Soreen, Lev, Viswak, Olga

Attendance Marker Project Plan

An application for teachers to mark attendance of their students

Metropolia University of Applied Sciences
Information and communication technologies
Software Engineering
Software Engineering project 1
20.1.2026

Project Overview

aTendy is an attendance marking platform that is used by teachers and students to mark attendance for each lecture, aTendy helps teachers avoid using excel sheets and shouting names one by one when marking the students attendance, as well as minimizing time taken to mark the students currently at the class as well as the ones who arrive later, our platform shifts the responsibility of marking attendance from the teacher to the students which will save time for teachers.

Sometimes teachers forget to mark the attendance of the students, that might be okay for some courses, however for courses which depend on the attendance percentage when grading it could be a huge problem, moreover students don’t usually come all at the same time but they arrive in different intervals, having the teacher go each time to mark their attendance and disturb the class might affects the teachers productivity negatively.

Our solution provides the teacher with one click action at each lecture to generate a QR code and put it on the screen, then students will scan the QR code whenever they arrive without having to interrupting the teacher, teachers could create courses and import students to the courses through excel sheet, then each lecture will have a QR code students can scan and mark their attendance to, all the data is being stored in a database and the teacher has access to lecture wide statistics as well as course wide.

Students on our platform will have to create account and verify it with their student ID, when student scan a QR code, we will have a check that ensures their existence at the campus set by the teacher. That way we prevent the user from scanning while at home.

aTendy consists of two main components which are the student and the teacher, and features concerning each components are as follows:

Teacher Component:

- Teacher can create account or login to existing one.
- Teacher can create class and add a list of students to it.

- Teacher can create lectures for that class and mark students attendance on that lecture.
- Teacher will have a statistical view of main metrics i.e student engagement levels and arrival intervals.
- Teacher can set campus for a specific lecture so that students will not be allowed to attend outside the campus.
- Teacher can create a numeric code or QR code for each lecture for students to scan.

Student component:

- Student can create create and verify account or login.
- Student can see the courses they are enrolled in.
- Students can scan QR code or enter the lecture numeric code to mark their attendance.
- Students can view their attendance history.

Project Objectives

The main objectives of aTendy project:

- **Reduce time spent on attendance marking by 80%:** aTendy focuses on shifting the responsibility of marking attendance from the teacher to the students, which will save the teacher 80% of the time taken to mark every student, the rest 20% will be spent on initial setup and manual edits if any needed.
- **Increase teacher productivity by 30%**: by shifting the responsibility to students, we ensure the teacher focuses 100% on the lecture without any interruptions by newly arrived students. When every student arrives marks their own attendance, this will keep the teacher focused 100% on the lecture.
- **Reduce errors in attendance marking process by 70%:** teachers usually tend to focus more on the lecture which might lead to forgetting to mark the attendance of their students or when students arrive after the marking process they might be left unmarked, which will affect the grade for the courses that depend on the attendance percentage as part of the grading process.
- **Support teacher’s decision making process when grading by 40%**: our platform provides the teacher with valuable statistical data about each students

engagement rate, arrival times, attendance percentages and more, which will help the teacher make decisions when grading students.

Scope and Deliverables

Scope

The scope of the project includes the development of a web-based attendance marking system.

In scope:

- A relational database for storing users, courses, campuses, rooms, and teaching sessions.
- A teacher/admin interface for managing students, rooms, and sessions.
- Authentication for teachers and students using, with access restricted to users signing in with an institutional student email address.
- A teacher view for selecting a session and generating a QR code or numeric attendance code.
- A student view for entering the attendance code or accessing the system via a scanned QR code.
- Attendance validation and presence handling.
- A statistics view for teachers showing attendance data for the current day.
- A backend API to support communication between the frontend and backend.
- Basic authentication and role-based access control.

Out of scope:

- Mobile applications.
- Student dashboards with personal attendance statistics.
- Scheduling or calendar functionality.
- Aggregated attendance statistics beyond the current day.

Deliverables

- A working web-based MVP of the attendance system.
- Backend and frontend source code stored in a GitHub repository.
- A configured database schema.

- Project documentation and a final project presentation/demo.

Project Timeline (Sprint-Based)

The project is developed iteratively using a Scrum-based approach and follows the official sprint schedule of the course.

Sprint 1 (14.01 – 28.01): Project Setup and Planning

- Team setup and role distribution.
- Final project topic selection.
- Definition of project scope, requirements, and MVP goals.
- Creation of the project plan and project vision.
- Initial environment setup and repository configuration.

Sprint 2 (28.01 – 11.02): Core Foundations

- Definition of system architecture (frontend, backend, database).
- Implementation of initial database schema.
- Setup of core backend structure and essential API endpoints.
- Initial implementation of authentication and role-based access (teacher/student).
- Preparation of Review 1 materials (Sprint Review Report) and Sprint Planning.

