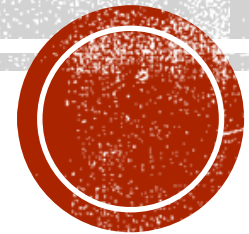


# BOOKLY

midterm presentation

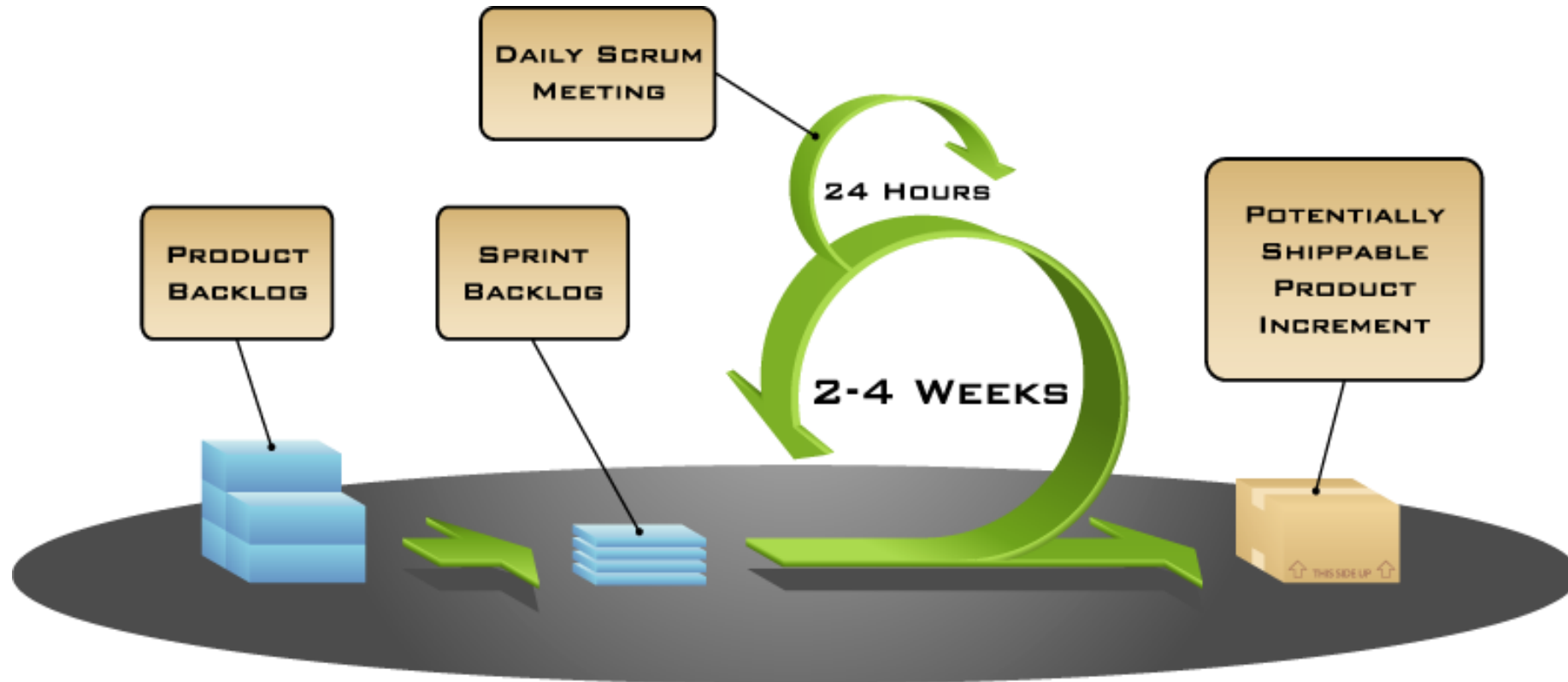


# OUTLINE

- Project Management
  - Scrum and YouTrack
  - Burn-Down-Chart
  - RUP
  - Extend of functionality
  - Use Cases for Mockup
- Architecture
- Automation
- Demo



# SCRUM AND YOUTRACK



# RUP- RATIONAL UNIFIED PROCESS

- **Alexandra**

- **Requirements Specifier:** Describe and phrase the features / design of the website with the team
- **Project Manager:** Phrase acceptance criteria of the features + Coordinate sprints + Manage YouTrack + Assign Tags

- **Jeanne**

- **Implementer:** Create database structure, logic, and interfaces (API)
- **Tester:** Test the interfaces (APIs) between Frontend, Backend and Database and the functionality of the Frontend

- **Nico**

- **Deployer:** Host domain and integrate CI (and Docker)
- **Designer:** Implement the design described in the Use-Cases + Web tests



# SCRUM AND YOUTRACK

YT

Issues | Dashboards Agile Boards Reports Projects

Create

SA

?

bookly Project Develop... Filter cards on the board

Sprint 1 14 — 28 Nov

S XL TV

Open In Progress 0m Code Review 0m Testing 0m Done 1w2d1h10m

Uncategorized Cards 14 cards

BOOKLY-45 Update use case diagram

JH Normal Task Not estimated 10m

BOOKLY-14 create Read.md

JH Normal Task Not estimated 1h



# SCRUM AND YOUTRACK

[Issues](#) ▾[Dashboards](#)[Agile Boards](#)[Reports](#) ▾[Projects](#)[New Issue](#) ▾[Everything](#) ▾

sort by: Updated desc



Project ▾

Assignee ▾

Priority ▾

State ▾

Tag ▾

+ More filters

## PROJECTS

Get one-click access to issues in your favorite projects — open this list to find a favorite

