Project-Oriented Teaching Approach of Decentralised Applications for Undergraduate Students

Sandi Gec
Faculty of Computer and Information Science, University of Ljubljana, Ljubljana, Slovenia
sandi.gec@fri.uni-lj.si

Petar Kochovski
Faculty of Computer and Information Science, University of Ljubljana, Ljubljana, Slovenia
petar.kochovski@fri.uni-lj.si

Vlado Stankovski
Faculty of Computer and Information Science, University of Ljubljana, Ljubljana, Slovenia
vlado.stankovski@fri.uni-lj.si

Dejan Lavbič
Faculty of Computer and Information Science, University of Ljubljana, Ljubljana, Slovenia
dejan.lavbic@fri.uni-lj.si

ABSTRACT

The emergence of new technologies and software engineering methodologies in the decentralised Web, referred to as Web 3.0, often requires using new approaches to supplement conventional development concepts such as object-oriented programming and design patterns. Therefore, it is essential to understand the blockchain fundamentals and modular development of distributed systems built from multiple components. This paper proposes a project-oriented teaching approach of smart contracts-based distributed applications (dApps) integrated into existing (Web 2.0) Web applications. First, we present the main overview of the Web programming course for a bachelor study program, prior knowledge requirements, and the characteristics of the lectures. Second, we define the dApp functional requirements for student projects that intend to consolidate the dApp theoretical concepts with a practical approach to complement the Web 3.0 concepts. The study of our teaching approach is based on an experiment where we thoroughly analyse performance on the course level, projects and individual students. The results indicate a high correlation between students' performance in Web 2.0 and Web 3.0 development, and a higher degree of iterative process is also associated with better-performing students. Moreover, the student project outcomes also provide a basis for possible future course improvements.

CCS CONCEPTS

• **Information systems**; • **Digital cash**; • **Browsers**; • **Networks**; • **Cloud computing**; • **Web services**; • **Applied computing**; • **Education**; • **Social and professional topics**; • **Software engineering education**;

KEYWORDS

teaching, decentralised application, smart contract

ICETC 2023, September 26–28, 2023, Barcelona, Spain

ACM ISBN 979-8-4007-0911-1/23/09…$15.00
https://doi.org/10.1145/3629296.3629328

ACM Reference Format:
Sandi Gec, Petar Kochovski, Vlado Stankovski, and Dejan Lavbič. 2023. Project-Oriented Teaching Approach of Decentralised Applications for Undergraduate Students. In *The 15th International Conference on Education Technology and Computers (ICETC 2023), September 26–28, 2023, Barcelona, Spain.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3629296.3629328

1 INTRODUCTION

The increasing complexity of software solutions and the rise of Cloud-to-Edge architectures have led to the adoption of novel development approaches such as Distributed Ledger Technology (DLT), particularly Ethereum-based smart contracts. These contracts can be deployed on other ledgers for improved performance and enable the creation of dApps for various use cases, including supply chain management [1].

However, training dApp developers is a challenge due to the vast range of concepts and technologies involved [2]. Traditional online courses often lack a comprehensive overview of the development process, from defining requirements to deploying and integrating the solution.

To address this, we propose a project-based teaching approach for undergraduate students to consolidate their knowledge of dApp development. The course consists of practical lectures and lab work, with students working in teams to develop their custom dApp use case. The course begins with Web 2.0 development and gradually shifts towards developing dApps (Web 3.0) by upgrading their existing solution. The students are provided with online and offline support throughout the course, and their performance is analysed through quantitative and qualitative measures.

The remainder of this paper is structured as follows. Section 2 places our work in the context of other related works. Section 3 describes the development process of dApps in the context of the *Web development* course. Section 4 presents the experimental study with quantitative and qualitative results, and section 5 concludes.

