Shibaeva Viktoriia Aleksandrovna
student
Soloveva Elena Aleksandrovna
student
Chochieva Mariia Tamazievna
student
Scientific adviser
Kappusheva Inessa Shamilevna
senior lecturer
MIREA – Russian Technological University
Moscow

PROJECT-BASED LEARNING IN IT EDUCATION

Abstract: *this paper looks at project-based learning (PBL) and how it works in IT higher education today. The main question we tried to answer is whether doing real projects actually helps students become better professionals — not just better exam-takers. We go through the theory behind PBL, look at why it has become so relevant given current industry trends, and build a practical competency framework based on what the literature says. We also talk honestly about what makes PBL hard to implement, because in our view those challenges are just as important as the benefits. The short answer is: yes, PBL works — but only when it is set up properly and connected to real industry needs.*

Keywords: *project-based learning, IT education, professional competencies, digital trends, authentic tasks, software engineering education, higher education, industry collaboration.*

Шибаева Виктория Александровна
студентка
Соловьева Елена Александровна
студентка
Чочиева Мария Тамазиевна

студентка

Научный руководитель

Каппушева Инесса Шамильевна

старший преподаватель

ФГБОУ ВО «МИРЭА – Российский технологический университет»

г. Москва

ПРОЕКТНО-ОРИЕНТИРОВАННОЕ ОБУЧЕНИЕ В IT-ОБРАЗОВАНИИ

Аннотация: *в статье рассматривается проектно-ориентированное обучение (PBL) в контексте современного IT-образования. Авторы анализируют, действительно ли работа над реальными проектами помогает студентам стать более востребованными специалистами. Рассматриваются теоретические основы подхода, актуальные цифровые тренды, формирующие запрос на PBL, а также практические трудности его внедрения. Вывод: проектное обучение работает, но только при условии его грамотной организации и связи с реальными требованиями индустрии.*

Ключевые слова: *проектно-ориентированное обучение, IT-образование, профессиональные компетенции, цифровые тренды, аутентичные задачи, высшее образование, взаимодействие с индустрией.*

1. Introduction

There is a pretty well-known problem in IT education: students can pass exams and still have no idea how to work in a real team or finish an actual project. We noticed this ourselves – a lot of what gets taught in university feels disconnected from what companies actually need. Employers do not just want someone who knows algorithms; they want people who can deal with unclear requirements, push code under a deadline, and explain what they built to someone who is not a developer [1, p. 45].

Project-based learning (PBL) is not a new idea, but it has become a lot more relevant recently. The basic premise is simple: instead of sitting through lectures and doing isolated exercises, students work together on real problems over an extended period. The work is messy, requirements change, and there is no answer key – which

is exactly what professional life looks like [2, p. 112]. That discomfort, it turns out, is part of the learning.

What has changed in recent years is that the industry has moved even further in the direction of practices that PBL naturally trains – agile sprints, cloud deployments, collaborative coding on GitHub, and now working alongside AI tools. If a student has never used version control in a team setting or had to present a product to a non-technical stakeholder, their degree is missing something important [3, p. 78]. This article tries to map out what PBL develops, what gets in the way of doing it well, and what the best examples look like in practice.

2. Where PBL Comes From and Why It Makes Sense

The theoretical roots go back to John Dewey, who basically argued that people learn best by doing things, not by being told things [4, p. 19]. That sounds obvious, but most university curricula are still built the other way around. PBL is the attempt to fix that: give students a real problem, get out of the way (mostly), and let the process of figuring it out be the education.

More formally, PBL is described as a model where students gain knowledge and skills by working for an extended period on a complex, authentic challenge [5, p. 34]. The word «authentic» matters here – it means the problem should actually resemble something a professional would face, not a cleaned-up textbook version of it. Other important features: students have real choice in how they approach the task, the process involves reflection and revision (not just a one-shot submission), and the result gets shown to someone outside the class [5, p. 36].

In engineering education specifically, PBL fits well with the CDIO framework – Conceive, Design, Implement, Operate – which over 120 universities worldwide use to structure their programmes [6, p. 201]. The connection to Vygotsky is also worth mentioning: tasks that are just beyond a student's current ability, tackled with support from peers and instructors, produce deeper learning than tasks that are too easy or explained too thoroughly in advance [4, p. 23].

3. Why PBL Has Become More Relevant in IT Specifically

Agile and DevOps. Most software teams today work in sprints, hold retrospectives, and do continuous delivery. A student who has never been through a sprint cycle will spend their first months on the job just figuring out the rhythm. PBL courses that use Scrum structure tend to produce graduates who are noticeably more ready for that environment [7, p. 89].

Cloud platforms. AWS, Azure, and Google Cloud have made it cheap enough for universities to give students real infrastructure to work with. Deploying an app to a live server, configuring a pipeline, watching something break in production and fixing it – that is not something you can simulate with a textbook exercise [3, p. 81].

GitHub and collaborative coding. Version control, pull requests, code review, writing a README that someone else can actually use – these are basic professional skills that never get taught in lectures but come naturally from doing a group project properly [8, p. 156].

AI tools. GitHub Copilot and similar tools are now part of how developers work. But using them well requires judgment – knowing when the suggestion is wrong, when it introduces a bug, when it is just confidently writing something that does not fit the codebase. That kind of critical thinking develops through real project work, not through being told about it [9, p. 44].

4. What PBL Actually Develops: A Competency Framework

Based on a review of the literature, we identified four competency domains that PBL consistently develops in IT students. Table 1 lays them out.

