# *Development Project II (420-K50-HR)*

# *Assignment 1 – Setup - Review, reassess and refactor, versioning, release notes prep*

Date assigned: Monday, January 19, 2025

Date due: **Friday Feb 9th ,** before midnight (Individual)

**Learning Objectives**

Upon successful completion of this assignment, the student will be able to:

* General student setup for a new semester and new assignment in Computer Science.
* Review the process, standards, bugs and refactor proposals defined in Development Project I to carry forward.
* Analyse best practices from another project to determine refactor proposal for current project.
* Update and prioritize project bug/issue tracker items as a team.
* Learn and adopt Dependency Injection Containers and MOQ. Introduce these frameworks into your project Test approach.
* Research best practices for software product versioning
* Research best practices for software product release notes
* Propose versioning and release notes to be applied to their project
* Understand the requirements for going to production
* Determine the current status of their project
* Perform a gap analysis on where their project needs to be and propose an action plan

*(set of tasks or work items)*

To do:

This assignment is to be completed as **both** a team and individual activity.

For the **individual** parts of the assignment:

* create a new Word document named **YourUserName\_Teamxx\_K50\_A01.docx.** i.e. celliott\_K50\_Team16\_K50\_A01.docx
* Make sure that your file name includes your shortened username and team number.

**All student responses should be highlighted in BLUE TEXT to be easily read by the instructor.**

All submissions that are group-based (i.e. have “TEAMXX” in the title) should be submitted to Moodle in the “Group submissions” folder. Individual submissions are to be submitted to the “Individual Submissions” folder on Moodle.

# Section 1 – Individual Setup

## Section 1 Part A – Organization and Plan – Mostly Individual

Objective: Understand the goals of the week. Come up with a plan and timetable.

***Individual Work***

Analyze this assignment and plan.

1. Identify the individual deliverables:

The individual deliverables include:

Section 1

Organizing ourselves and planning out the assignment so we can plan accordingly

Section 3

Prioritize the backlog

Section 4

Versioning and release notes

Parts of Section 5

Planning for the future – deliverables, challenges, effort

1. Identify the team deliverables:

The team deliverables include:

Section 2

Research, presentation, demo, refactoring

Parts of Section 5

Teamwork convergence

1. What is your estimate on when the individual portion will be complete? Negotiate with your team to agree upon a time you’ll meet and organize to start working on the team portion.

The idea will be that we will be meeting up at certain points to work as a team as we make progress individually. Ill be trying to finish all the individual stuff as soon as possible to be prepared for a discussion whenever.

1. State your estimate and the team rendezvous time as well as any concerns or risks. These must be written down and stated professionally and in a detailed fashion.

Our team has prepared a rendezvous Jan 28th for section 2 and Feb 4th for section 5.

## Section 1 Part B – Updating Work items from Dev I - Individual

Based on your Final comprehensive assignment of 420-K40-HR

1. Add your System Test bugs as “issues” in Azure DevOps.

Your bug report must have the details required to reproduce the bugs. Be sure to classify this as a “bug” work item and not a product backlog item.

1. Add Refactoring Proposals as a backlog item with tag “refactor”.
2. Do not worry if there are duplicates with someone else’s bug or refactor task. You’ll collectively clean that up later.

*Please note for future reference that the* ***cloud-based*** *Microsoft Azure DevOps implementation, Microsoft calls everything generically an “issue” even if it is what we in Computer Science would call a bug. This is due to the “Kanban” nature of tasks in Azure DevOps.*

**Update the tables below with your updates:**

**Bug List**

|  |  |
| --- | --- |
| **Azure DevOps Issue URL** | **Bug Title** |
|  | Your response |
|  |  |
|  |  |

**Refactor Task List**

|  |  |
| --- | --- |
| **Azure DevOps Issue URL** | **Backlog Item** |
|  | Your response |
|  |  |
|  |  |

## Section 1 Part C – Review Process and Standards - Individual

1. Review the team’s coding standards document and reconcile against samples of your code. We will be holding code inspections later on. Ideally, your team’s code meets this standard.
2. Find at least 3 defects and submit them to Azure DevOps.

|  |  |
| --- | --- |
| **Defect parameter** | **Possible values** |
| Origin | Requirements, Design, Implementation, Testing |
| Type | Missing, Wrong, Extra, Usability, Performance, Style, Clarity |
| Severity | Major, Minor |

**Code inspection defect report**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Filename/DevOps version/line#** | **DevOps Issue #** | **Origin** | **Type** | **Severity** | **Description/Comments** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

1. Review the team’s coding standards document. If the document is wrong or needs to be updated, now is your chance to make some proposed changes.