## > SAVED SEARCHES

## > TAGS

1 ▾



3 ▾

17 ▾



Matches 89 issues

Sorted by Updated ▾

S



L

**BOOKLY-85** [integrate video in some blog entry](#)

● Normal

Task

In Progres

stober.alex

Sprint 2

Not estim: 5h

2h 30m

● Requirer

● Elaborat

JHelm

19:14

**BOOKLY-98** [Fix Deploy Bug at Alex' IDE](#)

● Normal

Bug

Done

stober.alex

Sprint 2

3h

5h

● Configur

● Elaboratic

JHelm

Yesterday

**BOOKLY-97** [MidtermGC.md Overview](#)

● Normal

Task

Done

stober.alex

Sprint 1

Not estim: 2h 30m

2h 25m

● Requirer

● Elaborat

stober.alexandra

Yesterday

**BOOKLY-20** [create Use Case Diagrams for each scribble](#)

● Minor

User Story

Done

stober.alex

Unschedu

Not estim: 5h

4h 30m

● Analysis

● Inceptio

JHelm

Yesterday

2

**BOOKLY-89** [Update DB Scheme](#)

● Normal

Task

Done

JHelm

Sprint 2

Not estim: 20m

15m

● Analysis

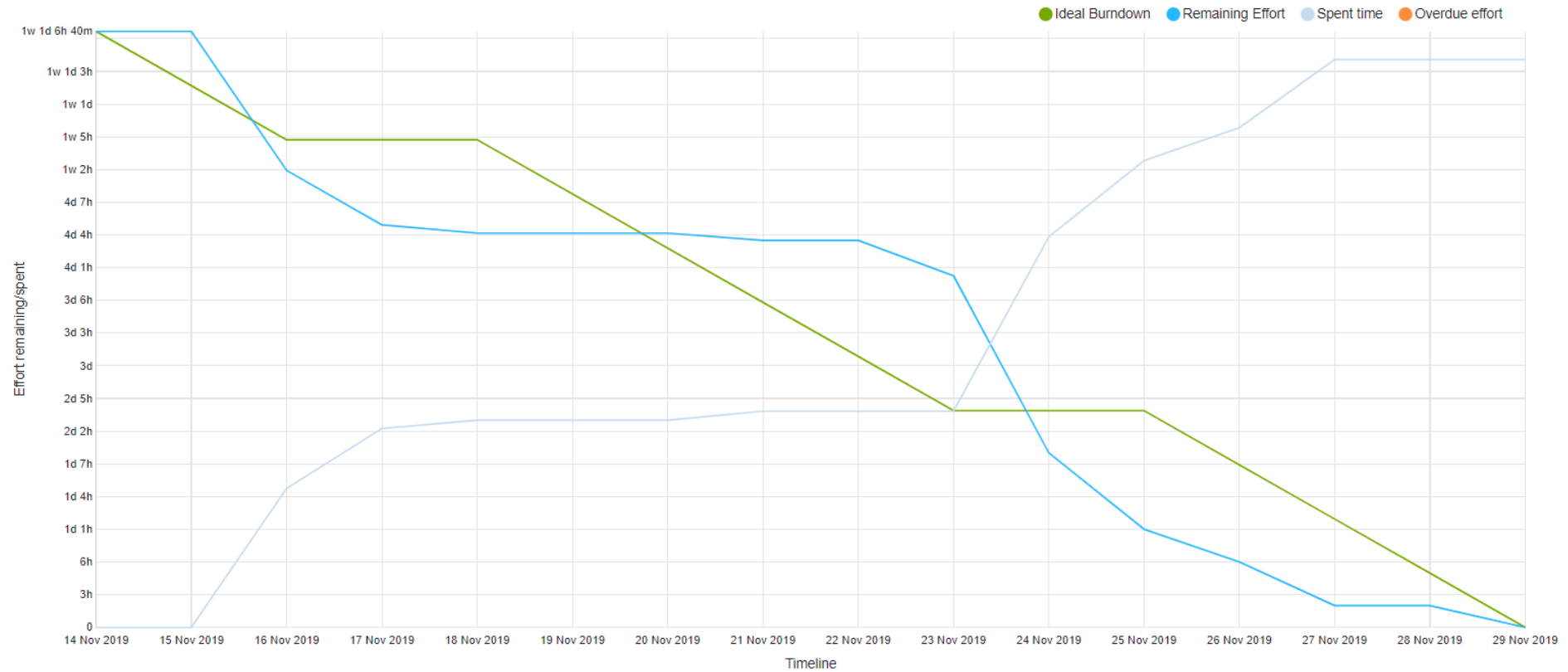
● Elaborat

JHelm

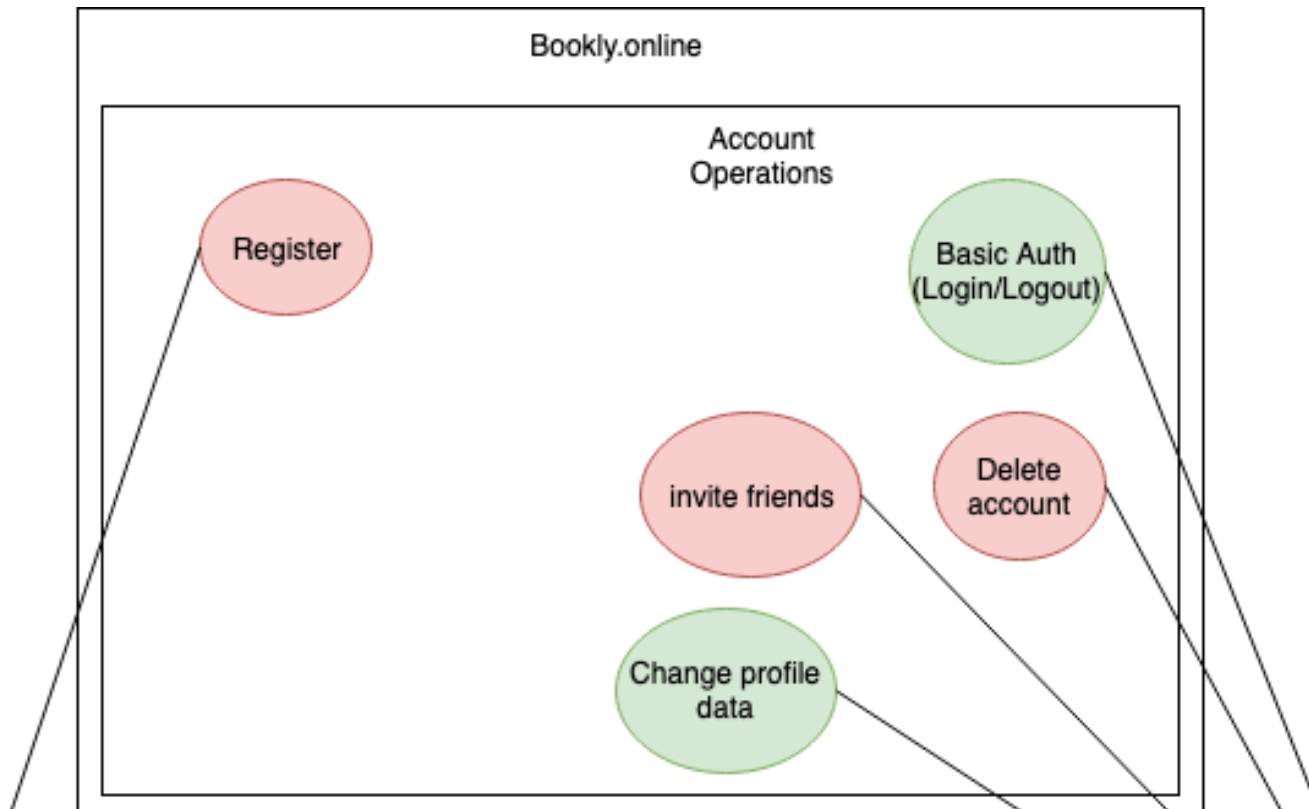
Yesterday



# BURN-DOWN-CHART

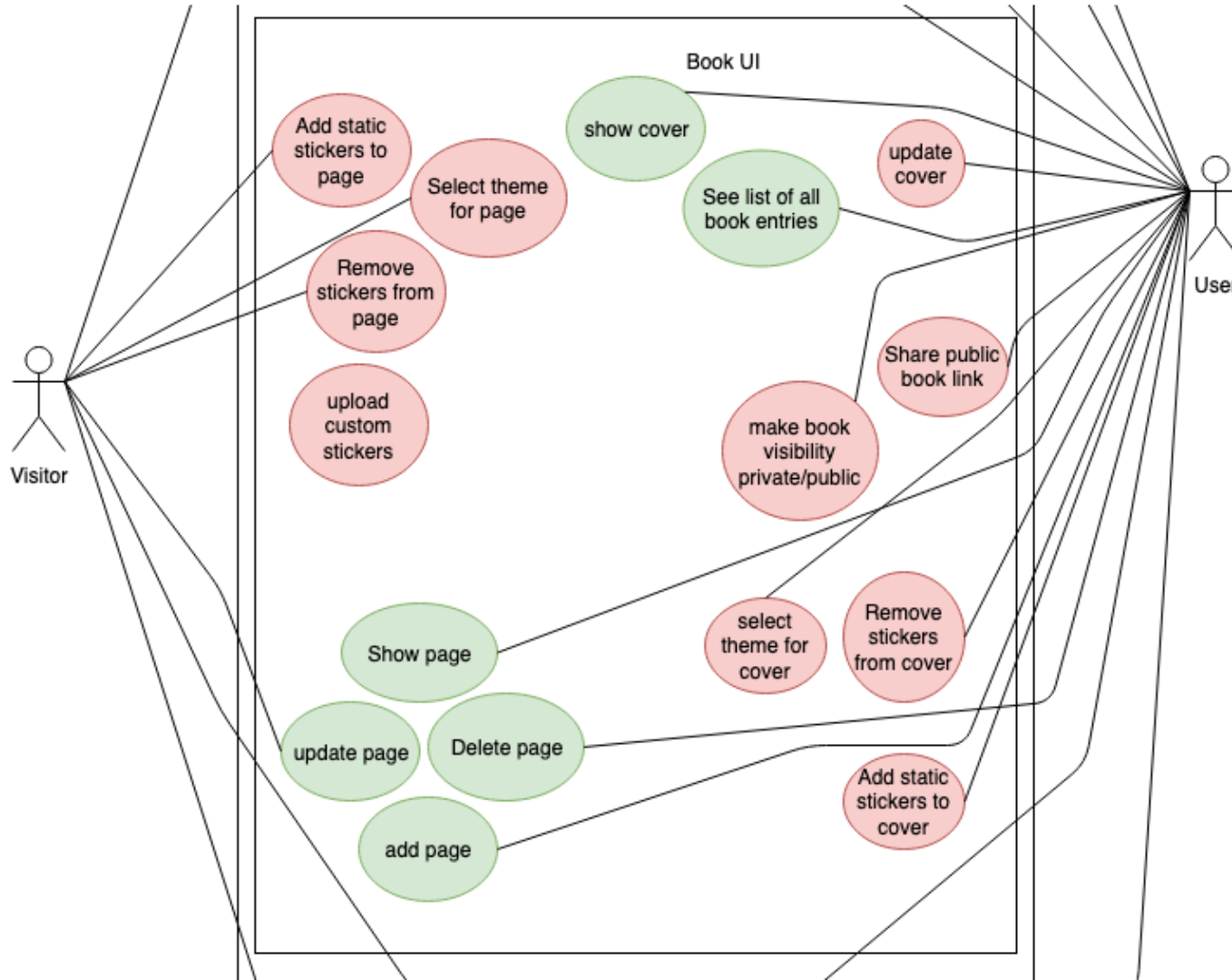


