Project B

Console application

Description of the course

Course name:	Project B – Developing a console application with Scrum
Course code:	**INFPRJ03B**
ECTS:	13 ECTS
Study points and workload:	This course provides you with thirteen (13) study points, which corresponds to a workload of 364 hours. The recommended distribution of these 364 hours during the study weeks is as follows: **Onsite project lessons, 16 weeks * 6 hours a week: 96 hours** **Project week: 24 hours** **Work from home days, 16 weeks * 15 hours a week: 244 hours**
Prior knowledge:	For this course there is prior knowledge required from BaseCamp.
Testing:	A shippable product that meets the requirements of the stakeholders needs to be delivered. The management of the product and process will also be assessed.
Learning outcome:	After finishing Project B, you are capable of realizing a working software product through the use of Scrum, version control and a software framework, taking into consideration the wishes of the stakeholders.
Learning goals:	1. The student has shown a professional approach concerning critical & analytical attitude 2. The student knows the principles of the project methodology Scrum and can apply them within the group while working on the project. 3. The student is able to set up a requirement specification for the console application based on the requirements of the Product Owner (PO). 4. The student can develop a console-based application with attention to architecture and implementing technical concepts in the application. 5. The student can improve the application through testing to validate the quality of the final product.
Contributes to:	Learning goal 1 & 2 contributes to the competence Professional skills, learning goal 3 to the competences Analyze & Design and learning goal 4 & 5 contributes to the competence Realization
Content:	You learn to work in a team context (*process*) and to realize a project assignment (*product*) for a client.
Notes:	Attendance is obligatory for the project meetings because you are working in a team and are therefore dependent on each other. When you have been absent more than four times, you will get an ND unless otherwise advised by a dean (decaan) The teams will be created according to the instructions of the teachers.
Course coordinator:	K. Krul & G. Bax
Date:	August 2024

1. General information

1.1. Introduction

In this course you will work on creating a console application in a team of students. Using a chosen case study, you will begin to work through the software development lifecycle. This includes working on requirements, programming and testing.
You might think that programming applications and building software is the main focus. Yet, in practice, it is not the only thing that counts. You will need to work on developing your professional attitude during the

course. Imagine you are a software developer who has excellent programming skills, but cannot communicate properly or work together with colleagues. Working together, consulting, presenting etcetera can lead to interesting innovations and solutions in complex situations.

1.2. Learning materials

Provided knowledge:
For this course the peercoaches are available for support. For each class a peercoach is assigned and scheduled. If this is not the case, please contact your teachers: Tutor and/or Product Owner (P.O.). They are available for project related questions.

Workshops Semester 2
Workshops Scrum
Workshop Interviewing
Workshops GIT
Workshop Use Case Diagram (UML)
Workshop Three-layer model
Workshop Presenting skills
Workshops Testing

2. Program and contents

2.1. Case

The Product Owner’s client is in need of a reservation console application. Due to the fact that the economy is in a lift, people tend to enjoy life more these days. They travel, go out to eat in restaurants or visit the cinema. Because of the increasing amount of visitors and the pressure that comes with it, a solution is needed for the business owners. A solution in the form of an application that can be used by the clients to see the availability of the services the business provides. The Product Owner has three concepts of console applications that he/she wants to be developed and sell to his/her clients. To help you meet the needs of the Product Owner we have written down some basic functionalities to help you start, so you can begin and further work out your case.

1. Case 1: Cinema
   - the ability to see the available movies
   - per movie you can see the genre, dates and times & prices
   - the ability to reserve 1 or more seats for a specific movie
   - etcetera
2. Case 2: Restaurant
   - the ability to see the menu
   - the ability to reserve 1 or more tables for a certain amount of people for a specific day and time
   - etcetera
3. Case 3: Airline
   - the ability to see information about the different flights and information about the airport
   - the ability to book a specific flight for a specific day and time
   - the ability to book a flight for a specific amount of people and a specific chair
   - etcetera

It is your task to create a console application which satisfies the wishes of the Product Owner and meets the requirements as specified in section 2.2 **Technical requirements** of this document. During the meetings with the Product Owner, you have to get the information you need to fulfill the demands of the Product Owner regarding the console application you have to make. Keep in mind that you have to use Scrum as a project methodology.