Fill in the table below with your proposal. No more than 5 items.

**Proposed Changes to Coding Standards**

|  |  |  |
| --- | --- | --- |
| **Proposal #** | **Category** | **Proposal/Description** |
| **1** |  |  |
| **2** |  |  |
| **3** |  |  |

1. Reflect on last semester’s sprints to generate process change proposals

Identify if new lessons learnt should be added. Define TWO process changes. Modify the below table as necessary.

What behaviours, approaches or processes would you change to improve the sprint process this semester to improve in the following categories:

* 1. Ability to deliver and complete User stories
  2. Ability to deliver quality (cover code standards, white and black box testing)
  3. Team and individual evaluation and scoring
  4. Meeting overhead (scrums, sprint reviews/planning, retrospectives).

**Proposed Changes to Process**

|  |  |
| --- | --- |
| **Category** | **Proposal/Description** |
| **Ability to complete User Stories** |  |
| **Ability to Deliver quality** |  |
| **Team and Individual Scoring** |  |
| **Meeting Overhead** |  |
| **[<make your own category as you see fit>]** |  |

## Section 1 Part D – Budgeting - Individual

1. Review the timeframes for each of the following and provide the available person-hours to be applied to each.

Hint: count the number of your teammates. Look at the course outline for the expected hours per week in this course, then do the math. The course outline also contains the dates for each release. Assume “end of week” deadlines apply to assignments and sprints *( i.e. midnight on Fridays )*.

This is to understand how many hours were available in past timeframes, so that you can map it onto what is achievable moving forward.

|  |  |  |
| --- | --- | --- |
| **Timeframe** | **Start date** | **Available person hours** |
| Release 0.1 – Sprint 1  *(Fall 2024)* |  | Note: ponderation and team size |
| Release 0.2 – Refactor Sprint 1  *(Winter 2025)*  *(2nd dev sprint overall)* |  | Your response |
| Release 0.3 – K50 Dev Sprint 1  *(Winter 2025)*  *(K50 Dev Sprint 1, 4th dev sprint overall)* |  |  |
| Release 0.4 – K50 Dev Sprint 2  *(Winter 2025)*  *(K50 Dev Sprint 1, 4th dev sprint overall)* |  |  |
| Release 0.5 – K50 Testing Sprint 1  (Winter 2025) |  |  |
| Release 0.6 – K50 Testing Sprint 1  *(Winter 2025)* |  |  |
| Beta Release 1 – K50 Testing Sprint 2  *(Winter 2025)* |  |  |

# Section 2 – Setup Team Work

## Section 2 Part A – Team research, presentation and demo

As a team, you will research the following items to be incorporated into your projects. As a team, figure out how to enter this into Azure DevOps, how to sub-task it, and how tasks are assigned. All team mates must contribute.

The purpose of the slide pack is to capture the best practices for the team moving forward.

Include an overview on what the term “Best Practices” means and how this will be applied to your project.

For each research item:

Research the topic

Put together a slide presentation that includes

For each topic

What problem does this solve, why bother?

The basic approach with sample.

Proposal on application to your project and the standard moving forward

A single demonstration/application of the topic on your project .

Check this demonstration or application into Azure DevOps.

Are there any other best practices that you’d like to apply to your project? Discuss and agree as a team.

Collate the presentation in a single PPT file for the whole team.

Book a team presentation time slot, invitees are for all team members as well as Prof Chris. Not to exceed 1 hour. Determine how you’ll ensure these practices are followed for the remaining of the project (consider process update).

After the team presentation.

Provide meeting minutes.

Create DevOps taskboard tasks for updating the project for each topic (recall only sample coverage was previously done).

Update the code standards (or provide a checklist for use during code review) to capture the newly adopted practices.

