

# URL-Archiver

## Intermediate Presentation

January 2, 2024

N. Dora, A. Vejseli, K. Wampfler | School of Engineering and Computer Science

# Table of Content

## Topics Covered

### ► Problem Statement

- Project Goal
- Requirements

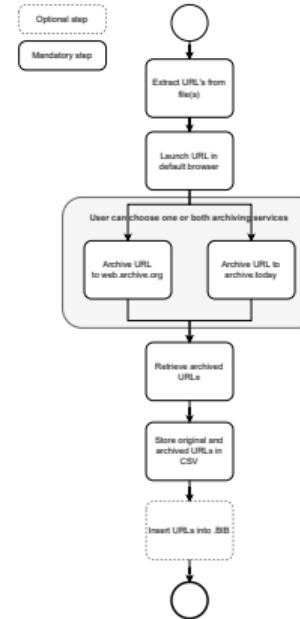
### ► Solving The Problem

- Process Model
- Architecture
- Data Model
- Technologies

### ► Project Management with SCRUM

- Scrum Team
- Our Scrum Adaptions
- Scrum implementation

### ► Questions?



# Problem Statement

# Problem Statement "URL-Archiver"

The ever changing internet

- **ever-changing internet** (websites are taken down or changed)
- not a problem for everyday internet users but...
- ...what about **documents** (e. g. theses, documentations etc.)
  - invalid links to content-relevant information
  - invalid quote links
  - old documents may contain numerous broken hyperlinks
- another case: **Forensic**
  - e.g. **legal report** about a statement on the Internet



# Problem Statement

## Simple Solution

- Archive the **actual state** of the linked websites
  - Wayback machine
  - archive.today
  - Memento Time Travel
  - and more...
- But who has the **time** and **motivation**?
  - ...to search each hyperlink
  - ...to manually archive each website



# Project Goal

- Develop a software that...
  - ...searches hyperlinks within documents
  - ...is capable of **archiving websites**
  - ...can **store** current and archived links in a file
  - ...is **easy** and **intuitive** to use



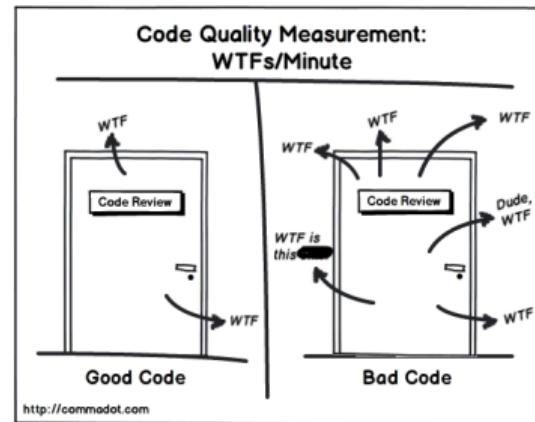
# Functional Requirements

- The software must be...
  - ▣ ...a **Java application** that is compatible with multiple platforms
  - ▣ ...Free and Open Source Software licensed (**FLOSS**)
  - ▣ ...capable of archiving websites on **archive.ph** or/and **WayBackMachine**
  - ▣ ...capable of generating a **CSV-file** with key-value (URL, archived URL)



# Non-Functional requirement

- Coding Norm
- publicly accessible
- Project-Method: SCRUM
- public documents in English
- project report with **LaTeX**