2 RELATED WORK

From the release of the Bitcoin [3] in 2009, as the first public DLT, the interest in dApps rapidly increased. The worldwide usage sustained the pillar blockchain benefits such as distribution, data immutability, transparency, consensus, security, and increased capacity. After that, many DLT solutions were released mainly to cover additional use cases through the adopted Quality of Service requirements

(e.g. decreased block time, different consensus mechanism, adjusted distribution level) or by introducing novel technological concepts. One of the most resonant was the introduction to smart contracts in the release of Ethereum (https://ethereum.org/) DLT solution in 2015.

Since dApps development differs from conventional applications, it is essential to address the challenges and adapt the pedagogical approach to effectively present the concepts in a life-cycle-based process as presented by Z. Zheng et al. [4]. Blockchain technology became prominent 6 years ago in education, and the integration aspect shows that only a handful of papers provide a working solution or are even integrated with core learning educational technologies [5]. Therefore, in our work, we strive to address this process in detail to teach students how to incorporate smart contracts in the native HTML and/or Angular Web applications to fully exploit the dApps capabilities (e.g. asynchronous execution). In the study of ongoing educational approaches on DLT, it is possible to categorise the components throughout components among individual layers used in the dApp development process as a framework of blockchain teaching platform and other dedicated modules [6]. Similarly, our project-oriented approach describes the components used in the development process tightly in conjunction with the Model–view–controller (MVC) architectural pattern. A similar systematic approach was successfully applied and proposed by Y. Shang et al. [7] higher education blockchain courses where the pillar topics are divided into layers, from a low-level layer overview to a high-level GUI presentation layer. A dApp-oriented research study on smart contracts in education was proposed by A. Alkhajeh [8], where the major identified disadvantage is often the lack of provision of the scope for practical skill development apart from theoretical knowledge. To overpass this problem, our course strives to encourage practical development that is often presented as a supplement to the theory during the course. A reasonable approach suitable for a social sciences student was proposed in the form of a blockchain-based visual programming framework called SmartBuilder [9] that allows building contracts using extended Google Blocky libraries. Nevertheless, the approach may introduce technological limitations for computer science undergraduate students in the dApp development process of advanced features.

Another very popular pedagogical perspective of blockchain and related technologies is online courses that are more time flexible but often need immediate support and/or feedback [10]. To improve the quality of online teaching content, A. Garg et al. [11] proposed a blockchain-based content review system for online educational courses that ensures the integrity of content reviews by Subject Matter Experts (SME). In practice, many potential blockchain software developers make use of online courses. For example, very popular online courses are Coursera Smart Contracts (https://www.coursera.org/learn/smarter-contracts), Ethereum and Solidity(https://tinyurl.com/25h5733k) providing 23.5 hours of video content, React framework oriented course (https://tinyurl.com/mrx9xry6) and many others. In comparison to the mentioned online blockchain and smart contract courses, ours consist of 9 hours of lectures, 6 hours of lab work, and additional online or offline support, covering the blockchain basics, the complete development process of Ethereum-based smart contracts, and comprehensive integration into an existing three-tier Web application. The overall course lectures covering Web 2.0 and Web 3.0 content are spread over 14 weeks total 42 hours, whereas the presentation of the practical content, without defence and consultation weeks, in the lab work is spread over 10 weeks total of 20 hours.

In this work, we analyse the traditional Web and dApp development students' performance using the project-oriented teaching approach at the *Web programming* course for the winter semester 2022/2023. The course is prepared based on the good practices and scientific findings presented in the following sections.

3 DEVELOPMENT OF DECENTRALISED APPLICATIONS IN WEB PROGRAMMING COURSE

This section presents a comprehensive overview of the traditional and dApps development processes in the Web programming course. First, we overview the DLT background and discuss the selection of the technologies in the context of the teaching approach. Second, we discuss the required prior knowledge to follow the Web programming course seamlessly. Third, we present the course methodology from the course overview to the smart contract development process with presented use cases during the course. Finally, we designate the dApp evaluation criteria assessed at the final lab work defence.