**Best Practices as Suggested by Faculty**

|  |  |  |
| --- | --- | --- |
| Dependency Injection  “New is glue”. Essay on decoupling with DI | Make sure your code is structured for modularity and testability | [Link 1](https://ardalis.com/new-is-glue)  [Link 2](https://docs.microsoft.com/en-us/aspnet/core/fundamentals/dependency-injection?view=aspnetcore-2.2)  Sample project provided on DevOps collection:Samples |
| Single Responsibility Principle (SRP) | Design pattern | [Link 1](https://deviq.com/single-responsibility-principle/) |
| CICD and Pipelines | Deployment technology | [Link 1](https://www.atlassian.com/devops/devops-tools/devops-pipeline)  [Link 2](https://docs.microsoft.com/en-us/azure/devops/pipelines/?view=azure-devops)  [Link 3](https://docs.microsoft.com/en-us/azure/devops/pipelines/create-first-pipeline?view=azure-devops&tabs=java%2Ctfs-2018-2%2Cbrowser) |
| Don’t Repeat Yourself (DRY) | Design pattern | [Link 1](https://deviq.com/don-t-repeat-yourself/) |
| Unit testing - Testing (MOQ and DI) | Design pattern, new infrastructure and tools | [Link 1](https://docs.microsoft.com/en-gb/aspnet/core/mvc/controllers/testing?view=aspnetcore-2.2)  Sample project provided on DevOps collection:Samples  This is an approach to testing controller, but the concept can be applied to testing any layer.  (Use DI and MOQ, together, to isolate and unit test) |
| Integration Testing | Design pattern, new infrastructure/tools | [Link 1](https://docs.microsoft.com/en-gb/aspnet/core/test/integration-tests?view=aspnetcore-2.2) |
| Model Validation | Robustness | [Link 1](https://docs.microsoft.com/en-us/aspnet/core/mvc/models/validation?view=aspnetcore-2.2) |
| CSRF proof | Security | [Link 1](https://www.youtube.com/watch?v=vRBihr41JTo&t=406s)  Investigate the [ValidateAntiForgeryToken] decorator in .Net core |
| Overposting | Security | [Link 1](https://docs.microsoft.com/en-gb/aspnet/core/data/ef-mvc/crud?view=aspnetcore-2.2#recommended-httppost-edit-code-read-and-update) |
| DB Failures | Robustness.  Don’t always assume DB requests return with success. | [Link 1](https://docs.microsoft.com/en-gb/aspnet/core/data/ef-mvc/crud?view=aspnetcore-2.2#the-read-first-approach-to-httppost-delete) |
| Sanitizing input | Security | [Link 1](https://www.w3schools.com/sql/sql_injection.asp)  Sanitizing JS and SQL (vs XSS, SQL injection). |
| (Bonus best practice) | TBD | Suggest another best practice not already listed for bonus points |

## Section 2 Part B – Refactor exit criteria

On exiting refactoring ensure the following goals are met:

|  |  |
| --- | --- |
| **Goal Description** | **Deliverable** |
| Full stack | The stack must be production stack/architecture  (no layer/block completely spoofed). |
| Authentication and Authorization | Your application must be IAAA system integrated  *(will be discussed further in class)* |
| Best Practices are implemented | Best practices are captured in team documents and implemented |

Ensure Azure DevOps’ taskboard is completed, and that tasks are added to accomplish these goals.

## Section 2 Part C – Process

Only one person on each team needs to submit the team portion of this assignment. Nominate a team member to do this. Create a task on the Devops Taskboard indicating who was chosen, and assign it to that person.

From your own review of the above Process review during individual work: have a team discussion about this. Capture these discussed items in the form of a single memo capturing the general discussion, specific points, and any new, open actions that need to be taken.

Capture what actions were done and need to be done on the DevOps Taskboard, assigned by whom and document all key decisions.

Make sure the team collectively reviews the minutes before they are submitted.

There will be a follow-up meeting later to discuss the team proposals with your project manager.