It is your job to keep track of your individual development as well as that of your team. You achieve this by sufficiently using Scrum, presenting adequately and using the right meeting skills. Besides keeping track of the progress (individually and of the team), you will need to present visible proof of your commitment (process) and contribution to the product. This is part of the Final Deliverables (see section 2.4 **Deliverables**).

2.2. Technical requirements

The application must be made in C#, in VS or VS Code IDE. Data should be stored in JSON files.
The application must be made with the three-layer architecture.

2.3 Time table

In the following table you can see the planning of each week. There is time scheduled to work on the projects, you are expected to work independently with your team the rest of the week.

Agenda items are per week. The teacher is free to plan the items on which day of the week is the most convenient for them.

OP1		
Week	Sprint	Items
1	Project Alpha	Get to know the class Scrumfest Workshop Github 1: Introduction
2	Project Alpha	
3	Project Alpha	
4	Project Alpha	Finish and present your game.
5	0	Workshop Interview -> teams interview the PO Teams work on product backlog
6	0	Workshop Three-layer model Workshop UML -> teams create UML diagram Finish product backlog + sprint backlog 1
7	1	Start coding! Workshop Presenting Workshop Github 2: Branches & merging
8	1	Workshop Github 3: Merge conflicts
9	2	OODP Exam Workshop Testing 1: unit testing *Project lesson might be cancelled, but keep working on project*
10	2	Sprint reviews
OP2		
11	3	Workshop Testing 2: System test Teams present midterm product
12	3	Sprint reviews
13	4	Workshop Testing 3: User acceptance test
14	4	Sprint reviews
15	5	
16	5	Sprint reviews
17	6	Project week Work on finalizing all documents Teams present final product Teams hand-in final delivery (Friday 23:59)
18		
19		
20		Resit hand-in

2.5 Sprint deliverables

Each sprint must contain the deliverables as follows:

- Sprint backlog
- Burndown chart
- Summary of the retrospective (Description of used methodology and summary of outcome)
- Sprint review presentation
- Reflection on scrummaster role
- Testscript for unit testing, starting from sprint 2 (including testresults)
- Testscript for system test, starting from sprint 3 (including testresults)
- Testscript for user acceptance test, starting from sprint 4 (including testresults)

2.6 Final deliverables

The deliverables need to be uploaded on Brightspace. The deliverables are:

Individual:

1. Written evidence of the individual contribution to the product (code snippets, Git contribution, summary of contribution / completed tasks, logbook)
2. Documentation in accordance with the scrum master role:
   - Summary and the outcome of the retrospective
   - Screen shot of the sprint backlog after sprint planning
   - Short summary of the refinement session
   - Short summary of the review (minutes + action list)
   - Burndown chart of the sprint you were scrum master
   - Handouts of the presentation of the sprint review
   - Reflection on your role as scrum master and your role as presenter during the sprint review.

Team:

1. Proof of applying Scrum (components such as: product backlog, sprint backlogs, scrumboard, burndown charts, reviews, retrospectives, acceptation criteria, definition of done)
2. UML: Use case diagram
3. Proof of implementing the requirements of the PO in the application each sprint (evidence of functional improvements, e.g. screenshots, videos)
4. Proof of implementing testing:
   - Unit testing implemented in the code
   - Risk level per user story in product backlog and sprint backlog
   - Completed testscripts for systeemtest (each sprint) starting from sprint 3 (excel template)
   - User acceptance test completed (excel template)
5. Proof of implementing the coding concepts in the application. For each concept you should specify why you used it and where (include code snippets: screenshots as evidence). Specify the decisions you made. For instance why did you choose an if / else statement (selection) instead of a for loop. **If a programming concept listed below was not applicable in your application, you should give a written explanation of why it was not applicable.**
   - Primitive types, Operators
   - Selection (if, switch)
   - Loops (for, foreach, while)
   - List
   - Class
   - Constructor

