Faculty of Engineering & Information Technology

Software Engineering

Introduction

Asst. Prof. Dr. Ahmed A.O. Tayeh

Contact Details

- If nothing urgent, please contact me via emails
  - atayeh@israa.edu.ps
  - expect a reply in 24 hours
- Office Hours
  - Sunday 8 – 11 AM
  - Sunday 1 – 3 PM
  - Monday 8 – 10 AM
- Do not hesitate to discuss things during lecture breaks
- If questions are related to course topics, raise them during the lectures so others can learn

Prerequisites

- No official prerequisites declared, *nevertheless, students should have*
  - learned basic programming skills
  - built or participated in building simple or complex software projects
  - have learned **O**bject **O**riented **P**rogramming (**OOP**) concepts
  - …

Course Topics

- Introduction
- Software Project Management & Planning
- Software Processes
- Requirement Engineering & System Modelling
- Component-based, Distributed & Service-Oriented Software Engineering
- Design & Implementation
- Testing and Evolution
- Reliability, Safety, Security & Resilience Engineering
- Software Reuse

Study Material

- Slides are the main study material
- Lecture discussions and notes should be considered
- Reference Book "*Software Engineering*" *tenth edition by Ian Sommerville*
  - course topics are not covered with the same order as the book
  - keep the book or its future editions as a reference for your software engineering career
  - anything that is not covered during the lectures, are not part of the exam
- Study material uploaded before each lecture at the study portal and the course repository GitHub account
  - https://github.com/atayeh-israa-university/SoftwareEngineering-2023

Acknowledgment

- Course slides are largely based on the slides prepared by the authors of the course main book
  - "*Software Engineering*" *tenth edition by Ian Sommerville*

- Other content and information are taken from other sources and explicitly referenced in their corresponding slides

- The slides and topics are carefully chosen and organised by the teacher of this course, Ahmed A.O. Tayeh, to meet the learning objectives of this course
  - Organisation of the topics do not match the reference book

Grading

- Midterm exam 30%
- Exercises, presence & project 20%
- Final exam 50% (*10-20% - might be more - of the final exam will touch your team project*)

Project

- Students will engineer 5 university systems
  - HR System
  - Student Admission System
  - Academic Portal
  - Inventory & Logistics System
  - University Public Website
- Students are divided into 5 groups, representing 5 different teams in the same organisation
  - each team should choose one project to engineer
  - each team member plays a specific role in their team
- We define the deliveries and tasks of each team at the end of each lecture

Project…

- Systems are somehow integrated with each other to exchange information & data
  - communication among teams are facilitated by Business Analysts (BAs) & Project Managers (PMs) appointed in each team
- Deliveries should be committed to each team’s GitHub repository
  - each student must have a GitHub account and proves that they contribute to the teamwork
  - each team will present their work in each lecture
  - each team member should present his own task and progress
- *If you do not contribute to your teamwork, you should not expect the same grade as your mates!*

Project…

- You are not writing code to produce a working software
- Modern software engineering practises is based on iterations and evolution
  - you will not produce good results from the first iteration!
- We are not seeking perfect designs
  - learn by doing things wrongly. Do mistakes and learn (the more mistake you do, the more you learn, the more grades you might get!)
- Learn by experience and do not reinvent the wheel
  - check if others have engineered the same software
  - do not copy paste others work, if you do so justify and credit others

Introduction

Fundamental Concepts

- A Software
  - computer programs and associated documentation
  - software products may be developed for a particular customer or may be developed for a general market
- Software Engineering
  - an *engineering discipline* that is concerned with all aspects of software production
    - technical process, project management, development tools, methods, support of software production, maintenance and evolution
- Engineering discipline
  - using appropriate theories and methods to solve problems
    - bearing in mind organizational and financial constraints

Fundamental Concepts…

- Fundamental Software Engineering Process Activities
  - *Software Specification*
    - customers and engineers define the software that is to be produced and the constraints on its operation
  - *Software Development*
    - the software is designed and programmed
  - *Software Validation*
    - the software is checked to ensure that it is what the customer requires
  - *Software evolution*
    - the software is modified to reflect changing customer and market requirements