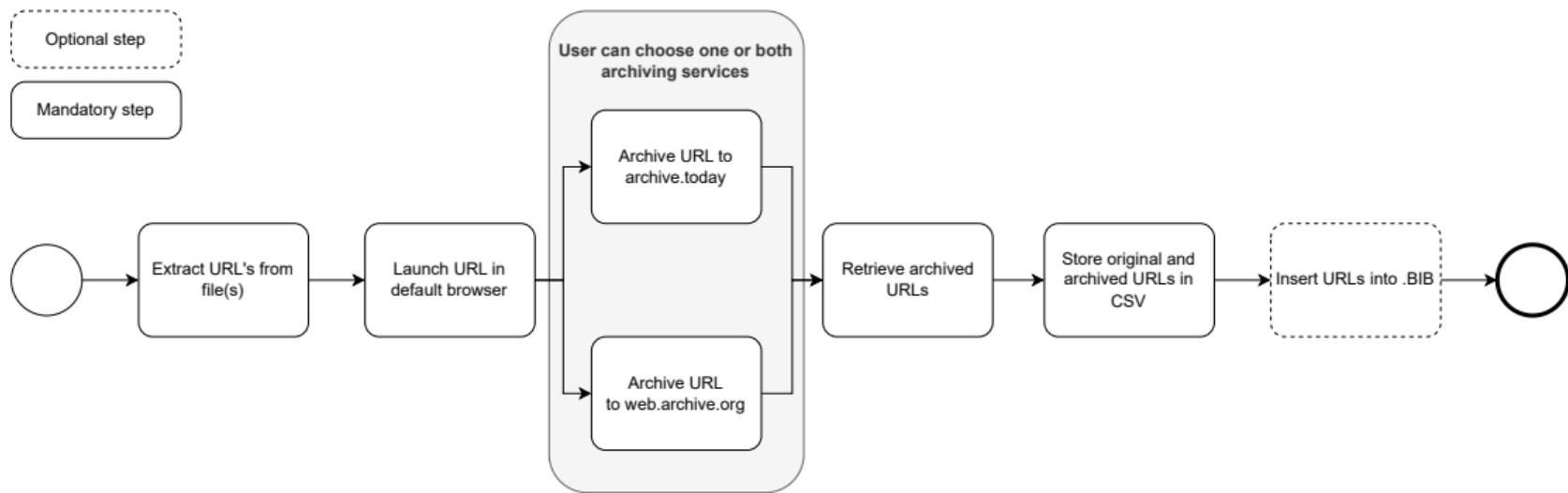


# Solving The Problem

## Process Model

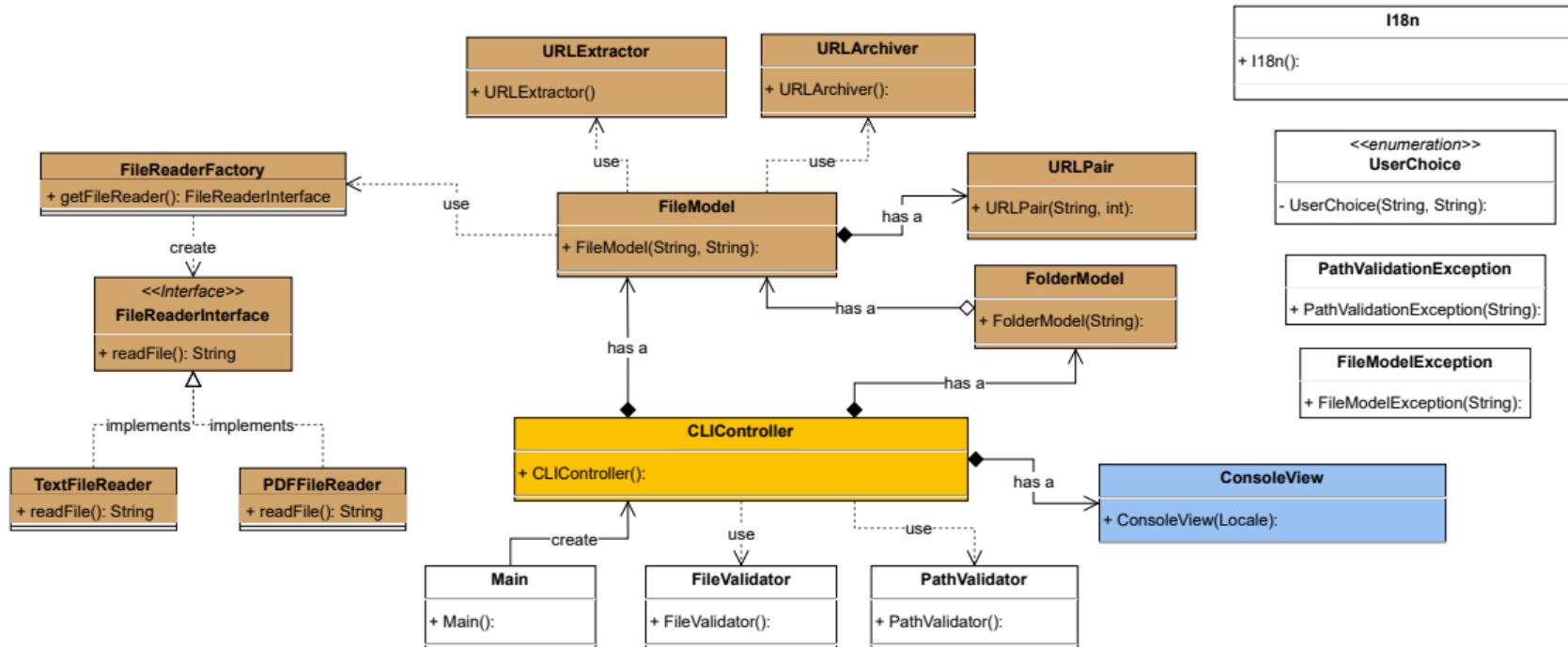
## Workflow for URL Extraction and Archiving

