





Application Developer Project

Version 2.0

Overview

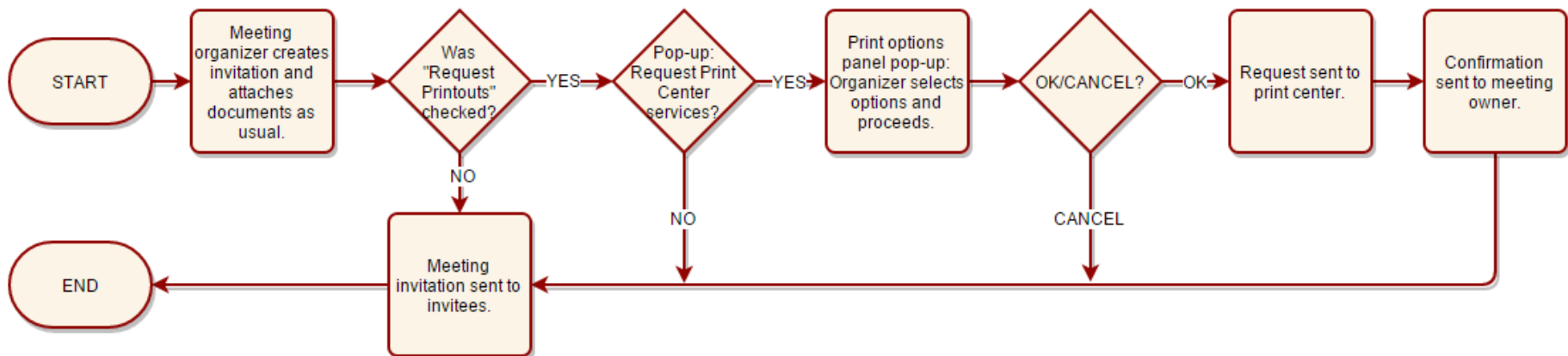
Please create the following project deliverables for an Agile Scrum project using the details provided in this document

-  1. Questions / Assumptions
-  2. Class Diagram
-  3. Implemented Feature
-  4. Code Review Feedback

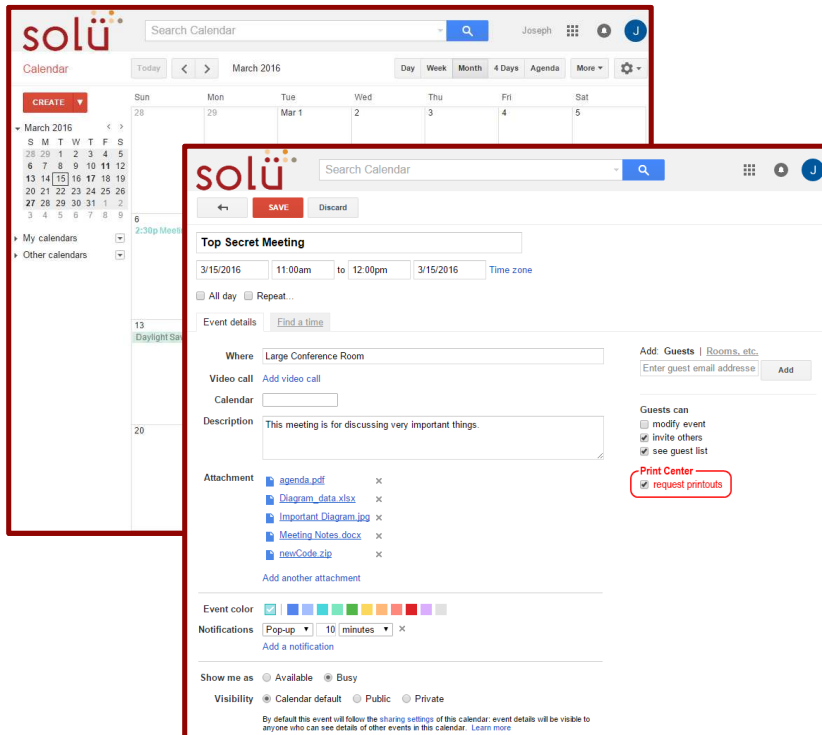
❖ Some project deliverables should be worked on in parallel as you would during a real project

Project Summary – Use Case

You are working on a new plugin for common calendar applications such as Microsoft Outlook or Google Calendar (compatible with desktop and web versions). Marketed towards corporations, the plugin will be called **Print Center** and will allow a meeting organizer to request printouts for documents they attach to an event. After choosing print options, the request will be sent to a print center where the documents will be printed and delivered at a later date.