3.1 Background and motivation

The primary purpose of the initially introduced public distributed ledgers (e.g. Bitcoin) is the support of base transactions as executing transactions from one wallet to another. A more sophisticated architecture has to be proposed for more complex features, like smart contracts. Therefore, the Ethereum foundation proposed a distributed state machine which enables data structures to be changed from block to block according to a pre-defined set of rules defined in a script-like program called a smart contract. The Ethereum's ledger component Ethereum Virtual Machine (EVM) defines the specific rules of changing states.

The open-source availability of Ethereum's EVM allows integration of smart contracts into other ledgers developed to enable new use cases and thus allow the migration of the smart contracts on other chains. Moreover, the EVM-based smart contracts programming language Solidity encapsulates major object-oriented concepts. It is similar to the programming languages JavaScript and TypeScript, both presented in the traditional Web development of the project.

Instructors proposed the potential dApp use case ideas to the students based on prior development and research experience, mainly from the projects in the Horizon2020 EU's research and innovation programme such as the past project DECENTER (https://www.decenter-project.eu/) and ongoing projects ONTOCHAIN (https://ontochain.ngi.eu/) and TRUSTCHAIN (https://trustchain.ngi.eu/) .

3.2 Prior knowledge requirements

The Web programming course is suitable for undergraduate computer science students or other technical fields of study from the

University of Ljubljana, Slovenia. It is a free elective for study programs such as the Faculty of Computer and Information Science and master students of Computer Science and Mathematics. The course is mandatory for students of the Multimedia study program at the Faculty of Electrical Engineering. Common to all of the study programs is the prior knowledge acquired through the compulsory courses from whom the most relevant is (Java) programming. To be more concrete, the students need to understand data types, arithmetic logic operations and object-oriented concepts. Students upgrade the programming concepts with new software engineering approaches during the Web programming course.

3.3 Course methodology

The *Web programming* course aims to provide students with an overview of heterogeneous technologies and provide the capacity for autonomous learning of new technologies in Web development (Web 2.0). A particular emphasis is given to full-stack development of distributed and decentralised Web applications (Web 3.0).

3.3.1 High-level course overview. The key characteristics of the *Web programming* course are described from a high-level perspective. During the course, the students will gain the pillar Web development knowledge through the overview of technologies used within the Web, Web servers, Web browsers, Web applications and blockchain.

At the beginning of the course, the students are acquainted with the basics of Web page development, supplemented with a comprehensive overview of client-side development using native JavaScript programming language, server-side development, and implementation of REST API access to the database. Since the course strives to present the Web development process of production-ready applications, particular attention is given to the upgrade from a security perspective, improvements with Progressive Web Application (PWA) approaches, and development of dApps on the Ethereum-based blockchain, including smart contracts.

The course consists of lectures, laboratory exercises and a Web project that students present throughout the semester. Each project comprises 4 to 5 students who select the problem domain at the beginning of the semester and define the functionalities to be implemented.

The content of the course is divided into the following 6 topics:

- *Introduction to Web programming*; Examination of basic Web concepts such as MEAN stack architecture, HyperText Markup Language (HTML) as the standard markup language for documents designed to be displayed in a Web browser and Cascading Style Sheets (CSS), a style sheet language used for depicting the presentation of a document written in HTML. The general goal of the topic is to understand the pillar front-end building blocks.
- *Programming the Web on the client and server side*; JavaScript, as one of the most used programming languages of the Web on the client and server side, is utilised. The MVC concepts are introduced where the practical experience, apart from development, extends to hosting in the cloud.
- *Back-end and data access*; Development of the REST API server-side using Node.js framework and allows interaction with the document-oriented database MongoDB. Moreover, students are encouraged to define and develop the API documentation via the framework Swagger to exhaustively get acquainted with good practices.
- *User interface*; Due to the latest practice guidelines, particular attention is dedicated to the front-end Single Page Application (SPA) development approach as a leading technology Angular framework is used to upgrade the existing static HTML pages into performant dynamic ones. In addition, the authentication using JSON Web Tokens (JWT) is studied, on both the front-end and back-end, to increase the overall security of the Web application significantly.
- *Decentralised applications and blockchain*; The fundamental concepts of the blockchain are presented (Web 3.0) to complement traditional Web development (Web 2.0) applications. The topic aims to offer a complete step-by-step Ethereum-based smart contract-oriented development methodology that exploits the potential of the blockchain features on concrete use cases – lecture projects, lab work projects and student projects. A more detailed development course plan is described in the following subsection.
- *Additional security overview*; More modern security approaches such as Progressive Web Apps (PWA) are examined to maximise the user experience on mobile devices and, at the same time, address particular security concerns (e.g. caching resources, offline behaviour of the Web page etc.).