# EXTEND OF FUNCTIONALITY





# EXTEND OF FUNCTIONALITY



# USE CASES FOR MOCKUP

- CRUD: Manage page:
  - C: create book entry - S E
  - R: read an entry - S E
  - U: update/edit an entry - S E
  - D: delete an entry - S I
- CRUD: Manage Book:
  - C: create a book - S E
  - R: show all entries & cover - S E
  - U: update the cover - X I
  - D: delete the whole book - S I

Legend:

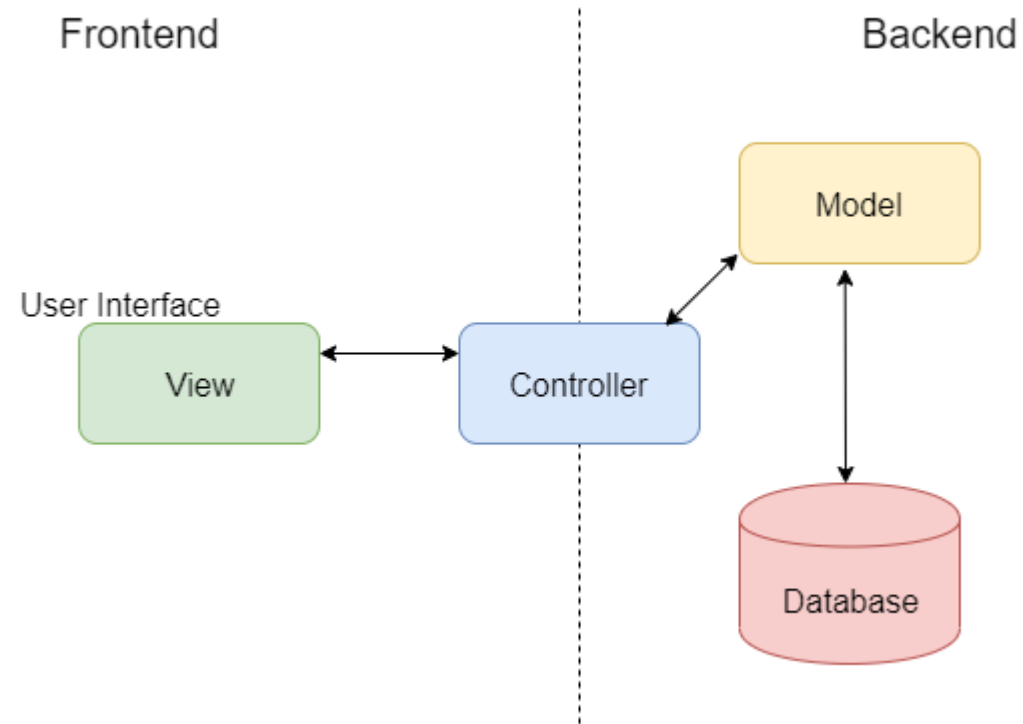
S=Scope; X = out of scope;

I = Included in CRUD (not counted as UC)