Fundamental Concepts…

- Software Engineering VS Computer Science
  - Computer Science focuses on theory & fundamentals
  - Software Engineering concerned with practicalities of developing and delivering useful software
- Software Engineering VS System Engineering
  - System Engineering concerned with all aspects of computer-based systems development including hardware, software and process engineering
  - Software Engineering is part of System Engineering

Fundamental Concepts…

- Costs of Software Engineering?
  - roughly 60% of software costs are development costs
  - 40% are testing costs
  - maintenance & evolution costs!
    - evolution costs often exceed development costs
- Best Software Engineering techniques and methods
  - different techniques are appropriate for different types of systems
  - no method is better than another
  - cannot find “one size fits all” technique nor method
- Web Impact on Software Engineering
  - availability of software services
  - possibility of developing highly distributed service-based systems
  - advances in programming languages and software reuse

Software Products

- Generic products
  - stand-alone systems that are marketed and sold to any customer who wishes to buy them
  - graphics programs, project management tools; document editing software
  - specification is owned by the software developer
    - decisions on software change are made by the developer
- Customised products
  - software that is commissioned by a specific customer to meet their own needs
  - embedded control systems, air traffic control software, traffic monitoring systems
  - specification is owned by the customer of the software
    - customers make decisions on software changes that are required

Software Essential Attributes

- *Maintainability*
  - software can evolve to meet the changing needs of customers
  - software change is an inevitable requirement of a changing business environment
  - maintainable software? good software architecture, design patterns, OOP, clean code, documented code, etc.
- *Dependability and Security*
  - includes a range of characteristics including reliability, security and safety
  - should not cause physical or economic damage in the event of the system failure
  - malicious users should not be able to access or damage the system

Software Essential Attributes

- *Efficiency*
  - should not make wasteful use of system resources such as memory and processor cycles
  - includes responsiveness, processing time, memory utilisation (Data Structures & Algorithms!)
- *Acceptability*
  - must be acceptable to the type of users for which it is designed
  - must be understandable, usable and compatible with other systems that they use

General Issues Affecting Software

- Heterogeneity
  - systems are required to operate as distributed systems across networks that include different types of computer and mobile devices
- Business and social change
  - changing incredibly quickly as emerging economies develop and new technologies become available
  - need to be able to change their existing software and to rapidly develop new software

General Issues Affecting Software…

- Security and trust
  - trust software with all customers data and valuable assets
- Scale
  - software must be developed across a very wide range of scales
    - very small embedded systems in portable or wearable devices
    - Internet-scale, cloud-based systems that serve a global community

Application Types

- Stand-alone applications
  - that run on a local computer, such as a PC
  - include all necessary functionality and do not need to be connected to a network
  - Microsoft Word, Notepad, PDF readers
- Interactive transaction-based applications
  - execute on a remote computer and are accessed by users from their own PCs or terminals
  - include web applications such as e-commerce applications
- Embedded control systems
  - control and manage hardware devices
  - numerically, there are probably more embedded systems than any other type of system

Application Types…

- Batch processing systems
  - business systems that are designed to process data in large batches
  - process large numbers of individual inputs to create corresponding outputs
- Entertainment systems
  - primarily for personal use and which are intended to entertain the user
- Systems for modelling and simulation
  - developed by scientists and engineers to model physical processes or situations
  - include many, separate, interacting objects

Application Types…

- Data collection systems
  - collect data from their environment using a set of sensors
  - send that data to other systems for processing
- Systems of systems
  - are systems that are composed of several other software systems

Software Engineering Ethics

- Software engineering involves wider responsibilities than simply the technical skills
- Software engineers must behave in an honest and ethically responsible way
- Ethical behaviour is more than simply upholding the law but involves following a set of principles that are morally correct

Professional Responsibility

- Confidentiality
  - engineers should normally respect the confidentiality of their employers or clients
  - confidentiality should be respected irrespective of whether a formal confidentiality agreement has been signed
- Competence
  - engineers should not misrepresent their level of competence
  - should not knowingly accept work beyond their competence
- Intellectual property rights
  - engineers should be aware of local laws governing the use of intellectual property such as patents, copyright, etc.
  - they should ensure that the intellectual property of employers and clients is protected

Professional Responsibility…

- Computer misuse
  - engineers should not use their technical skills to misuse other people’s computers
  - computer misuse ranges from relatively trivial (game playing on an employer’s machine, say) to extremely serious (dissemination of viruses)

The ACM/IEEE Code of Ethics

1. **PUBLIC** - Software engineers shall act consistently with the public interest.