The course workflow, depicted in Figure 1, follows the presented 6 topics and is represented from an MVC architectural perspective. All development material (lectures, lab work and student projects) is stored, hosted and managed under the Web-based version control management system GitHub available in private repositories based on different access rights (e.g. GitHub repository "student group 1" is available only to the students within the group 1). Moreover, students are encouraged to provide the source code and document the project requirements, descriptions and other material using the markup language Markdown within their own GitHub repositories. In the context of the project-oriented teaching approach, the students work on two pillar development methodologies as follows:

- *Traditional Web development (Web 2.0)* that covers topics 1 to 4 where the students apply the concepts in a practical development approach most of the time during the semester.
- *dApp development (Web 3.0)* covers topic 5, where the dApp development methodology and the project-oriented dApp requirements are presented in detail in the rest of the section.

3.3.2 Smart contract development cycle. The development cycle of the smart contracts is splitted into individual phases, from the initial dApp project to the deployment on the testing environment as depicted in Figure 2. The initial support for Ethereum-based smart contracts in Solidity programming language is generated by creating Truffle project (https://trufflesuite.com/). The rough initial correctness of the smart contract is analysed with the Solidity compiler and deployed on a single local node Ethereum testing environment Ganache as a component of the Truffle suite. Before the development of an actual smart contract, it is highly recommended that the developer defines the use case's functional requirements, main functionalities, workflow, user groups and other features. Then,

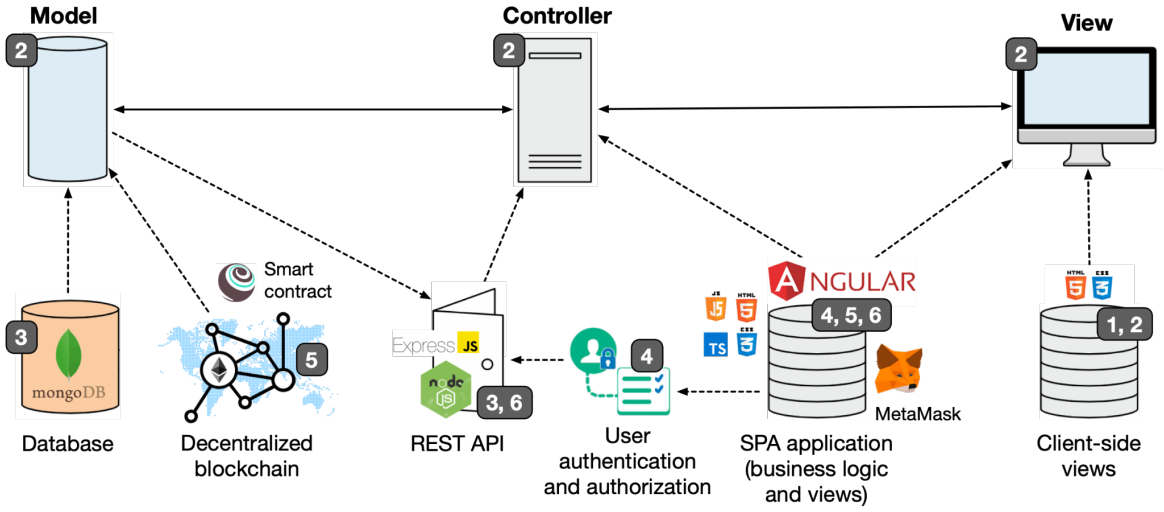

Figure 1: Outline of the Web programming course through an MVC architectural perspective where the numbers represent the 6 pillar course topics.