Sprint 3 (11.02 – 04.03): Feature Implementation and CI

- Implementation of core MVP features (teacher/admin pages, session selection, QR/numeric code generation, student attendance flow).
- Unit tests for critical backend functionality.
- CI integration (automated build and tests).
- Preparation of Review 2 materials (Sprint Review Report) and Sprint Planning.

Sprint 4 (04.03 – 11.03): Deployment and MVP Finalization

- Containerization of the application using Docker.
- Deployment to a remote/cloud environment.

- Bug fixing, stabilization, and final testing.
- Finalization of the MVP and preparation for the final demo.

Final Presentation (11.03)

- Final project presentation and live demo of the MVP.

Resource Allocation

- **Team Members and Roles:**
  - Scrum master: Helps organize teamwork, coordinates Scrum activities, and keeps track of submission deadlines.
  - Fullstack software developers: Each team member is responsible for developing specific functionalities, covering both backend and frontend development. This approach was intentionally chosen to reduce communication risks and misunderstandings within the team, as well as to support educational goals by allowing each member to gain experience across the full technology stack. Additionally, team members participate in code reviews before any developed functionality is merged into the main branch.

- **Software, Hardware, and Tools:**
  - Project Management: GitHub Projects
  - Backend: Java, Spring Boot
  - Database: relational DB: e.g. PostgreSQL, MariaDB
  - Frontend: React
  - Testing: JUnit, manual testing, Postman
  - CI/CD: Jenkins
  - Version control: GitHub

- **External Resources or Support:**
  - Agile Scrum framework
  - University lectures and the teacher support

    - Open-access study resources such as online articles and videos

Risk Management

Potential risks:

- Too ambitious project goals (low)
    - Impact**:** Not all planned functionalities may be completed within the project timeline.
    - Mitigation Strategy: Prioritize core features and ensure that essential functionalities are implemented first before optional features.
- Bad tasks planning (low)
    - Impact: Tasks may overlap, resulting in multiple team members working on the same functionality or unclear task ownership.
    - Mitigation Strategy: Clearly define tasks in GitHub Projects, assign a single owner per task, and review task allocation during sprint planning meetings.
- Misunderstanding in the team (low)
    - Impact: Some team members may misunderstand requirements or implementation details, leading to inconsistencies or delays.
    - Mitigation Strategies: Maintain clear documentation, use UML diagrams where applicable, and ensure regular communication through team chat and meetings.
- Underestimating project complexity (medium)
    - Impact: As beginner programmers, the team may underestimate the time and effort required to implement certain functionalities, leading to schedule delays.
    - Mitigation Strategies:
- Technical skill gaps (medium)
    - Impact: Development may slow down, and some tasks may need to be carried over to the next sprint.
    - Mitigation Strategies: Use technologies covered in the course materials to have an opportunity to get support from the teacher. Select additional libraries with good documentation.

Testing and Quality Assurance

Testing and quality assurance are carried out throughout the development lifecycle to ensure the reliability, correctness, and usability of the aTendy platform. A combination of automated testing, end-to-end testing, and manual testing approaches is used to validate both backend and frontend functionalities.

Unit testing is performed on critical backend components such as authentication, attendance validation, QR code generation, and database operations using JUnit. These tests ensure that individual modules function as expected and help prevent regressions during development. Backend API endpoints are tested using Postman to verify request handling, response correctness, and error scenarios.

For the frontend, end-to-end (E2E) testing is implemented using a modern testing framework such as Cypress or Playwright. These tests simulate real user interactions in the browser and validate complete user flows, including login, course selection, lecture creation, QR code scanning, and attendance submission. E2E testing ensures that the frontend integrates correctly with the backend and that critical user journeys work as intended from start to finish.

In addition to automated testing, manual testing is conducted to evaluate usability, responsiveness, and edge cases such as invalid attendance codes, late arrivals, and location-based attendance restrictions. During the final sprint, full system testing is performed after deployment to ensure all components work together correctly in a production-like environment

Documentation and Reporting

Documentation is maintained continuously throughout the project to support development, collaboration, and evaluation. Project documentation includes the project plan, system architecture descriptions, database schema, API documentation, and setup instructions for running the application locally or in a deployed environment.

All source code is version-controlled and documented within a GitHub repository, with meaningful commit messages and inline code comments where necessary. GitHub

Projects is used to track tasks, sprint progress, and issue resolution, ensuring transparency and clear task ownership within the team.

Regular reporting is carried out in alignment with the Scrum-based development approach. Sprint planning documents, sprint review reports, and progress updates are prepared according to the course requirements. A final project report and presentation are delivered at the end of the project, including a live demonstration of the aTendy MVP, a summary of achieved objectives, and lessons learned during development.