2. **CLIENT AND EMPLOYER** - Software engineers shall act in a manner that is in the best interests of their client and employer consistent with the public interest.

3. **PRODUCT** - Software engineers shall ensure that their products and related modifications meet the highest professional standards possible.

4. **JUDGMENT** - Software engineers shall maintain integrity and independence in their professional judgment.

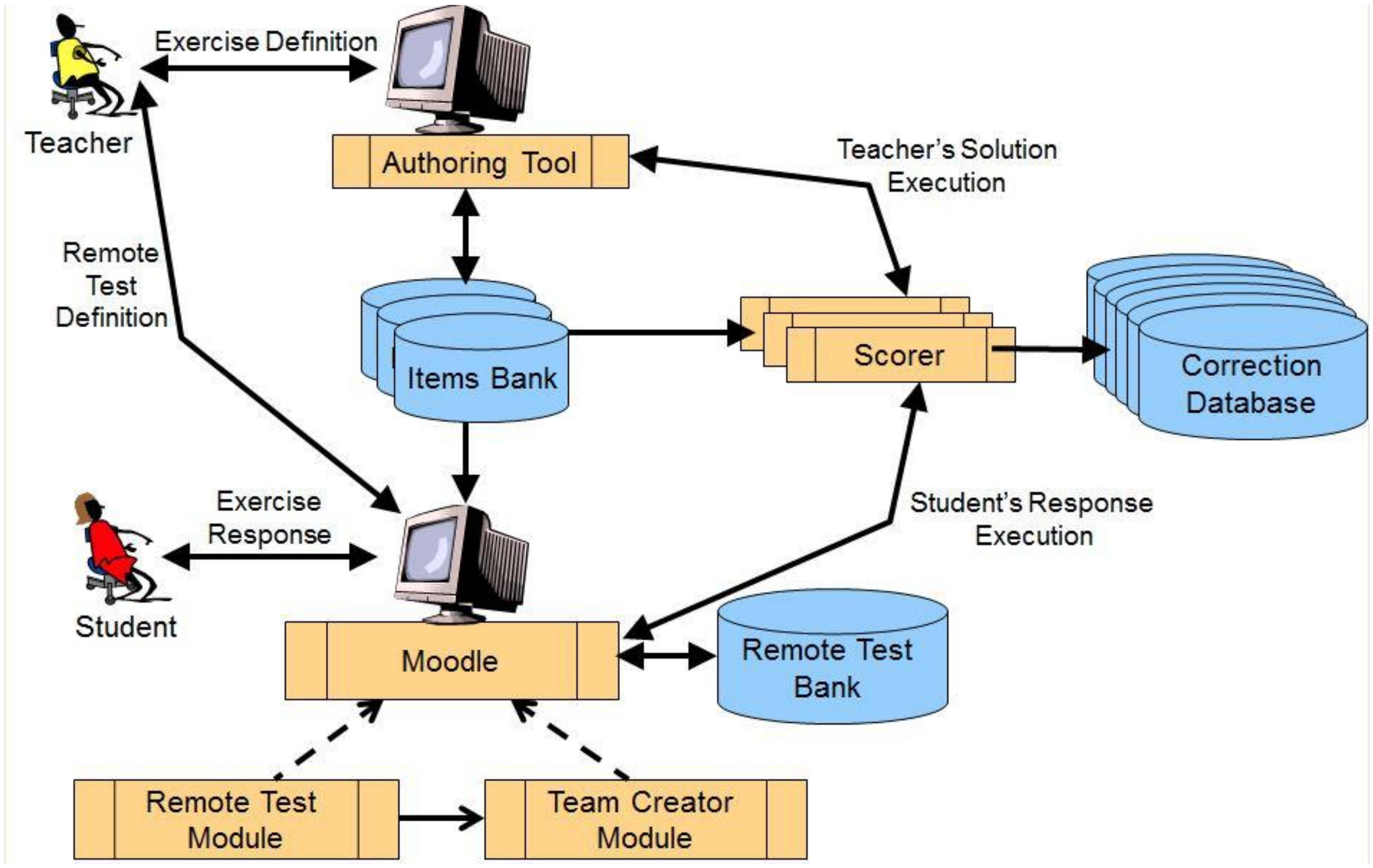
5. **MANAGEMENT** - Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance.