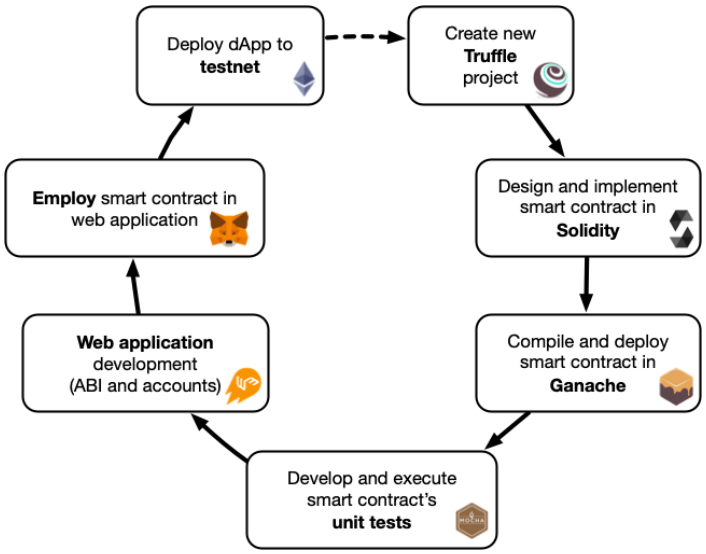

Figure 2: We propose a full-stack-oriented programming plan that tends to iteratively cover all fundamental aspects of dApp development: design, implementation, testing, integration and deployment.

the result of the analysis is combined using an object-oriented approach (e.g. generalisation, structures definition, state definition with enums and others) and Ethereum's smart contract development methodology (e.g. events, transactions) combined with the smart contract building blocks. The workflow of the developed smart contract should not be tested only manually, but adequate unit tests should be developed to ensure the general use case workflow and boundary conditions. After a thorough testing phase, the integration in the Angular-based SPA begins with the definitions of (wrapped) smart contract functions in the Angular Service component (Web3 service). Since ordinary Web browsers do not enable blockchain integration by default, the end-user-based interaction is performed using the most widely used Web plugin MetaMask. Smart contract functions defined in the Web3 service are integrated into preferred Angular component or multiple ones.

3.3.3 Smart contract use cases. Use cases play an essential role in the dApps teaching process. The fundamental dApp concepts focused on smart contracts are described during the course. Since the course is practically oriented, the building blocks and the most used development patterns and/or features are illustrated in practical smart contract use cases. The characteristics of the smart contracts, features, description and the level of integration are depicted in Table 1.

Table 1: Smart contract examples presented during the course.

Smart contract(s)	Features	Front-end integration	Short description
AuthorizedToken	base presentation of all building blocks, mocked transfer	×	In-contract token minting and ownership illustration.
SimpleCoin, Ownable	upgraded AuthorizedToken, abstraction (contract), coin transfer implemented	×	Transfer management of newly defined coins within the smart contract.
SimpleCrowdsale, Pausable, Destructible, ReleasableSimpleCoin	owner role able to pause, destroy and release coins	×	Crowd sale use case with safety features pause, release and destroy.
Inheritance	multi-level inheritance of contracts	×	Proof-of-concept inheritance of 3 contracts.
AbstractContract	abstract contract features	×	Example of an abstract contract.
SampleContract, SampleInterface	interface example	×	An example of a contract extending an interface contract.
Calculator, SafeMath[a]	external library integration	×	An example of an external library inheritance.
SimpleVoting	all building blocks, state-based functionalities	√	A comprehensive bidding use case integrated into the main lecture project.
Aucton, AuctonInterface	all building blocks, state-based functionalities, abstr. (interface)	√	A comprehensive auction use case integrated into the lab work project.

