Generate a complete Java 17 REST API project for a ToDo application using Spring Boot 3. The application should support full CRUD operations for tasks. Use clean architecture with layered structure (controller, service, repository, model, dto, and mapper). Include the following:

1. Use Spring Boot 3 with Maven as the build tool.
2. Create a `Task` entity with fields:
    - id (Long, auto-generated)
    - title (String, required)
    - description (String, optional)
    - status (Enum: PENDING, IN_PROGRESS, COMPLETED)
    - createdAt (LocalDateTime, auto-set)
    - updatedAt (LocalDateTime, auto-set on update)
    - dueDate (LocalDateTime, required)
3. Provide a `TaskController` with REST endpoints:
    - GET `/api/tasks` - list all tasks
    - GET `/api/tasks/{id}` - get task by ID
    - GET `/api/tasks?status=COMPLETED` - filter by status
    - POST `/api/tasks` - create task
    - PUT `/api/tasks/{id}` - update task
    - DELETE `/api/tasks/{id}` - delete task
    - PATCH `/api/tasks/{id}/done` - mark task as completed
4. Use DTOs for request and response. Map using MapStruct (or do it manually if needed).
5. Implement input validation using Jakarta Bean Validation annotations (e.g., @NotBlank, @Future).
6. Handle exceptions with `@ControllerAdvice` and return meaningful error responses.
7. Use JPA/Hibernate with in-memory H2 database for development.
8. Auto-generate OpenAPI/Swagger docs using springdoc-openapi and expose Swagger UI.
9. Add unit tests for the service layer using JUnit 5 and Mockito.
10. Add integration tests for the controller using `@SpringBootTest` or `@WebMvcTest`.

Project should be self-contained and runnable with `mvn spring-boot:run`.

Optional: Use Lombok to reduce boilerplate (e.g., @Getter, @Setter, @Builder).