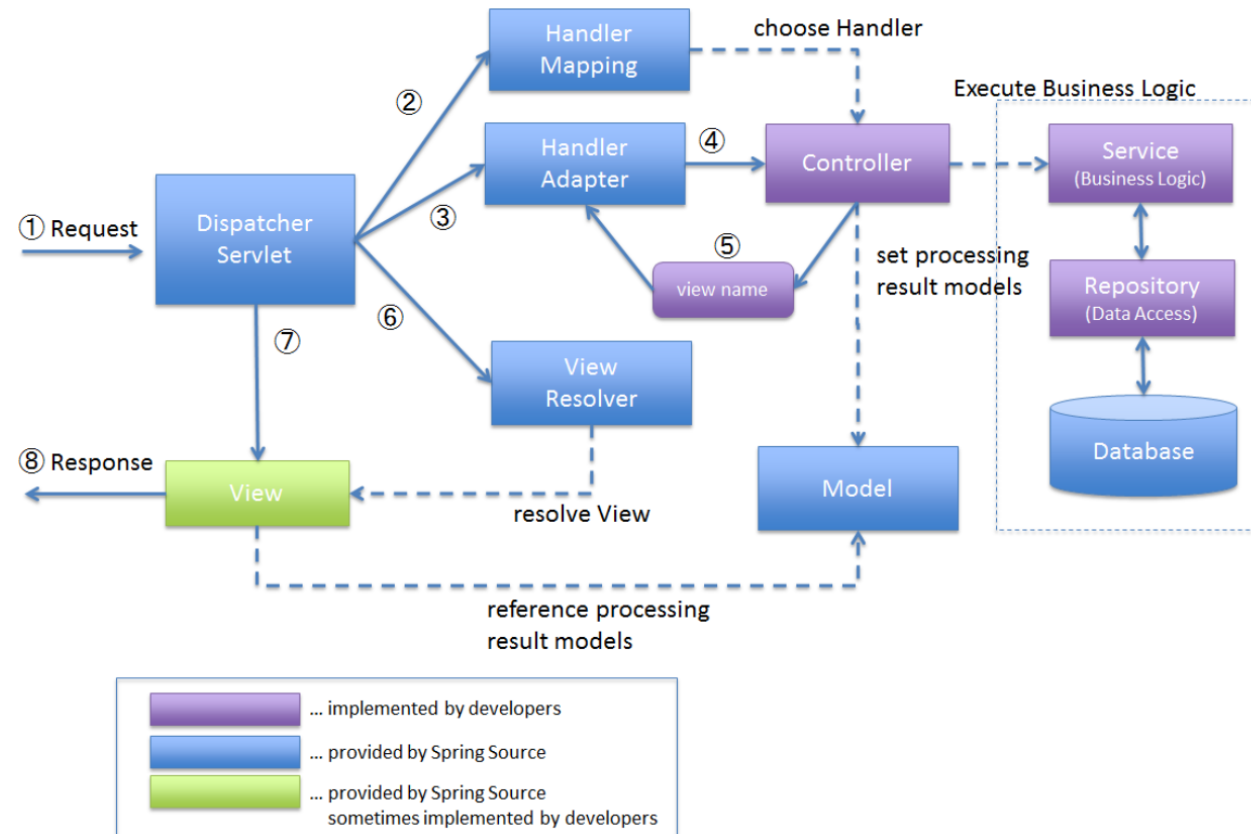
E = Excluded (external UC document -> counted as UC)