- User can skip URLs, launch them, access help or quit the program at any time



# High Level Class Diagram

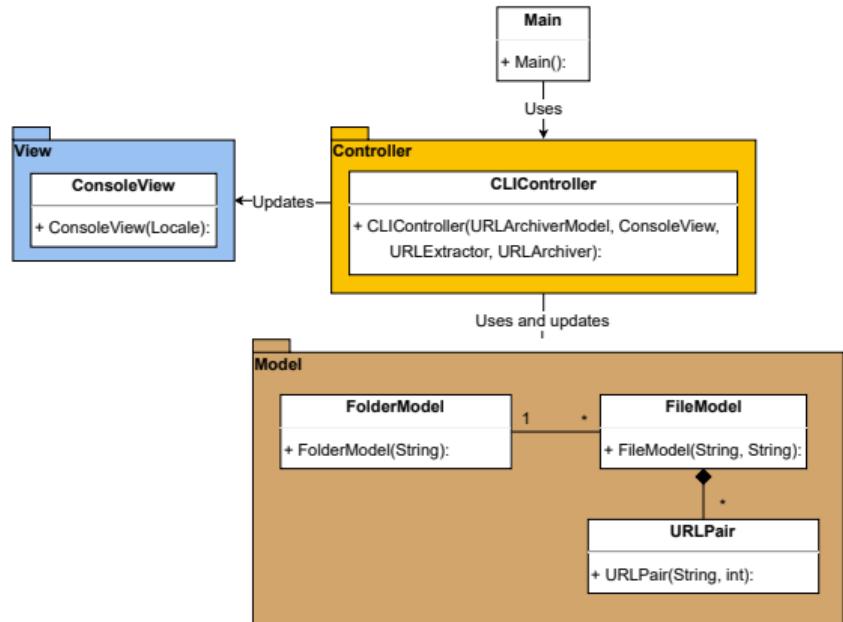
## Overview



# Architecture

## Employing the MVC Pattern

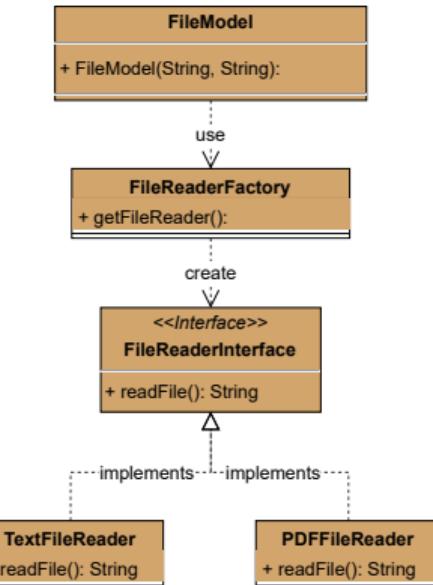
- Modular design for **easy extension**
- Separate data, view, and control flow
- Facilitates the potential **addition of a GUI**