  - Fields
  - Methods
  - Classes: objects as fields / in methods
  - Static classes; methods; fields; constructors;
  - JSON (reading / writing)
  - JSON (exception handling)
  - LINQ
  - Polymorphism: overloading
  - Inheritance
  - Access modifiers
  - Properties
  - Abstraction: interfaces
  - Abstraction: abstract classes
  - Generics
  - Tuples
  - Array's
6. The final application (code)
7. Statistical reports and direct link to version control (GIT)
8. Demonstration video: a video in which you explain and show the application (3 minutes max.)
9. Team presentations (PowerPoints)

3. Evaluation

3.1 General evaluation

The project is a team effort, to which everyone contributes individually. The grade is given at a team level; so the whole team gets the same grade. To receive a grade for this project, you show a professional attitude approach concerning critical & analytical attitude. You need to meet the following individual prerequisites:

Individual:

- You meet the attendance requirements;
- You stick to the agreements in the cooperation contract (technical contribution, active team player, deadlines are met, etc.);
- Your presentation during sprint review needs to be sufficient;
- You adequately show to be capable of presenting through presenting a substantial part of the presentation Alpha project, mid-term demonstration, final presentation);
- You adequately show in-depth and comprehensive documentation in accordance with your scrum master role.

If, during the course of the project, you fail to meet any of the above-mentioned criteria, you will receive an official warning. You will then be given the opportunity to make the necessary changes to meet the missing criteria.

- If you succeed within the first delivery of the project, you will get a team grade;
- If you do not succeed to fix it in time, you receive a No Go/ND for the first try, and you will have to prove yourself in the resit;
- If you do not succeed in changing *and* it is not possible to fix, you will not be able to retake the project. This means you have to retake the project next semester. The P.O and tutor will decide whether your contribution is fixable or not.

If you fail to meet the attendance criteria, you will receive a No Go/ND. You will not be able to retake the project. This means you have to redo the project next semester.

3.2 Evaluation demonstration *(Rubric)*

Learning goal	Indicator	Very poor: 0	Insufficient: 1	Satisfactory: 2	Excellent: 3
2) The student knows the principles of the project methodology Scrum and can apply them within the group while working on the project. **20%**	*a) Applying scrum* ***20%***	The students can't show that they applied the elements that belong to the Scrum process.	The students show they applied the elements that belong to the Scrum process imperfectly.	The students show they applied the elements that belong to the Scrum process globally but sufficiently.	The students show they applied the elements that belong to the Scrum process in detail and sufficiently.
3) The student is able to set up a requirement specification for the console application based on the requirements of the Product Owner (PO).	*a) The application.* ***10%***	The students have not met the minority of the requirements as set out by the P.O. and have *not* met the expectations of the P.O. when meeting these requirements.	The students have met the minority of the requirements as set out by the P.O. and have *not* met the expectations of the P.O. when meeting these requirements.	The students have met the majority of the requirements as set out by the P.O. and have met the expectations of the P.O. when meeting these requirements.	The students have met all of the requirements as set out by the P.O. and have *exceeded* the expectations of the P.O. when meeting these requirements.