6. **PROFESSION** - Software engineers shall advance the integrity and reputation of the profession consistent with the public interest.

7. **COLLEAGUES** - Software engineers shall be fair to and supportive of their colleagues.

8. **SELF** - Software engineers shall participate in lifelong learning regarding the practice of their profession and shall promote an ethical approach to the practice of the profession.

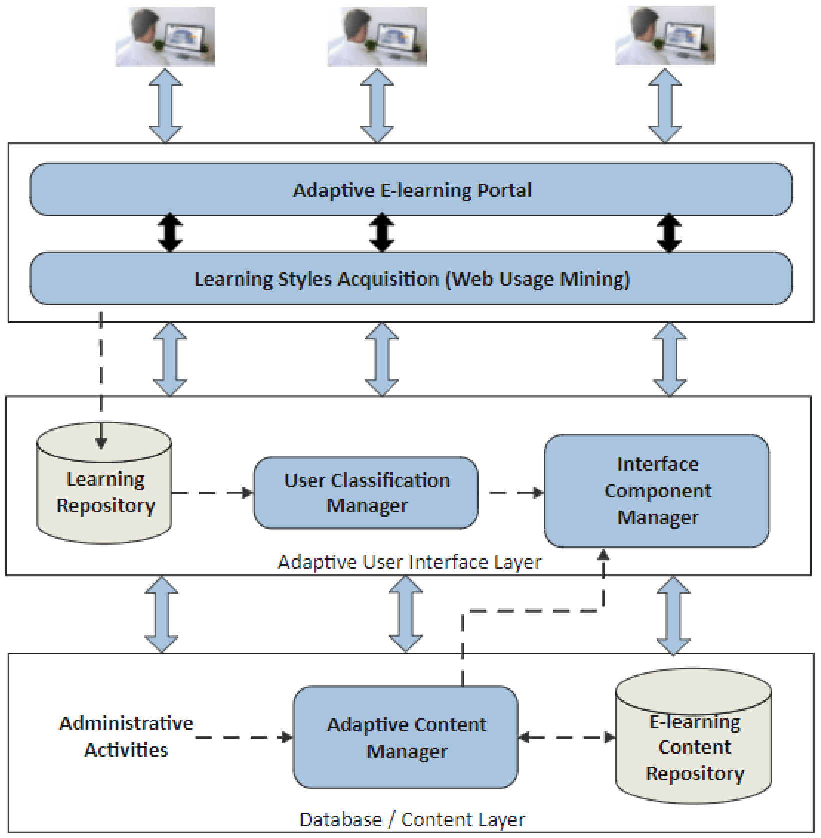
Case Study – eLearning Platform



eLearning platform high level architecture

Source: https://www.edrawsoft.com/article/system-architecture-diagram.html

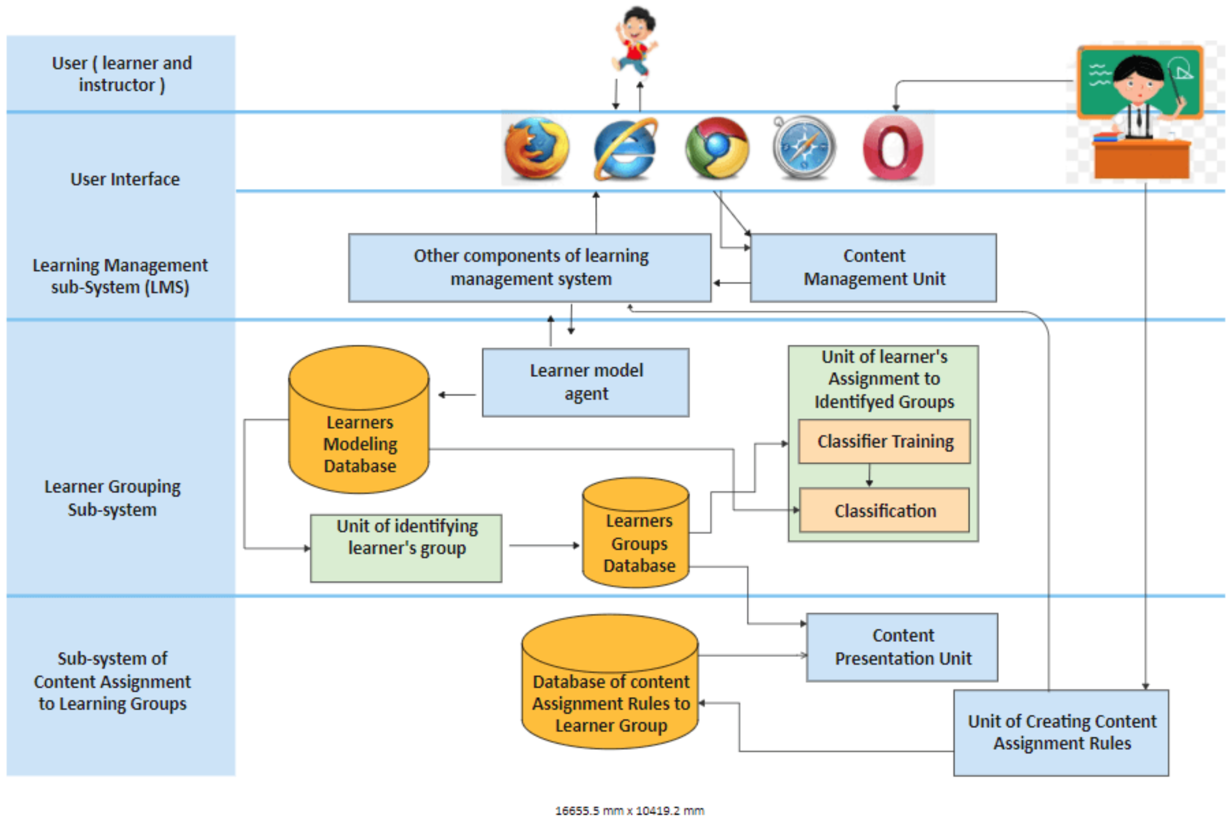
Case Study – eLearning Platform…


Adaptive E-learning Portal
Learning Styles Acquisition (Web Usage Mining)
Learning Repository
User Classification Manager
Interface Component Manager
Adaptive User Interface Layer
Administrative Activities
Adaptive Content Manager
E-learning Content Repository
Database / Content Layer

eLearning platform business architecture

Source: https://www.edrawsoft.com/article/system-architecture-diagram.html

Case Study – eLearning Platform…



User (learner and instructor)
User Interface
Learning Management sub-System (LMS)
Other components of learning management system
Content Management Unit
Learner Grouping Sub-system
Learner model agent
Learners Modeling Database
Unit of learner's Assignment to Identifyed Groups
Classifier Training
Classification
Unit of identifying learner's group
Learners Groups Database
Sub-system of Content Assignment to Learning Groups
Content Presentation Unit
Database of content Assignment Rules to Learner Group
Unit of Creating Content Assignment Rules
16655.5 mm x 10419.2 mm