# ARCHITECTURE - OVERVIEW



# ARCHITECTURE — OVERVIEW

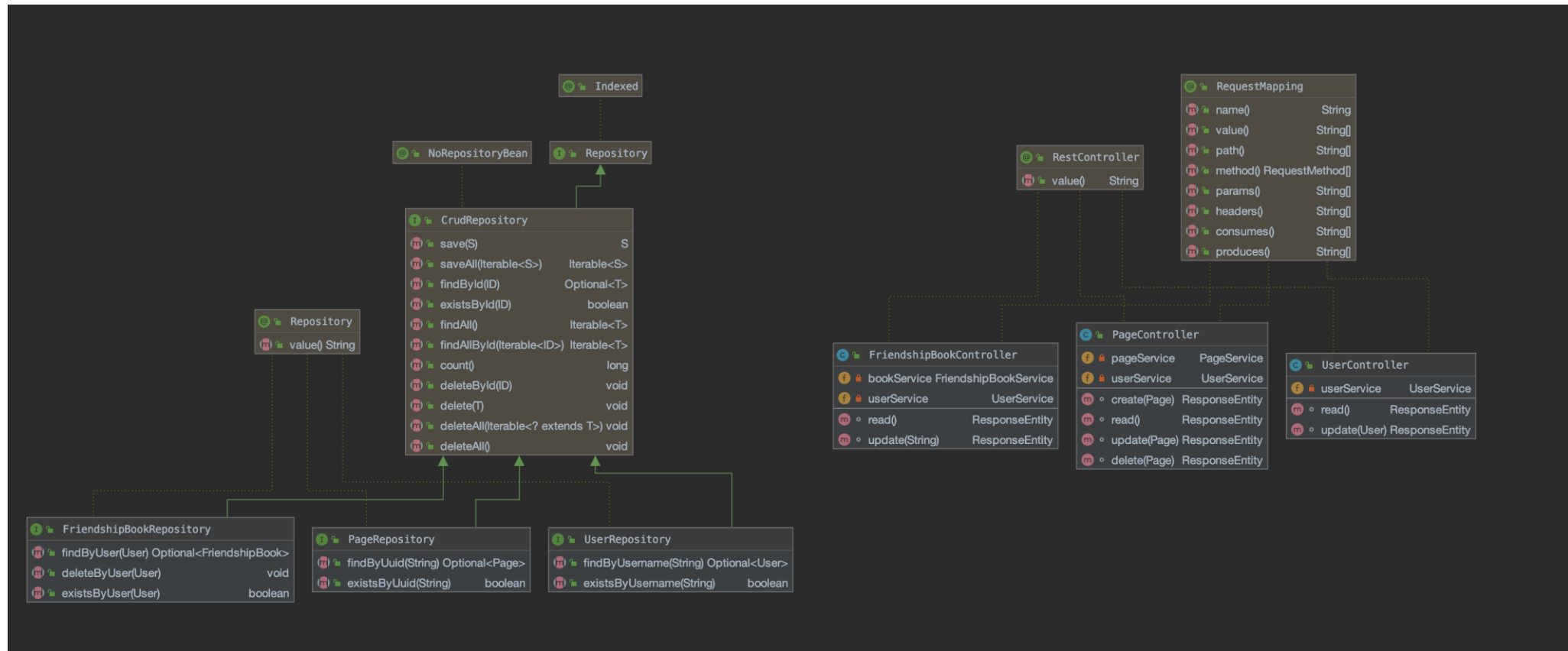


The screenshot displays a Java project structure in IntelliJ IDEA. The main classes and their attributes are as follows:

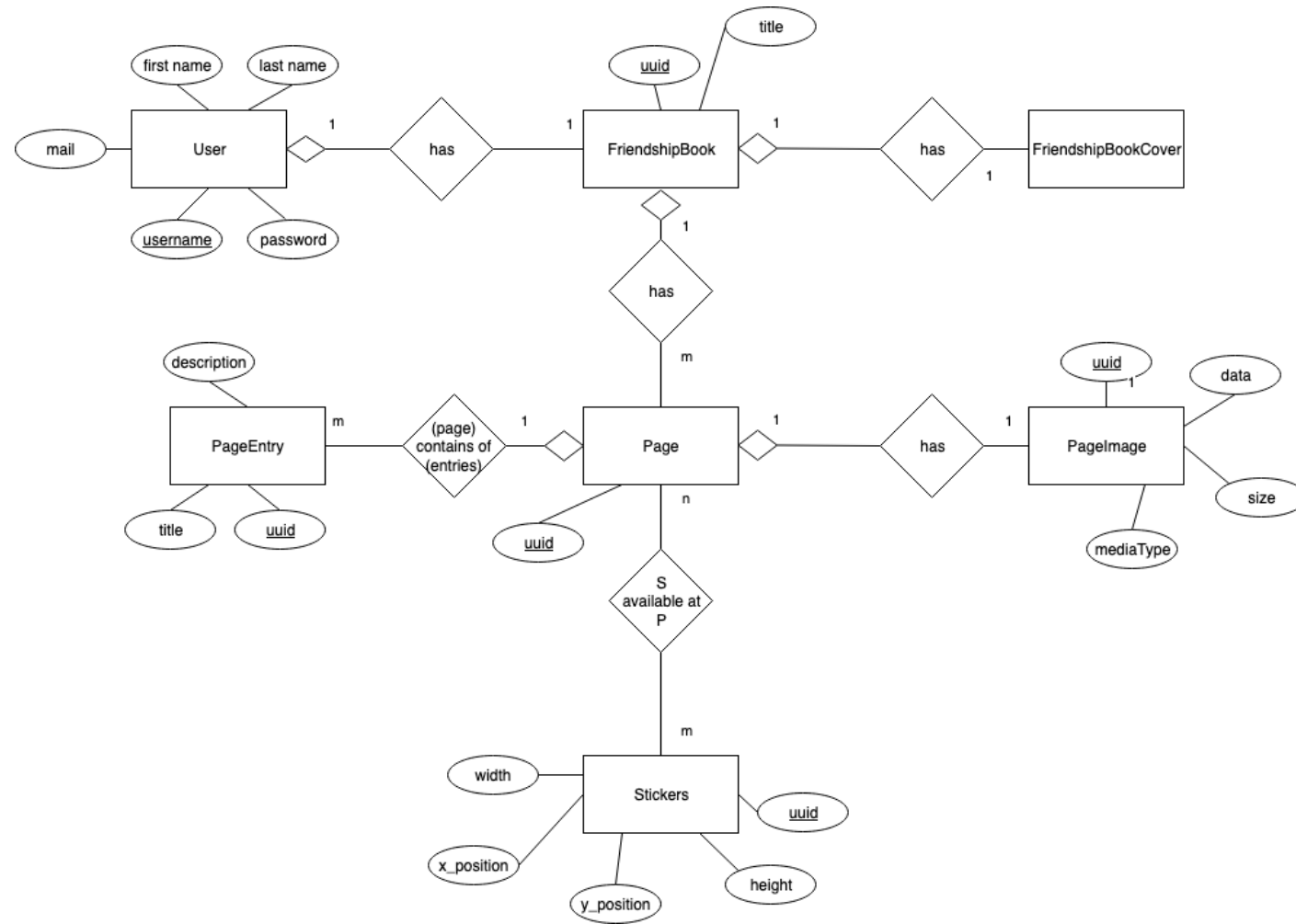
- AllArgsConstructor**:
  - staticName() String
  - onConstructor() AnyAnnotation[]
  - access() AccessLevel
- Entity**:
  - name() String
- Data**:
  - staticConstructor() String
- NoArgsConstructor**:
  - staticName() String
  - onConstructor() AnyAnnotation[]
  - access() AccessLevel
  - force() boolean
- Builder**:
  - builderMethodName() String
  - builderMethodName() String
  - builderClassName() String
  - toBuilder() boolean
  - access() AccessLevel
- AnyAnnotation**:
  - field() String
  - method() String
  - isStatic() boolean
- Default**:
  - Default
- User**:
  - username() String
  - mail() String
  - User(String, String) User
  - builder() UserBuilder
  - getUsername() String
  - getMail() String
  - setUsername(String) void
  - setMail(String) void
  - equals(Object) boolean
  - hashCode() int
  - toString() String
- UserBuilder**:
  - username(String) UserBuilder
  - mail(String) UserBuilder
  - build() User
  - toString() String
- FriendshipBook**:
  - uid() String
  - title() String
  - user() User
  - pages() List<Page>
  - FriendshipBook(String, String, User, List<Page>) FriendshipBook
  - builder() FriendshipBookBuilder
  - getUid() String
  - getTitle() String
  - getUser() User
  - getPages() List<Page>
  - setUid(String) void
  - setTitle(String) void
  - setUser(User) void
  - setPages(List<Page>) void
  - equals(Object) boolean
  - hashCode() int
  - toString() String
- FriendshipBookBuilder**:
  - uid() String
  - title() String
  - user() User
  - pages() List<Page>
  - FriendshipBookBuilder() FriendshipBookBuilder
  - uid(String) FriendshipBookBuilder
  - title(String) FriendshipBookBuilder
  - user(User) FriendshipBookBuilder
  - pages(List<Page>) FriendshipBookBuilder
  - build() FriendshipBook
  - toString() String
- Page**:
  - uid() String
  - entries() List<PageEntry>
  - Page(String, List<PageEntry>) Page
  - builder() PageBuilder
  - getUid() String
  - getEntries() List<PageEntry>
  - setUid(String) void
  - setEntries(List<PageEntry>) void
  - equals(Object) boolean
  - hashCode() int
  - toString() String
- PageBuilder**:
  - uid(String) PageBuilder
  - entries(List<PageEntry>) PageBuilder
  - build() Page
  - toString() String
- PageEntry**:
  - uid() String
  - title() String
  - description() String
  - PageEntry(String, String, String) PageEntry
  - builder() PageEntryBuilder
  - getUid() String
  - getTitle() String
  - getDescription() String
  - setUid(String) void
  - setTitle(String) void
  - setDescription(String) void
  - equals(Object) boolean
  - hashCode() int
  - toString() String
- PageEntryBuilder**:
  - uid(String) PageEntryBuilder
  - title(String) PageEntryBuilder
  - description(String) PageEntryBuilder
  - build() PageEntry
  - toString() String
- Service**:
  - value() String
- UserService**:
  - userRepository() UserRepository
  - authenticationService() AuthenticationService
  - friendshipBookService() FriendshipBookService
  - exists(String) boolean
  - getUser() User
  - update(User) User
  - create(String) User
- FriendshipBookService**:
  - friendshipBookService() FriendshipBookService
  - friendshipBookRepository() FriendshipBookRepository
  - pageRepository() PageRepository
  - log() Logger
  - read(User) List<Page>
  - add(User, Page) List<Page>
  - getPage(String) Page
  - delete(User, Page) List<Page>
  - update(User, Page) List<Page>
  - setMissingUidsToPageEntries(Page) void
- SpringBootApplication**:
  - exclude() Class<?>[]
  - excludeName() String[]
  - scanBasePackages() String[]
  - scanBasePackageClasses() Class<?>[]
  - proxyBeanMethods() boolean
- StartBootApplication**:
  - main(String[]) void