# Architecture

## Factory Pattern

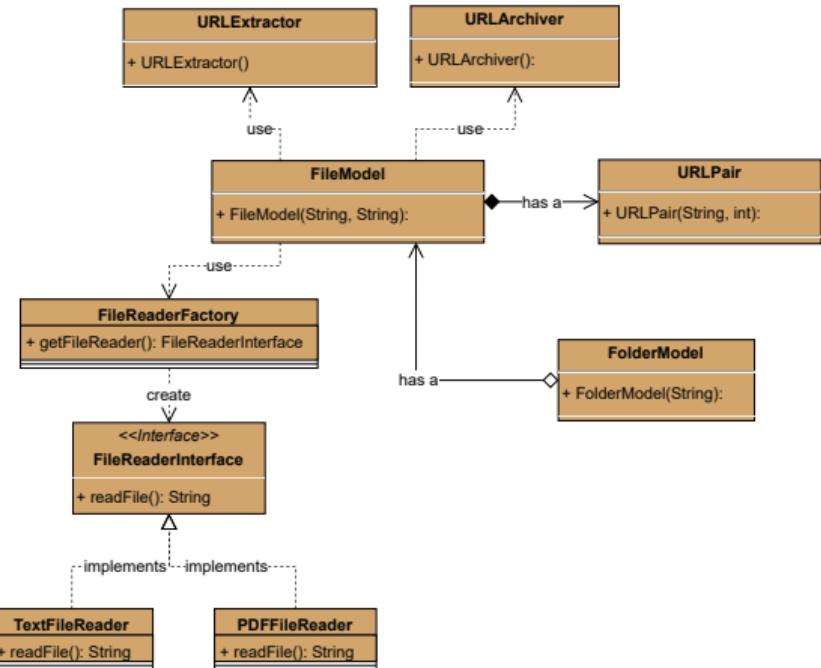
- Scalable and robust design
- Ensures maintainability and clear code structure
- Commitment to best programming practices



# Key Components of the Data Model

## Overview

- **FileModel:** Represents and handles individual files
- **FolderModel:** Manages a collection of FileModel instances
- **URLPair:** Links extracted URLs with their archived counterparts
- **FileReaderFactory:** Organizes and maintains URLPairs
- **URLExtractor:** Used for URL extraction
- **URLArchiver:** Used for URL archiving



# Technologies I/II

- **Java:** The primary programming language for our application
- **LaTeX:** Used for documentation and presentation
- **JUnit 5:** Utilized for unit testing
- **PDFTextStripper (PDFBox library):** Used for extracting text from PDF documents
- **Selenium:** Web automation tool used for tasks like inputting URLs, handling **captchas**, and retrieving archived URLs due to the absence of an official API from archive.today
- **Maven:** Essential for compilation, dependency management, and building the project



# Technologies II/II

- **Archive.today:** One of the services used by the URL-Archiver to archive URLs
- **Wayback Machine:** One of the services used by the URL-Archiver to archive URLs
- **JetBrains IntelliJ IDEA:** Used to develop the program and create the report and presentation in Latex
- **GitHub:** A platform for code hosting and collaboration, allowing us to manage our program's development and version control
- **Atlassian Jira:** A tool for tracking progress and managing our project goals and tasks



# Project Management with SCRUM

# Our Scrum Team



**Nicolin Dora**  
Product Owner & Developer



**Abidin Vejseli**  
Scrummaster & Developer



**Kilian Wampfler**  
Developer

## Other Roles



**Dr. Simon Kramer**  
Stakeholder & Customer



**Frank Helbling**  
PM-Advisor

# Our Scrum Adoptions

## Sprint

<b>Sprint</b>	<b>Sprint planning</b>	<b>Daily Scrum</b>	<b>Sprint Review</b>	<b>Sprint Retro</b>
Two Weeks Sprint Goal (SMART)				

# Our Scrum Adoptions

## Sprint Planning

Sprint	Sprint planning	Daily Scrum	Sprint Review	Sprint Retro
	<p>First Monday</p> <p>Estimate User Stories</p> <p>Define Goal</p> <p>Choose User Stories</p>			

# Our Scrum Adoptions

## Daily Scrum

Sprint	Sprint planning	Daily Scrum	Sprint Review	Sprint Retro
		<p>Daily</p> <p>Twice a week</p> <p>Current state</p> <p>MS Teams</p>		

# Our Scrum Adoptions

## Sprint Review

Sprint	Sprint planning	Daily Scrum	Sprint Review	Sprint Retro
			Last Day Completed? Problems? Testing Product increment	

