Module Description Software Testing

Key words: testing, quality assurance

Target group:	**6th semester SWB**	**module number:**	**SWB 105 6043**
Workload:	**5 ECTS**		**150 h**
of which	**Contact time**		**60 h**
	Self-study		**60 h**
	Exam preparation		**30 h**
Language of instruction:	**English**		
Module coordinator:	**Prof. Dr. rer. nat. Mirko Sonntag**		
Version:	**01.03.2023**		

Recommended prerequisites:

Principles of software engineering and knowledge of an object-oriented programming language.

Objective of module – intended learning outcomes:

Students will be able to ensure the fulfilment of functional and non-functional software requirements through quality assurance measures.

Knowledge – technical competences

The students know:

- Different levels and methods of software testing
- The benefits and costs of automating tests
- The effects of software errors and the importance of product quality
- Organisation and management of the quality assurance of software systems
- Essential norms, standards and processes for the development of software systems
- Tasks, methods and techniques of quality assurance for software systems
- Comprehensive overview of all activities related to the testing of software systems
- Basic aspects of tool support and automation of tests

Skills – methodological competences

The students are able to

- Design software requirements and tests at different levels
- Integrate (automated) tests into the development process
- Perform system tests
- Select and apply important static and dynamic quality assurance measures in a targeted manner
- Develop the test strategy for safety-relevant software systems
- Set up a test plan based on suitable verification and validation methods
- Perform various roles in a testing organisation

General competences

Students can use specifications, reviews and tests to ensure that high-quality software systems are developed.

Contents:

- Motivation for quality assurance and testing
- Test automation, test documentation, test management
- Test tools
- Black box and white box testing
- GUI tests
- Innovations, trends and technologies - potential and challenges
- System design using the example of EE systems and SW in the automotive industry
- Real-time capability, system reliability, fault management and functional diagnostics
- Motivation for quality assurance and testing
- Terms, standards, norms, basics of system and software validation
- Constructive and analytical quality assurance measures
- Unit tests, integration tests, system tests and acceptance tests
- Specification of requirements and test cases
- Development and test processes and special features of today's software assurance (e.g. in an agile development context using the example of Scrum)

Literature:

- Baumgartner et al.: Agile Testing, Hanser, 2. edition, 2018
- Lisa Crispin, Janet Gregory: Agile Testing: A Practical Guide for Testers and Agile Teams, Addison-Wesley, 2008
- Eran Kinsbruner: The Digital Quality Handbook: Guide for Achieving Continuous Quality in a DevOps Reality, Infinity P, 2017
- Jonathan Rasmusson: The Way of the Web Tester, O'Reilly, 2016
- Andreas Spillner: Aus- und Weiterbildung zum Certified Tester - Foundation Level nach ISTQB®-Standard, dpunkt Verlag, 2012

Offered:

each semester

Lecture parts and proof of performance:

Teaching and learning format:	lectures with exercises and project work
Examination:	written exam (90 minutes)
Weekly semester hours (45 mins):	2 SWS
Approximate workload of student:	90 hours

Learning objective:

Students master software testing as the most important means of quality assurance. They will be able to consider the costs, benefits and limitations of software tests when developing test concepts and develop tests independently.

Lecture parts and proof of performance:

Teaching and learning format:	lab exercises
Examination:	demonstration of lab results
Weekly semester hours (45 mins):	2 SWS
Approximate workload of student:	60 hours

Learning objective:

Students will be able to implement tests at different levels of the test pyramid and integrate them into the Continuous Integration Pipeline. In addition, they will be able to perform tool-supported system tests to ensure compliance with functional and non-functional software requirements.

Grading:

lecture: grades from written exam, lab: pass/fail