**Sync up with the Assistant Project Manager before any team deliverable is submitted.**

**All individual submissions should be complete before working on the team portion. If this is not possible or practical for your team, please discuss with the project manager.**

**See Moodle for details on how to submit the team portions. Discuss with the project manager if there are any clarifications needed on the deliverables or how to submit.**

1. Update the Action Item Register spreadsheet to cover all open action items from last semester. Get together as a team and review all past actions to get agreement on what was closed and what remains open.
2. **Update the Lessons Learnt tab within the same spreadsheet mentioned above to cover all the lessons learnt from last semester that the team agrees on. Provide a snapshot in the team memo. Ensure that the team has reviewed and agreed with the lessons learnt items carried forward.**
3. Ensure that the tasks for the Team deliverables are all captured and organized in Azure DevOps.
4. Discuss any individual coding standard change proposals. Determine if there are any agreed upon changes and if so capture these in the updated standards document and in the minutes. (Ref: Section 1.D). Add to table below.
5. Provide a Budget table (Ref 1.E) that you all agree on.
6. Discuss the rubric with your team and present any proposed changes to me by Tuesday noon.
7. Prioritize, filter out duplicates, and order the bugs and Refactoring items. For now, order this above the customer-visible backlog items. We have about a week to get this cleaned up before we resume sprints (depending on your estimates).
8. Roughly estimate the size of the bugs and Refactoring tasks. (enough to understand and chunk into categories: small/medium/big/enormous).
9. Re-prioritize and re-order if you need to the above list, if you need to
10. **Send an email to the team and the project manager stating your “Refactor Plan” and commitment.** This is just like a sprint goal/commitment email. Ensure that you:
    1. Provide a screen capture of your team’s agreed upon backlog list.
    2. Be clear which items the team is committing to. Use the iteration label “**Refactor**” which is scheduled to end one week out.
    3. If it is not obvious from your task names alone, provide a description of the beneficial project outcomes that result from this effort.
    4. List the resource availability to do so. i.e. total number of available person hours applied.
    5. Do not send this to the customer / end user. This is an internal matter.
    6. The title of your email MUST reflect the course standard: i.e. the subject line must include K50 and your team number. All team members must be included on the distribution list for this email. (i.e. they are in the TO: or CC: fields)
11. Discuss your proposed changes to process and capture the agreed upon proposals:

What behaviours, approaches or processes would you change to improve the sprint process this semester to improve in the following categories:

* 1. Ability to deliver and complete functionality
  2. Ability to deliver quality (cover code standards, white and black box testing)
  3. Team and individual evaluation and scoring
  4. Meeting overhead (scrums, sprint reviews/planning, retrospectives).

As a team, collate, discuss and reach agreement on proposed changes and update the table below:

**Proposed Changes to Process (Team)**

|  |  |
| --- | --- |
| **Category** | **Proposal/Description** |
|  |  |
|  |  |

# Section 3 - Execution

## Section 3 Part A – Work on backlog prioritized issue list

*Individual Work:*

Take a task (bug or refactoring) from your prioritized list for the Refactor iteration and get working!

This is of course all utilizing the CSAZURE DevOps taskboard, as for every other Dev Proj sprint.

Do NOT work on any task that has not been committed to without agreement with your team and your project manager.

Follow the process. All changes must be reviewed.

There will be a team assessment done for this next week. You will scrum daily on this and treat it like a regular iteration. We’ll have a quick review on the iteration results later in the week.

**To submit**

When you have completed the assignment, the following deliverables are required:

**Individual:**

upload the **YourUserName\_K50\_A01\_SetupReassessRefactor.docx** document to Moodle

**Team:**

Meeting Minutes from Best Practices Presentation (email)

Refactor commit (email)

Memo from Group discussions (Moodle, ref 2.C)

Team self assessment of commitments. Write a few statements indicating how you performed here, as a team.

Response to commit email at the end of the Refactor evaluation.

(email, end of refactor sprint).

*(“K50 Team #” must be in the email subject line or marks will be forfeited.)*

# Section 4: Versioning, Individual work

For the individual part of the assignment, create a new Word document named **YourUserName\_Teamxx\_K50\_A02a.docs.** Include your (short) username and team number.

Effort estimate for this: < 60 minutes (for a focused individual)

## Part A – Software Versioning

1. Propose a numbering plan on how your product should be versioned. Explain the rules/conditions for when each number is updated and how.
2. Propose a mockup of where the software version will be displayed on your product. It should be visible on every page.

