Symfony, Silex и др.).
5. Использовать менеджер зависимостей Composer для сторонних зависимостей, если они присутствуют в проекте.
6. Сделать создание таблиц в базе данных через миграции.
7. Сделать разные HTTP статус коды в ответах от API (400, 403, 405 и др.).
8. Загрузить приложение в репозиторий на Github.

• Дополнительные пункты (уровень 3):

1. Сделать возможность получения в ответе связанных данных из других моделей, которые указаны в запросе (например, ?expand=category).
2. Обязательно использовать фреймворк Yii 2.0 и его внутренний функционал.
3. Использовать миграции или сидеры для наполнения таблиц в базе данных тестовыми данными.
4. Сделать проверку при загрузке изображений, чтобы исключить дублирование идентичных файлов, а также ограничение на разрешение изображений.
5. Сделать контроль доступа на основе ролей, например, чтобы получение данных было доступно всем пользователям, а все остальные методы только администратору.
6. Реализовать кэширование данных из базы данных.
7. Реализовать создание нового пользователя через консоль.
8. Добавить в проект тесты (unit, functional).
9. Сделать возможность развёртывания приложения для разных окружений ("development" и "production"). В окружении "development" должно быть логирование входящих параметров в запросах на изменение данных.

• Примерный формат справки по методам API.