eLearning platform system architecture

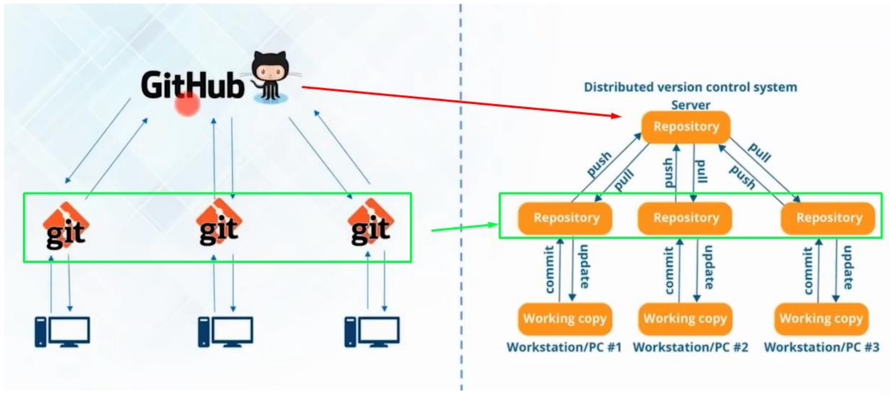
Source: https://www.edrawsoft.com/article/system-architecture-diagram.html

Case Study – eLearning Platform…

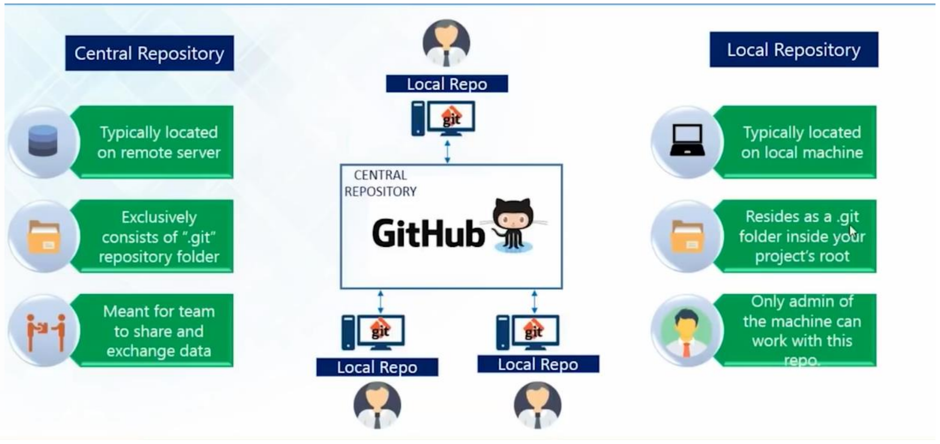
- Designs might get updated via iterations and revisions
- Define development stack and hire missing resources
  - technologies, software architectures, frameworks, deployment strategies, etc.
- Define testing strategies and release processes
- How the team should be organised to achieve the goals and meet deadlines?
  - which processes and methodologies to be used?
- How to define the risks logs and mitigate them?
- How to evolve and maintain the application?

Course Project

- Teams' organisation and systems assigned
- Each team member should create a GitHub account
  - check basic tutorials for Git and GitHub.
  - teacher will demo the basic concepts, if there is internet connection



Source: "https://stackoverflow.com/questions/13321556/difference-between-git-and-github"


Central Repository
Typically located on remote server
Exclusively consists of ".git" repository folder
Meant for team to share and exchange data
Local Repo
CENTRAL REPOSITORY
GitHub
Local Repo
Local Repo
Local Repository
Typically located on local machine
Resides as a .git folder inside your project's root
Only admin of the machine can work with this repo.