20%	*b) Modeling UML.* ***10%***	The students have a poor understanding of modeling in the form of Use Cases.	The students have little understanding of modeling in the form of Use Cases.	The students have sufficient understanding of modeling in the form of Use Cases.	The students have a full understanding of modeling in the form of Use Cases.
4) The student can develop a console-based application with attention to implementing technical concepts in the application. **40%**	*a) The use of technical concepts of OODP (see § 2.6 deliverables.* ***20%***	The students did not apply technical concept or did not use the three-layer model.	The students appropriately applied some of the technical concepts. The student used the three-layer model in some way but not good enough.	The students appropriately and consistently applied most of the technical concepts. The students used the three-layer model good enough.	The students appropriately and consistently applied all the technical concepts. The students implemented the three-layer model perfectly.
	b) The use of the three-layer architecture. ***20%***	The students did not apply the three-layer architecture.	The students did apply the architecture but made many mistakes in it. Showing no understanding the concept.	The students did apply the architecture, made some small mistakes in it. Showing understanding of the concept.	The students did apply the architecture flawlessly.
5) The student can improve the console-based application through feedback and testing to validate the quality of the final product. **20%**	*a) Test documents are complete tor proof the application has been tested.* ***10%***	The test documents are missing. There are no risk levels in the backlog.	The test documents are incomplete and does not cover all requirements. It is unclear how the user stories are tested.	The test documents are mostly complete and covers nearly all requirements. It is mostly clear how the user stories are tested.	The test documents are complete and covers all requirements. It is clear how the user stories are tested.
	b) The application has been improved after testing. ***10%***	The students did not make any improvements after testing.	The students only made some improvements after testing, only addressing some of the issues.	The students made significant improvements after testing, only addressing most of the issues.	The students made numerous significant improvements after testing, only addressing all issues.

3.3 Grading

The final grade will be determined by the Tutor and P.O. For every criterion your team can get 0, 1, 2 or 3 points. The evaluation form (Appendix B) will be completed by both teachers. You will receive the evaluation form via Teams/mail. If you have any questions about the evaluation, you can contact the teachers.

3.4 Cesure

- You pass the course if your individual score is *sufficient* and your team grade is *5.5 or higher.*
- If your individual score is insufficient, you can participate in the resit.
- If your individual score is insufficient because of attendance, you **cannot** participate in the resit.
- Your individual score must be sufficient in order to get your grade.
- If your team grade is insufficient (below 5,5), you can participate in the resit

3.5 Resit

In case of a final grade below 5.5 for project B or if you do not meet the prerequisites, but it is still fixable, you have to take part in the resit. You have to hand in the resit during week 20. The assignment for the resit is decided and communicated by the P.O. and Tutor. Before starting the resit make sure the assignment is clear. If you do not pass the resit, or if you get a ND because you do not meet the attendance criteria, you will have to take this course again next semester.

3.6 Compensation with OODP

You can compensate an insufficient grade from Project B or OODP with the other. If you pass the course with this compensation you're not eligible for a resit anymore. This can be done in only the following exact situations.
If your grade for Project B is a 5.0, but your OODP grade is a 7.0 or higher, you pass Project B.
If your grade for OODP is a 5.0, but your Project B grade is a 7.0 or higher, you pass OODP.

3.7 Test template

Learning goal	**Evaluated by**	**Evaluated through**	**Taxonomy (Miller)**
Individual			
1) You show a professional approach concerning critical & analytical attitude	Tutor & Product Owner	*All prerequisites have to be met:* *- Attendance* *- Professional attitude* *- Presentation* *- Documents in accordance with scrum / scrumaster role*	Shows how

Learning goal	**Evaluated by**	**Evaluated through**	**Percentage**	**Taxonomy (Miller)**
Team				
2) The student knows the principles of the project methodology Scrum and can apply them within the group while working on the project.	Tutor & Product Owner	*- All artifacts of Scrum*	20%	Shows how
3) The student is able to set up a requirement specification for the console application based on the requirements of the Product Owner (PO).	Tutor & Product Owner	*- The final code* *- Demonstration video* *- Final presentation*	20%	Knows how/shows how
4) The student can develop a console-based application with attention to architecture and implementing technical concepts in the application.	Product Owner	*- Demonstration of the latest working version on their local dev. environment* *- Demonstration of commit changes to the GIT repository* *- The implementation of the architecture in the code* *- The final code*	40%	Knows how/shows how
5) The student can improve the console-based application through feedback and testing to validate the quality of the final product.	Tutor & Product Owner	*- The necessary test documents.* *-Final presentation*	20%	Knows how/shows how

Appendix A

Evaluation Form Project B

The evaluation form is in a excel document. In the excel document there is one worksheet for the individual phase and a second worksheet for the team phase.

The excel document is provided at the same place as this file.