Veebirakenduse projekt (ITI0302)

Sissejuhatus

Siim Rebane

Introduction

- Lecturer: Siim Rebane
- Approximately 20 years of experience in software development
- Currently working full time as CTO in Lingvist

Course description

- Build modern web application
  - Java
  - Vue JS (can use Angular or React)
  - CI/CD
  - Do it in teams (up to 3 students)
  - Choose own topic (or choose from lecturer topics)

Lessons and materials

- Lessons
  - 1st part is lecture. Starts 14:15. Approximately 90 minutes
  - 15 minutes pause
  - 2nd part more practical samples
- All data in Moodle
  - https://moodle.taltech.ee/course/view.php?id=38359
  - Code: ITI0302-2026
  - Recordings (Echo360)

Project topic

- Web application
- Backend heavy
- Must use a database
- Some form of login
- Some form of integration

Final project — what it must have

The application

- Backend: Java + Spring, REST API
- Frontend: Vue / React / Angular — uses all the endpoints
- Database, schema managed with Liquibase
- Authentication
- Pagination, filtering, sorting

The engineering

- Layered architecture, DTOs
- Tests that actually validate behavior
- CI/CD pipeline, app running in Docker on your server
- Reverse proxy + domain name
- README + wiki documentation
- Clean git history: branches, merge requests

The exact checklist with numbers is published with the milestones — it adds detail to this list, it doesn't change the shape.

Prerequisites

- Basic programming skills (preferably in Java)
  - If you don't have it you can still declare
  - I help as much as I can, but lecture material expects basic understanding of software development
- Basic database knowledge — SQL, tables, primary and foreign keys
  - Most of you have passed a databases course
  - We will go over what this course uses (JPA, Liquibase), but SQL is not taught from zero

Grading

- Individual assignments: 2 × 10 points = 20 points
- Project milestones: 2 × 10 points = 20 points
  - Week 8 milestone: 10 points
  - Week 12 milestone: 10 points
- Written test: 30 points (must score ≥ 15 to pass)
- Final project defense: 30 points (must score ≥ 15 to pass)

Grade scale

Points	Grade
91–100	5
81–90	4
71–80	3
61–70	2
51–60	1
0–50	0

Semester deadlines

All submissions are due Sunday 23:59 of the given week.

Week	What	Points
5	Individual assignment 1	10
8	Milestone 1 (team)	10
11	Individual assignment 2	10
12	Milestone 2 (team)	10
14–16	Final defense	30
15–16	Written test	30

Individual assignments

- 2 weeks for each assignment — counted from when all the needed material has been covered in lectures
- 10 points each
- Deadlines:
  - Assignment 1 — week 5 (everything needed is covered by week 3)
  - Assignment 2 — week 11
- Feedback
  - In written form, defense not required

Milestones

- **Milestone 1** (at the end of 8th week):
  - Some backend services up and running
  - With database
  - Some simple frontend forms done
  - Feedback in written form, defense not required
- **Milestone 2** (at the end of 12th week):
  - Authentication implemented
  - CI setup done
  - Some frontend views done
  - Feedback in written form, defense not required
- **Final defense** (14th–16th week)
  - Everything ready
  - Defense for whole project done individually

Late submissions

- Miss the deadline: −3 points
- Miss the deadline > 1 week: −6 points
- Miss the deadline > 2 weeks: no points

Test

- On site
- On paper
- Week 15–16

Project defense

- You submit a project as a team
  - Quality of your project sets maximum points
  - Individual defense shows what percentage of the maximum points you get
- Project defense consists of 2 parts:
  - Showing / explaining code
  - Practical task
- Practical task
  - You need to do some changes into code
  - It might be your project or something new that you need to check out from git

How to lose points

- Your contribution for the whole project was very small
  - We actually check git logs
- You can't explain how code works
  - You need to understand your teammates' (or ChatGPT) code as well
- You can't write code on site
  - You need to be able to write it on your own

Materials

- https://javadoc.pages.taltech.ee/ — to refresh Java
- https://webdoc.pages.taltech.ee/ — our course materials

GitLab

When creating git repos in GitLab please name them:

- Individual repo (for individual assignments): **iti0302-2026**
- Project repo (multiple projects):
  - **iti0302-2026-backend**
  - **iti0302-2026-frontend**
- Project repo (single project):
  - **iti0302-2026-projekt**

Git workflow

- `main` must always be working — don't develop on it directly
- Work in **feature branches**: one branch per task/feature
- Every finished task comes to `main` through a **merge request**
- Teams of 2–3: a **teammate reviews** the merge request before merging
- Working alone: feature branches and merge requests still required — no committing everything straight to `main`
- From week 3: a merge request can be merged only with a **passing CI pipeline**
- Commit messages describe what changed — "fix", "asd", "final2" tell us nothing
- This is graded: at the defense we look at branches, merge requests and history, not just the end result

Your own server

- Every student gets a **small virtual server** for the whole semester (from week 2)
- Individual assignment 1 is deployed on your own server
- The team project is deployed on **one** of the team members' servers — the team picks which
- To get access: **submit your SSH public key in Moodle** (assignment "SSH key") before the week 2 lab
  - Submit the `.pub` file contents, never the private key
  - How to generate one is covered in Lab 1

LLM

- Using LLMs is permitted and even encouraged
  - Results matter, not how you got to the result
- You are 100 % responsible for the code you submitted
  - It is your responsibility to review all the code generated by LLMs
  - You need to fully understand what's going on and defend the code as if it's yours
  - LLM is a tool, you are the engineer and you are responsible for what you submit

Thesis and supervisor

- I have been supervisor for bachelor theses
- Some of the projects have started or have been developed in this lecture
- I only supervise practical work and no real research
- The thesis topic can also come from your employer — something you are dealing with at work
- Sample theses I have supervised:
  - Veebirakendus lühianimafilmide süžee arendamiseks
  - Avaliku rakendusliidese loomine Eesti Jalgpalli Liidule
  - Veebipõhine finantsanalüüsi platvorm

Common answers to common questions

- Team size of 1 is allowed, but you will still have to fill all requirements.
- Biggest reason why teams failed was a big discrepancy of skill level.

Expectations

- If there are problems and you can't complete something on time, please inform me **before** the deadline, not after. Getting extensions for valid reasons is not a problem as long as it's communicated on time.
- I expect you to participate in all parts of development — frontend, backend and setting up servers. One team member doing only frontend and one only backend will reflect badly on your grade.
- I expect you to be responsible for your time management.
- If there is a problem with teamwork, let me know. Joining another team / splitting a team etc. is acceptable later on as well.
- If some team member leaves or doesn't do their work, again let us know. I can maybe help with rearranging teams — or just know what's going on when evaluating work.

Questions

???