[a] https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/utils/math/SafeMath.sol

3.4 dApp requirements and evaluation criteria

The course aims to master the main dApp fundamentals within the development in a practical object-oriented approach. Since the project domain selection is entirely free, we define the functional requirements as generic enough not to limit the imagination of the students and, at the same time, thoroughly test the most critical concepts.

3.4.1 Functional requirements. The dApp development part of the course aims to upgrade traditional Web development student projects following the terms defined as functional requirements. One or more existing Web pages within the project should be upgraded to allow comprehensive dApp interaction, namely:

- The chosen technology has to support native Ethereum ledger, including support for smart contracts by using either the Ether (ETH) cryptocurrency or generated own non-fungible ERC-20 crypto tokens. Fungible tokens are not allowed to be used.
- The core implementation of the dApp must be implemented in the front-end, either native HTML or preferably Angular, using the Web browser plugin MetaMask that enables the blockchain interaction within the library Web3.js (https://Web3js.readthedocs.io/).
- Upgrade the existing (MongoDB) database with dApps attributes (e.g. account addresses, account roles, transactions, results of smart contract interactions) according to your problem domain so that it is possible to demonstrate the selected use case on a practical example.
- The dApp use case should be enabled for at least one type of logged-in user, where the used MetaMask wallet account must match the user's wallet address.
- When registering a user within the traditional Web application, save the user's wallet address or the public key.
- De-duplicate and reuse code as much as possible while using Solidity building blocks and use at least 1 external library.
- Use OpenZeppelin or other available open-source libraries.
- Development of at least 1 smart contract, where at least 2 dApps functionalities will be included, e.g.:
- Implementation of a function that allows content updates of a smart contract with at least 1 input data (e.g. data record, status update of existing attributes, rights update).
- Implementation of a function that allows either a native cryptocurrency or a token transaction by updating the content of the smart contract (e.g. sending funds, exchanging funds, voting).
- Integration of events into at least one function of the smart contract and implementation of the event listener within the *Web3.js* library in any component of the traditional Web application.
- For extra points: Implementation of an off-chain data read function using ChainLink DLT (https://chain.link).
- The core scenario supported by the smart contract should be fully covered by test cases implemented in the Truffle framework, where there is at least 1 positive and 1 negative test for each function.

The summary of the dApp functional requirements is depicted in Figure 3. The interaction between the end-user and the dApp should be focused on using MetaMask wallet where it is possible to confirm access to an individual account and further the communication with the local ledger through *Web3.js* library.

3.4.2 Evaluation criteria. All projects are evaluated step-by-step through the following evaluation criteria.

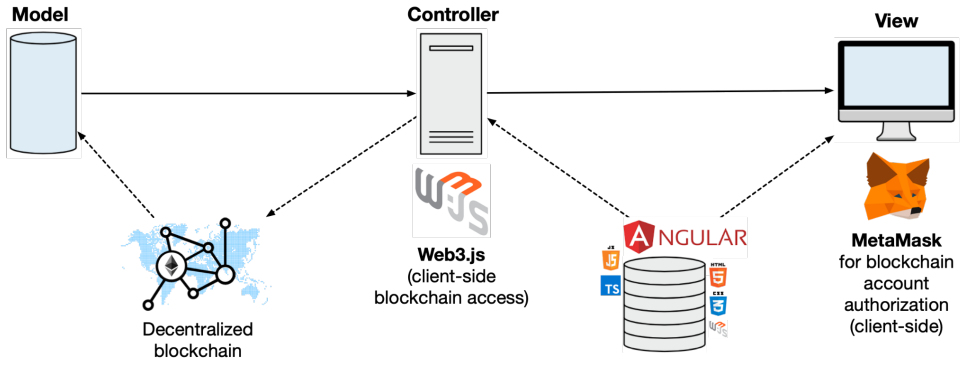

Figure 3: dApp functional requirements for the student projects.