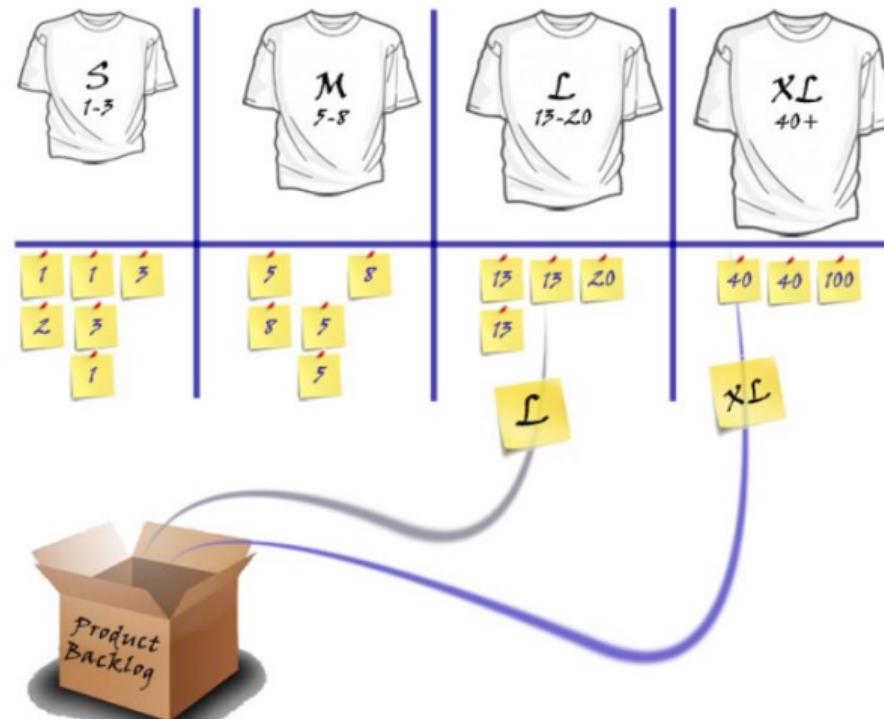
# Our Scrum Adoptions

## Sprint Retro

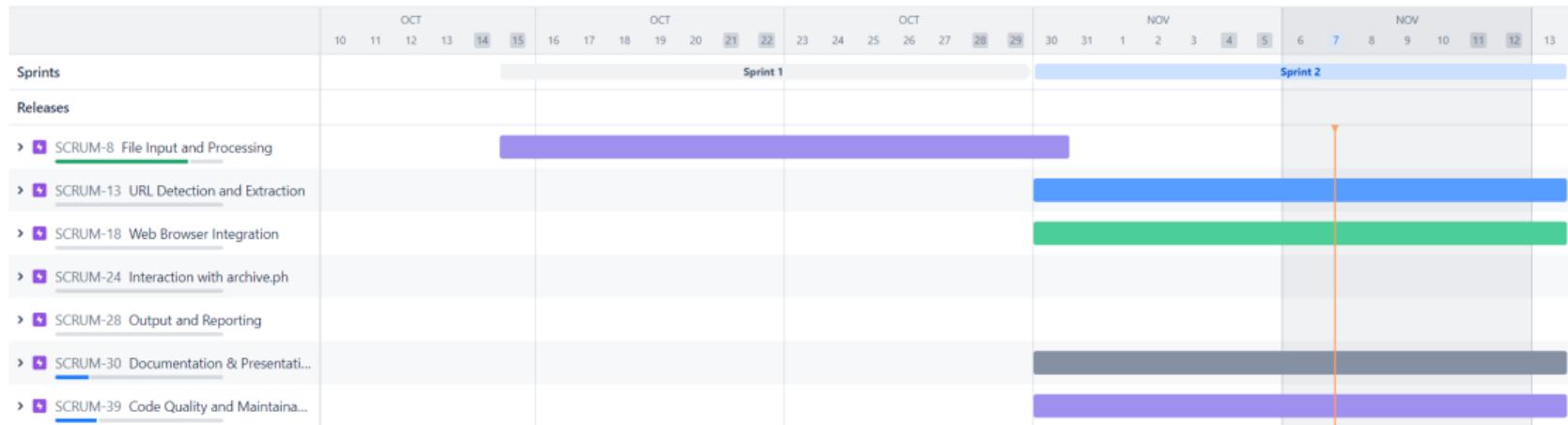
<b>Sprint</b>	<b>Sprint planning</b>	<b>Daily Scrum</b>	<b>Sprint Review</b>	<b>Sprint Retro</b>
				Last Day Feedback Do's & Don'ts