Project Summary – User Experience Mockup



The mockup shows a web interface for a calendar application. The top navigation bar includes the 'solü' logo, a search bar, and a user profile icon. The main content area features a calendar view for March 2016, with a sidebar for navigation and a list of events. A red box highlights the 'Top Secret Meeting' event details, which include a date/time picker, a description, a list of attachments (agenda.pdf, Diagram_data.xlsx, Important Diagram.jpg, Meeting Notes.docx, newCode.zip), and a 'Print Center' button. The bottom of the interface shows event color selection, notifications, and visibility settings.



Print Center Options

In order for the Print Center to best serve you, please fill in the information below.

Deliver to:

- ☐ Meeting Owner - Delivery will occur one business day prior to the meeting time.
- ☒ Meeting Room - Delivery will occur 15 minutes prior to the meeting time.

If you have any specific instructions pertinent to your entire order, please enter them here...

▼ File	▼ Print	▼ Color	▼ Advanced	▼ Notes
agenda.pdf	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Options	Notes...
Diagram_data.xlsx	<input type="checkbox"/>	<input type="checkbox"/>	Options	Notes...
Important Diagram.jpg	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Options	Notes...
Meeting Notes.docx	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Options	Notes...
newCode.zip*	<input type="checkbox"/>	<input type="checkbox"/>	Options	Notes...

*File is not compatible with Print Center

Submit

Cancel

Project Summary – Sample Project Story

Story ID: E03S05 **Story Title:** Ability to print multiple documents.

Story:

As a meeting organizer, I want to be able to request multiple documents be printed through the print center at the same time so that I can be more efficient when planning my meetings.

Priority:
HIGH

Acceptance Criteria:

Done when:

- I can attach multiple documents to a print request.
- I can select multiple documents attached to a print request.
- The print requests are successfully printed and delivered.

Estimate:
15

Questions / Assumptions

Document Questions / Assumptions that would assist you in creating the required project deliverables

Please consider

- List questions / assumptions in rough order from most critical to least critical.
- What role would normally be responsible or consulted to answer each question?

Class Diagram

Create a UML 2.0 diagram for Story E03S05 outlining the components of the design necessary to satisfy the acceptance criteria outlined by this story. Annotate your diagram with any relevant information not captured in the UML format.

Please assume

- The User Interface has been implemented
- A print API has been provided

Please consider

- What key classes, fields and methods would need to be created to implement the story?
- What interfaces would the classes expose to one another?
- What are the relationships between the classes and how would they interact with each other?
- How would this module interact with the rest of the system?

Implemented Feature – Background

A new request from your team's Product Owner requires the capability to merge two lists of documents into a single print request. Each document has a print priority (1 being the highest).

Please assume

- Duplicates are treated as a request for another copy of the given file
- Documents are grouped by file (i.e. print all copies of docA in direct succession)
- File groups are ordered based on the highest priority document in each

Example

Note: A document is represented in String format as {<document name>, <priority>} (e.g. {docA, 3})

- [{docA, 1}, {docD, 5}, {docE, 2}] *merged with* [{docA, 4}, {docB, 6}, {docD, 3}] *results in* [{docA, 1}, {docA, 4}, {docE, 2}, {docD, 3}, {docD, 5}, {docB, 6}]

Implemented Feature – Method Signature

Using Java, complete the body of the merge method in the provided PrintDocument.java file (method signature is also included below for reference).

Context Code

```
public class PrintDocument {  
    long priority;  
  
    public static List<PrintDocument> merge(List<PrintDocument> d1, List<PrintDocument> d2) {  
        // solution code here...  
    }  
}
```

Implemented Feature – Optional Stretch Goal

In the future, the company would like to expand beyond office applications and sell this product to customers in the print industry. As an optional (non-MVP) item, the Product Owner would like to ensure that the merge operation completes in under two (2) seconds for document lists reaching a combined size of 1,000,000 items.

Confirm the implementation meets this non-functional requirement and provide any documentation or artifacts utilized in the investigation.

Please consider

- What information is necessary to determine the performance of the `merge` method?
- How can it be shown that the performance falls within the required bounds?
- If necessary, how can the `merge` method be rewritten to meet this requirement?

Code Review Feedback

Another developer on the team has completed code (see provided DocumentMetaData.java) for a module that manages meta data for documents in the Print Center. Utilizing the comment form on the following slide, review the code using your preferred language/editor and provide feedback for the developer to address prior to delivering to the test team.

Please consider

- What makes code effective, efficient and maintainable
- If and/or how the code should be modified
- Whether or not the code is ready to deliver to test

Code Review Feedback – Comment Form

Utilize the following table to provide feedback on the Document Meta Data feature, adding additional rows/slides as needed. An example has been included.

Defect ID	Line	Category	Severity	Comment
-	17	Standards	Minor	Modify name of variable 'testvariable' to camelCase ('testVariable')
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				