- A smart contract in Solidity programming language is developed and is consistent with the description in the GitHub project repository file *README.md*.
- A smart contract uses abstractions (e.g. generalisation of contracts, interfaces) and reusable building blocks.
- A smart contract reasonably uses at least 1 external library.
- Within the dApp on the client side, there is communication with MetaMask, which enables the authorisation of transactions on the selected (local) blockchain. Individual wallets should be assigned to each user in the registration phase.
- The smart contract is developed using the Truffle framework, where at least 2 tests (1 positive and 1 negative) are present for each contract function.
- Communication between the dApp and the blockchain works and is supported using *Web3.js* on the client side. All smart contract functionalities are available.

4 EXPERIMENTAL STUDY

4.1 Baseline metrics

There were 108 students enrolled in the Web Programming Course for the winter semester of 2022/2023. 71.3 % of students have Computer Science background, 25 % Electrical Engineering background and 4.7 %other (Mathematics or Interdisciplinary). 102 students (94.4 %) actively participated in the course, attending lessons, doing lab work and working on assignments.

At the beginning of the course, students completed the introductory questionnaire for instructors to understand their foreknowledge level better. One of the questions for students was, what is your experience developing online information solutions? 61.4 % students responded that they are not professionally active in software development and want to dedicate themselves fully to their studies. 28.4 % students replied that, in addition to their studies, they occasionally participate in smaller projects, while 10.4 % students have been developing information solutions for several years. Another question was how well you know technologies related to Web development. As depicted in Fig. 4 more than 75 % of students were at least familiar with essential support technologies for Web development (e.g. object-oriented programming languages and Git version control). Blockchain and dApps knowledge were worse, with only around 20 % of students slightly familiar with Web 3.0 development.

4.2 Qualitative dApp projects analysis

At the beginning of the course, the students were divided into 21 groups of 4 to 5 students per group. The project assessment was divided into two pillar topics: traditional Web app development and dApp development. All the groups actively engaged in traditional Web app development and were on the defence. On the other hand, only 57 % of project groups were present at the defence of the dApp development assignment, of which 66 % fully integrated dApps into an existing Web application. The main reason for the decreased interest is most likely because the dApp development assignment counts only 6 % of the final score.

From the proposed dApp solutions, 25 % of the projects provided a slightly upgraded auction use case from the lab work, 25 % of the projects developed and integrated a slightly upgraded simple voting dApp from the lectures and 16 % proposed simple upgrades on other existing smart contracts presented on the lectures. A group project proposed an exciting approach of 2 smart contracts interacting with either SPA front-end or REST API server. The 3 most original dApp use cases are presented below.

An interesting dApp integrated a decentralised prize draw into social media like a Web application intended to connect people with common interests. The dApp relies on ERC-20 based dedicated token that supports the minting of tokens, lottery playing and its custom stochastic algorithm for selecting the winner.

Another interesting dApp complements the existing borrow or rent of goods Web app use case. The entire process is transparent and allows a simple monetisation use case with the native coins.

The most advanced dApp consists of dedicated 3 smart contracts: (i) generate own ERC-20 based token, (ii) exchange native coins for tokens on a predetermined ration and (iii) payment contract allowing reservations as a partial payment or full payments. All these contracts were comprehensively integrated into SPA front-end and REST API back-end. In the front-end, only privileged users (administrators) can generate the tokens, determine the exchange rate and check the status of the payments and token distribution. Other end-users are able to exchange the native (Ether) coins into