## Part B – Release Notes

1. Identify the purpose of Release Notes and the intended audience.
2. Identify 3 articles on Release Notes Best Practices. Provide the links and a summary for each. List 5-8 best practices that you’d like to adopt described that you’d recommend.
3. Design your Project Team’s Release Notes. Provide a template for Release Notes that you would propose to your team. Your proposal should have enough detail so that a teammate would understand how to fill it out.

Capture the above in your document. The goal is to convince your teammates that you’ve got a well thought-out and achievable approach. Include answers to the items 1, 2 above, and a sample of your proposal would look like for the first sandbox release you provided to the user.

# Section 5: Versioning: Teamwork Convergence

Meet as a team and converge on a common scheme for software versioning and release notes. Capture it in **Teamxx\_K50\_A02a.docx**

## Part A – Software Versioning

Put together a document explaining the software versioning scheme, conditions for when the components will be updated and where it will be displayed.

Capture it in **Teamxx\_K50\_A02a.docx**

## Part B – Release Notes

Put together a section in your document explaining the purpose, structure and format of the team’s release notes. Provide a sample for your first sandbox (doesn’t have to be technically correct, but does show the format, coverage and purpose).

Capture it in **Teamxx\_K50\_A02a.docx**

Effort estimate for this: < 45 minutes (for a focused team)

Note: after feedback, this content will be in your System Document so that the maintenance and development teams will have guidelines on how to do this.

**To submit**

When you have completed the assignment, upload the document to Moodle in the provided submission folders.

# Section 5 – Individual Work

During this part of the assignment, you and the team will need to:

1. Plan ahead to determine what needs to be done to the end of the project
2. Negotiate a migration plan for the project with the operational owner. How would they start using your product given existing workflows?

Read and analyze the K50 Production Criteria for Development Projects.

Also review the Course outline to understand the future deliverables.

Note all the work in this section is captured in a single spreadsheet document.

For the individual part of the assignment, create a new Word document named **YourUserName\_Teamxx\_k50\_A02\_ProductionPlanning.xlsx.** Include your (short) username and team number.

Effort estimate for this: < 90 minutes (for a focused individual)

## Section 5 Part A – Identify Requirements and Deliverables and current status

Put together a spreadsheet that captures:

1. Requirements and Deliverables. Organize and number these based on the numbering in the “K50 Production Criteria for Development Projects” document for traceability. (i.e. 3.2 refers to Operational documentation).
2. Current status of each deliverable: (New, In Progress, Done)
3. Comment/concerns/risks – add a text column to capture any comments, concerns or risks

## Section 5 Part B – Identify outstanding work items, loose planning

For the items that are not **Done** from Section 5 Part A, add columns in your spreadsheet to capture

1. Deliver by – Options are {Week 6,7,8,9}. **Propose a “deliver by” deadline.**
2. Effort estimate – in person hours, how much would this be to get to done?

(Exceptions: Do NOT estimate any of the system test and bug fix items)

## Section 5 Part C – Roll up the results and report

Either manually or using your spreadsheet prowess, report on

1. Number of Requirements and Deliverables
2. % done (vs required)
3. Number of outstanding hours to complete

As computer science near-graduates,it’s expected that this chart be comprehensive, detailed, have a proper narrative and correct in its content. An example will be provided.

# Section 5 Part D – Team work - Converge

## Part A – Planning

Collate your work and present one spreadsheet that represents the whole team’s view on the Requirements and deliverables.

Get general consensus on parts A, B, C from Section 1 that you’ll carry forward into the Team’s spreadsheet.

If any team members’ item has been dropped on the Team spreadsheet, please have indicate why in a separate tab “Dropped Requirements and Deliverables”.

Task this out in the DevOps task board. Make sure every outstanding task has a traceable taskId column in your spreadsheet indicating the DevOps task.

For the team part of the assignment, create a new Word document named **K50\_A02\_TeamXX\_ProductionPlanning.xlsx.** Include your team number.

Effort estimate for this: < 45 minutes (team)

**To submit**

When you have completed the assignment, zip and upload the document to Moodle in the provided submission folders.