# Estimation Method

## T-Shirt Sizes



# Velocity



# Backlog

SCRUM-16 Store URL Line Number or Context	URL DETECTION AND E...	TO DO ▾	
SCRUM-17 Compile a List of URLs	URL DETECTION AND E...	TO DO ▾	
SCRUM-20 Immediate Archiving Upon Decision	WEB BROWSER INTEGR...	TO DO ▾	
SCRUM-21 Track Archiving Progress	WEB BROWSER INTEGR...	TO DO ▾	
SCRUM-22 Store User Decisions for Reporting	WEB BROWSER INTEGR...	TO DO ▾	
SCRUM-23 Automated URL Submission	INTERACTION WITH AR...	TO DO ▾	
SCRUM-25 User Interaction for Captchas	INTERACTION WITH AR...	TO DO ▾	
SCRUM-26 Automatic Retrieval of Archived URL	INTERACTION WITH AR...	TO DO ▾	
SCRUM-27 Generate CSV File	OUTPUT AND REPORTING...	TO DO ▾	
SCRUM-29 Integrate Archived URLs into Supported Files	OUTPUT AND REPORTING...	TO DO ▾	
SCRUM-31 User Manual	DOCUMENTATION & PUBL...	TO DO ▾	
SCRUM-32 Intermediate presentation	DOCUMENTATION & PUBL...	TO DO ▾	
SCRUM-33 Final presentation	DOCUMENTATION & PUBL...	TO DO ▾	
SCRUM-34 UML diagram	DOCUMENTATION & PUBL...	TO DO ▾	
SCRUM-35 Installation manual & script	DOCUMENTATION & PUBL...	TO DO ▾	
SCRUM-36 Requirements document	DOCUMENTATION & PUBL...	TO DO ▾	
SCRUM-37 Implementation documentation	DOCUMENTATION & PUBL...	TO DO ▾	

# Epics

Epic

Issues without epic

- > File Input and Processing
- > URL Detection and Extraction
- > Web Browser Integration
- > Interaction with archive.ph
- > Output and Reporting
- > Documentation & Presentation
- > Code Quality and Maintainability

## File Input and Processing



### Description

As an user I want to input varius file types via the command line so that they can be prepared for further processing.

### Child issues

Order by ▾ ⋮ +

80% Done

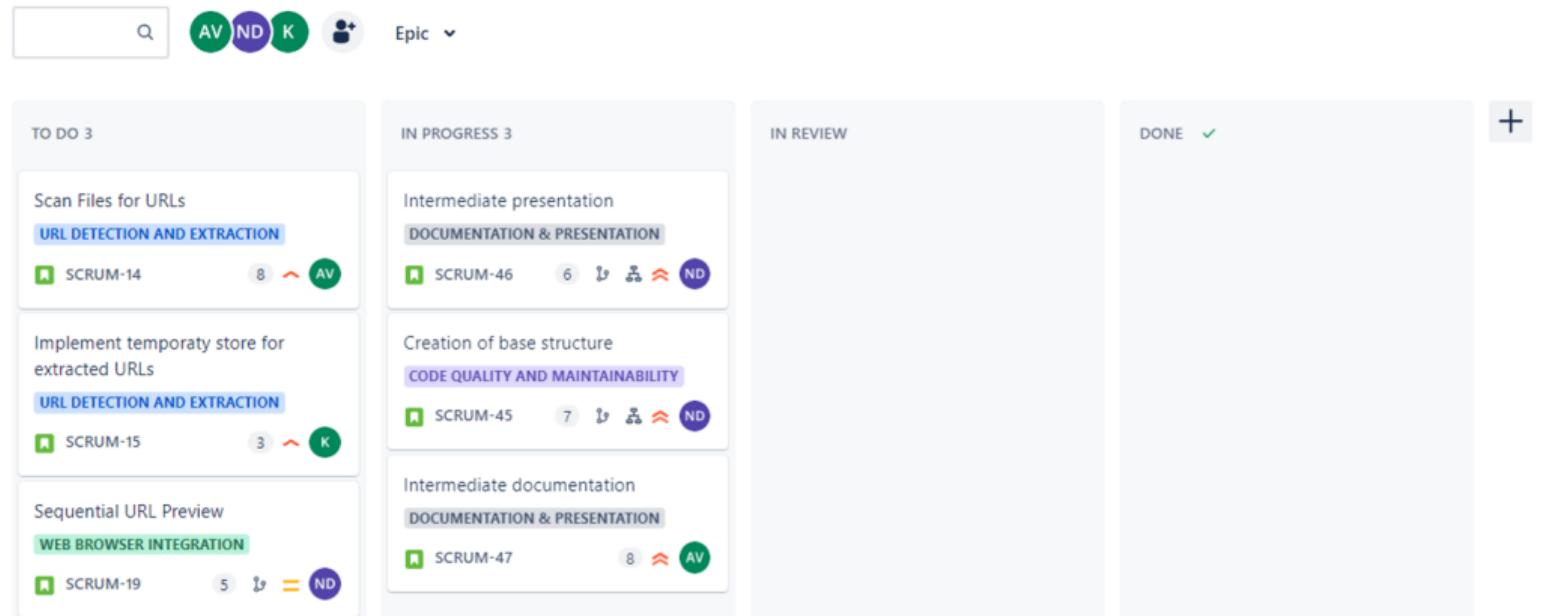
	SCRUM-10	Automatic File Type Detection		13	K	DONE	▼
	SCRUM-12	Processing Feedback		1	K	DONE	▼
	SCRUM-9	Prompt for File Path Input		3	ND	DONE	▼
	SCRUM-11	Processing of Directories		3	ND	DONE	▼
	SCRUM-48	Add user option for processing new path		-		TO DO	▼