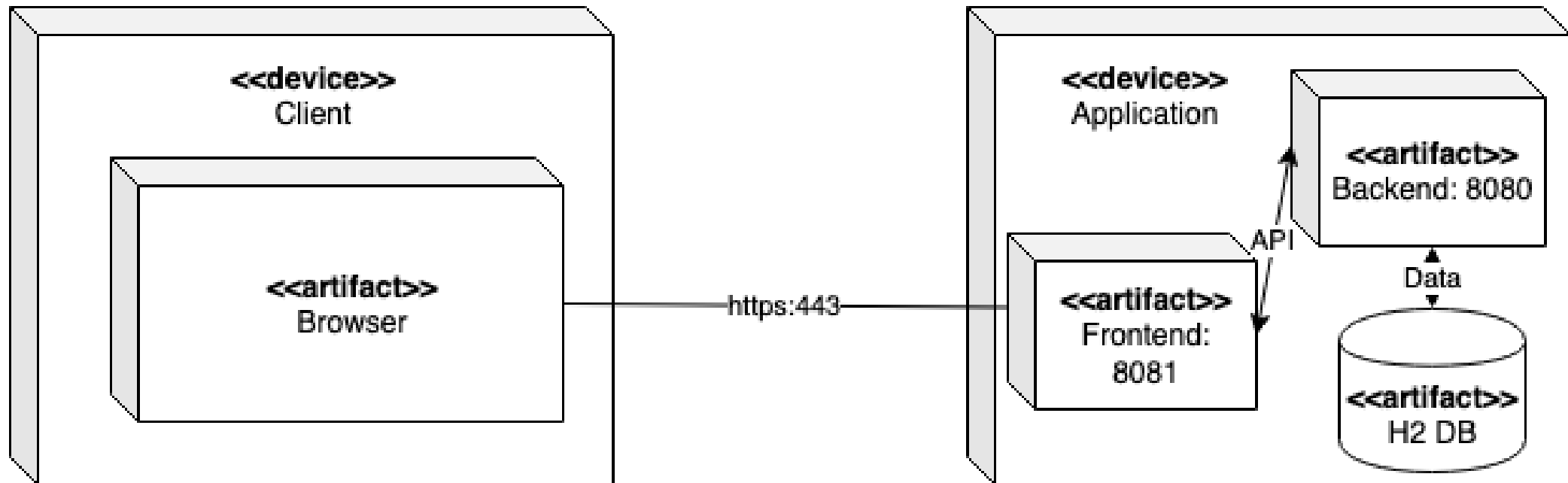
# CLASS DIAGRAM



# ARCHITECTURE – DATA VIEW



# ARCHITECTURE – DEPLOYMENT VIEW





# AUTOMATION

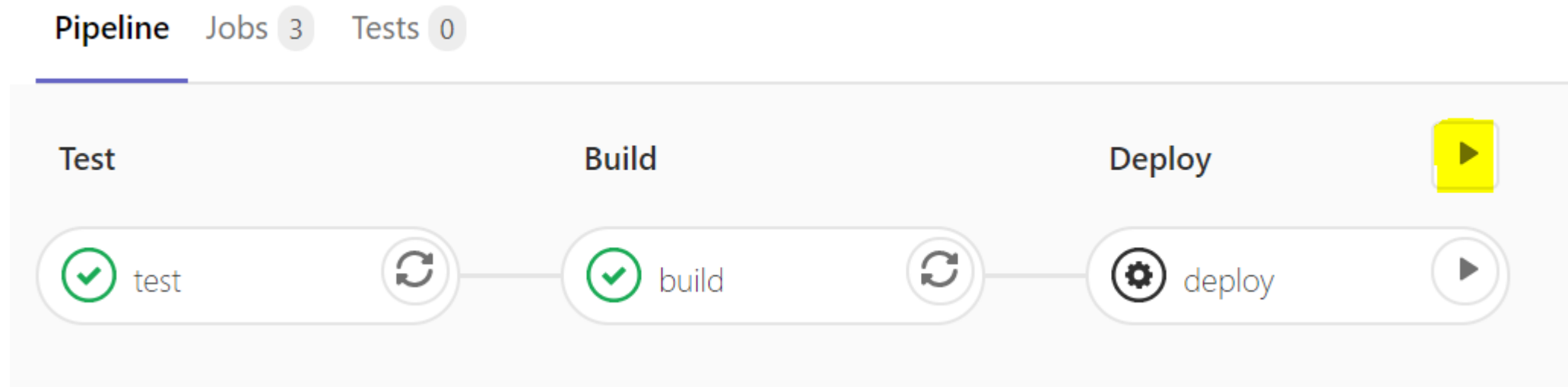
- BDD tests with Cucumber are up and running

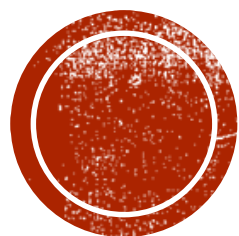
```
public class Stepdefs {  
  
    private String answer;  
  
    @Given("I login with {string}{string}")  
    public void i_am_logged_in(String username, String password) throws Exception {  
        TestRestTemplate testRestTemplate = new TestRestTemplate();  
  
        HttpHeaders headers = new HttpHeaders();  
        headers.setContentType(MediaType.MULTIPART_FORM_DATA);  
        String encodeBytes = Base64.getEncoder().encodeToString((username + ":" + password).getBytes());  
        headers.add(HttpHeaders.AUTHORIZATION, headerValue: "Basic " + encodeBytes);  
  
        MultiValueMap<String, Object> body = new LinkedMultiValueMap<>();  
  
        HttpEntity<MultiValueMap<String, Object>> request = new HttpEntity<>(body, headers);  
        ResponseEntity<String> response = testRestTemplate.exchange(  
            url: "http://localhost:8080/api/user", HttpMethod.GET, request, String.class,  
            new LinkedMultiValueMap<>());  
        answer = response.getBody();  
        System.out.println(answer);  
    }  
}
```



# AUTOMATION

- Pipeline: automatic Test -> Build -> Deploy with one click





# DEMO



BOOKLY - friendship book