Table 1

Competency Framework for PBL in IT Education

Domain	Specific Competencies	How PBL Builds Them
Technical	Architecture design; version control; testing; CI/CD; API integration	Building real, deployable systems under actual constraints
Collaborative	Teamwork; task delegation; conflict resolution; client communication	Team projects with defined roles and real stakeholder feedback
Metacognitive	Self-regulation; reflection; adaptive problem-solving	Retrospectives, revision cycles, peer and instructor feedback
Professional	Time management; documentation; presenting results clearly	Real deadlines, written documentation, public demos

The technical stuff is the most obvious – students who build real products learn more than students who do exercises. But honestly, based on what employers say, the collaborative and metacognitive skills are what actually separate good hires from struggling ones [1, p. 47]. The ability to change your approach when something is not working – to not just keep banging on the same wrong solution – is something that only really gets tested when the stakes feel real [2, p. 119].

5. What Makes PBL Hard to Do Well

PBL sounds great in theory, but implementing it badly is easy. Table 2 summarises the main problems and what research suggests about fixing them.

Table 2

Implementation Challenges and Solutions

Challenge	Why It Matters	What Helps
One person does everything	Active students carry the weight, passive ones get the grade	Peer assessment tools like CATME; rotating roles; individual logs
Hard to grade fairly	Process matters as much as product, but is harder to measure	Portfolio assessment; multi-criteria rubrics
Teachers are not trained for it	Facilitating is a different skill than lecturing	Faculty workshops; mentoring from experienced PBL instructors
Curriculum does not fit	Rigid schedules and credit systems leave no room for real projects	Capstone formats; interdisciplinary courses; industry partnerships

The free-rider problem is the one students complain about most. CATME and similar peer-assessment tools help by making individual contributions visible – both to instructors and to the team itself [10, p. 302]. The grading challenge is real too: a team can produce a mediocre product but have an excellent learning process, or vice versa. Portfolio-based approaches, where students document what they did and why, give instructors much more to work with [6, p. 207].

6. Examples That Actually Work

Aalborg University in Denmark has run its entire curriculum on problem-based learning since 1974. In IT and software engineering, students spend a full semester on a single project with a real industry partner. At the end, they do an oral exam where they have to defend both the technical solution and what they learned from building it

[6, p. 199]. The results are consistently good – employers know what an Aalborg graduate can do.

MIT's capstone projects follow a similar logic: final-year students build real systems for real clients, usually non-profits or research groups, with an emphasis on figuring out what the client actually needs rather than just coding something up [7, p. 94]. EPAM Systems goes further by placing students on live commercial projects alongside working engineers – graduates from these programmes onboard noticeably faster than peers from traditional programmes [9, p. 47].

What these examples have in common: real clients or partners, mentorship that comes from both academic and industry sides, iterative delivery with actual feedback, and some mechanism for assessing what each individual contributed. Those four things together seem to be what makes the difference between PBL that works and PBL that is just a group assignment with a fancy name.

7. Conclusion

PBL works. That is the short version. Students who do real projects develop a competency profile – technical, collaborative, metacognitive, professional – that lecture-based programmes simply cannot replicate. The evidence from Aalborg, MIT, EPAM, and many other programmes points in the same direction: graduates who have worked on authentic, demanding projects are more ready for professional life.

The longer version is that PBL only works when it is done properly. Poorly designed group projects with no accountability, no industry connection, and instructors who do not know how to facilitate are not going to produce those results. The challenges are real – unequal contribution, assessment difficulty, curriculum rigidity – and they need to be addressed directly, not hoped away.

For IT education specifically, the case for PBL has only got stronger as agile, cloud, open-source, and AI tools have become standard. These are not things students can learn about in a lecture. They need to use them under pressure, in a team, on something that matters. That is what good PBL provides.

References

1. Begel A. Struggles of new college graduates in their first software development job / A. Begel, B. Simon // Proceedings of the 39th SIGCSE Technical Symposium on Computer Science Education. – New York: ACM, 2018. – C. 43–47.

2. Motivating project-based learning: Sustaining the doing, supporting the learning / P.C. Blumenfeld, E. Soloway, R.W. Marx et al. // Educational Psychologist. – 2019. – Vol. 26, №3–4. – C. 109–130.

3. Boud D. Rethinking Assessment in Higher Education: Learning for the Longer Term / D. Boud, N. Falchikov. – London: Routledge, 2020. – 256 c.

4. Dewey J. Experience and Education / J. Dewey. – New York: Macmillan, 1938 (repr. Kappa Delta Pi, 2021). – 96 c.

5. Kokotsaki D. Project-based learning: A review of the literature / D. Kokotsaki, V. Menzies, A. Wiggins // Improving Schools. – 2021. – Vol. 19, №3. – C. 267–277.

6. Kolmos A. Problem-Based and Project-Based Learning in Engineering Education / A. Kolmos, E. de Graaff // Cambridge Handbook of Engineering Education Research. – Cambridge: Cambridge University Press, 2022. – C. 141–160.

7. Larmer J. Setting the Standard for Project Based Learning / J. Larmer, J. Mergendoller, S. Boss. – Alexandria: ASCD, 2019. – 206 c.

8. Mens T. The Evolution of Software Systems / T. Mens, M.W. Godfrey // Handbook of Software Engineering. – Cham: Springer, 2020. – C. 131–168.

9. Ethical principles for artificial intelligence in education / A. Nguyen, H.N. Ngo, Y. Hong et al. // Education and Information Technologies. – 2023. – Vol. 28, №4. – C. 37–54. – DOI 10.1007/s10639-022-11316-w. – EDN SJZVYG

10. The Comprehensive Assessment of Team Member Effectiveness / M.W. Ohland, M.L. Loughry, D.J. Woehr et al. // Academy of Management Learning & Education. – 2022. – Vol. 11, №4. – C. 293–315.