# Sprint 2

## Sprint Goal

### Sprint 2

Implement a function to scan a provided text in order to identify and extract any URLs contained within it and upgrade our current console-based interface to enable users to easily open any extracted URL using their default web browser.



# Spint 2

## Categories

### Sprint 2

Implement a function to scan a provided text in order to identify and extract any URLs contained within it and upgrade our current console-based interface to enable users to easily open any extracted URL using their default web browser.

TO DO 3	IN PROGRESS 3	IN REVIEW	DONE ✓
<p>Scan Files for URLs <b>URL DETECTION AND EXTRACTION</b></p> <p>SCRUM-14 8 ~ AV</p>	<p>Intermediate presentation <b>DOCUMENTATION &amp; PRESENTATION</b></p> <p>SCRUM-46 6 ↗ ⚡ ND</p>		
<p>Implement temporary store for extracted URLs <b>URL DETECTION AND EXTRACTION</b></p> <p>SCRUM-15 3 ~ K</p>	<p>Creation of base structure <b>CODE QUALITY AND MAINTAINABILITY</b></p> <p>SCRUM-45 7 ↗ ⚡ ND</p>		
<p>Sequential URL Preview <b>WEB BROWSER INTEGRATION</b></p> <p>SCRUM-19 5 ↗ = ND</p>	<p>Intermediate documentation <b>DOCUMENTATION &amp; PRESENTATION</b></p> <p>SCRUM-47 8 ↗ AV</p>		

# User Story

Description, Acceptance Criteria, DOR and DOD

## Scan Files for URLs

Attach Add a child issue Link issue ...

### Description

#### Description:

As a user, I want the system to scan my input files and identify any embedded URLs so that they can be extracted for archiving.

#### Acceptance Criteria:

- System can detect URLs in a variety of file formats including .BIB, .TEX, .HTML, and .PDF.
- System uses a robust regular expression or other reliable techniques to extract URLs pattern that matches most URL formats.
- Extracted URLs are validated to ensure they are in the correct format.

Checklist for Jira On-the-Fly

To Do Actions

**Details**

Assignee: Abidin Vejseli

Priority: High

Business Value: 8

Story point estimate: 8

Sprint: Sprint 2

**Definition of Ready** 100/100%

Add new checklist item

- :  Ensure a clear definition
- :  Define the functionality or requirement to be implemented
- :  Clearly defined and testable acceptance criteria
- :  Ensure there are no or minimal dependencies
- :  Understood by the whole team
- :  The user story has been estimated
- :  The scope of the user story is small enough that it can be implemented in a single sprint

**Definition of Done** 0/100%

Add new checklist item

- :  Coding standards and best practices are implemented
- :  Unit tests for the feature are written and passed
- :  Any changes to the code or functionality are documented

# Our learnings

Coding	Scrum
Branch per User Story	Create Tasks
Commit Messages	User Story good size
Code Reviews	Weeklys are valuable
Basic code structure	



# Questions?