Source: "https://stackoverflow.com/questions/13321556/difference-between-git-and-github"

Course Project…

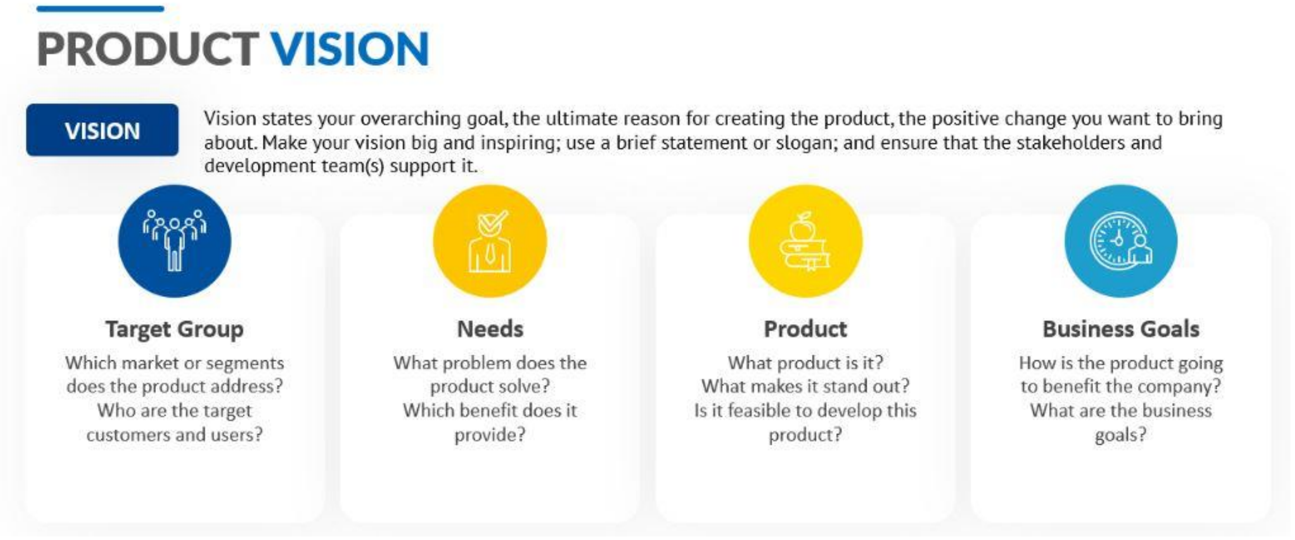
- One team member creates a repository for their project
  - name the repo (SE-2023-[SystemName])
  - make the repository public; other teams can see your documentation
  - add your teammates are collaborators
  - add this account (atayeh-israa-university; email: atayeh@israa.edu.ps) as a collaborator
- Appoint an enterprise architect and technical leaders for the organisation (if necessary)
- Choose a timeslot on Sundays/Mondays for general discussions and guidance

Course Project…

- Software projects starts from ideas or business needs
- Project managers and business managers start a project by drafting a “Product Vision”
- Product Vision specifies
  - Target group
    - which organisation/market the product should serve? Who are the customers?
  - Needs
    - what problems the product solves? What are the benefits?
  - Product
    - What is the product? Why it is important? Is it feasible to develop it?
  - Benefit goals
    - How the product benefits the organisation? What are the business goals?

Course Project…

- How to write a product/software vision?
- Fill in the template, not necessarily with the same format!



PRODUCT VISION

VISION — Vision states your overarching goal, the ultimate reason for creating the product, the positive change you want to bring about. Make your vision big and inspiring; use a brief statement or slogan; and ensure that the stakeholders and development team(s) support it.

Target Group	Needs	Product	Business Goals
Which market or segments does the product address? Who are the target customers and users?	What problem does the product solve? Which benefit does it provide?	What product is it? What makes it stand out? Is it feasible to develop this product?	How is the product going to benefit the company? What are the business goals?

Product Vision template
Source: "https://powerslides.com/powerpoint-marketing/analytics-templates/product-vision/"

Course Project…

- Week Sep 30 – Oct 5
  - GitHub accounts and repository settings
  - add to readme file the names of the team
  - PM and BA start drafting the product vision with the help of all team members
    - Not expected to be “finished” during this week
- Team communication should be reported
  - recommended to use business recommended tools (Teams, Slack, etc), use WhatsApp if this more comfortable
  - for face-to-face meetings use Google Meet, Zoom, or Teams
  - document your meetings in the project documentation with the meeting information (time, and attendees)
    - take screenshot of the attendee's info from the used tool, add the screenshot to the report Annexes

Thank You!

Next Lecture: Software Project Management & Planning