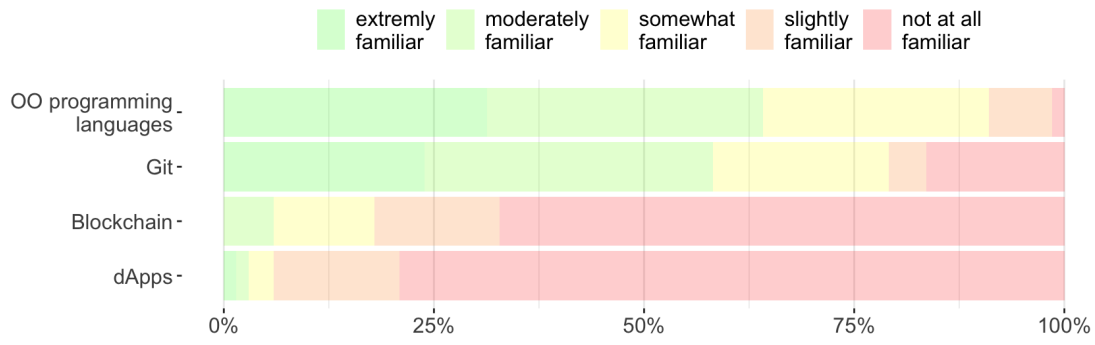

Figure 4: Student's foreknowledge of Web development-related technologies.

tokens to perform token payment on the car rental use case within the Web application.

4.3 Students' performance validation

When analysing students' performance in Web 2.0 and Web 3.0 development, we clustered the results into four groups, as depicted in Fig. 5. The cutoff in group formation for the evaluation score of students' assignments was 50 %.

43.1 % of students fit in outperformers in the Web 2.0 and Web 3.0 development group (outlined in green in the top right part of Fig. 5). The next group by size, including 38.3 % of students, were outperformers in Web 2.0 development and underperformers in Web 3.0 development (outlined in blue in the bottom right part of Fig. 5). A large part of this cluster consists of students who completed the development of the Web 2.0 assignment but did not submit the Web 3.0 assignment. The following 18.6 % of students were underperformers in Web 2.0 and Web 3.0 development (outlined in red in the bottom left part of figure 5). A large subset of this group also did not submit the Web 3.0 assignment after performing poorly on the Web 2.0 assignment. Interestingly, no students were found in Web 2.0 underperformers and Web 3.0 outperformers, indicating that Web 2.0 development knowledge is essential for Web 3.0 development.

When analysing the correlation between evaluation score in traditional Web app development (Web 2.0) and dApp development (Web 3.0) using the Pearson coefficient, the results $r = 0.56$, $p = 10^{-9}$ indicate a strong and statistically significant correlation. This finding also supports the linear regression model between Web 3.0 and Web 2.0 development performance with $k = 0.71$ and $n = -15.96$, as depicted in Fig. 5

When further analysing individual groups from Fig. 5, we can observe a higher degree of the iterative process with better-performing students in Fig. 6. For the development of dApp, students had a total available time of 3 weeks. The best group of students outperforming in Web 2.0 and Web 3.0 (the green group) utilised 90.9 % of available time. The blue group with Web 2.0 outperformers and Web 3.0 underperformers utilised 45.5 % of available time, while Web 2.0 and Web 3.0 underperformers (the red group) utilised only 18.2 % of available time. The worst-performing (red) group did most of the work just before the assignment submission. In contrast, the best performing group started working on an assignment when receiving the instructions and divided their time much more evenly.

5 CONCLUSION

The project-oriented teaching approach for decentralised applications is a highly effective method for undergraduate students to gain hands-on experience in developing and deploying these cutting-edge technologies. This approach involves an iterative process where students build and refine their skills through multiple projects, which helps to deepen their understanding and develop practical experience. The results emphasise a high correlation between success in traditional web applications and dApps. This approach is valuable because it considers that students should have a solid understanding of conventional web application approaches before delving into the decentralised world. The main limitation of our approach is the limited test set of students that will increase during the ongoing years. Moreover, the decentralised concepts may be further extended with mandatory novel requirements such as smart oracles and cross-chain approaches.

The project-oriented approach allows students to apply the theoretical knowledge they have gained in class to real-world projects, which helps to bridge the gap between theory and practice. Additionally, this approach emphasises the importance of teamwork and collaboration, critical skills in decentralised applications. Students can learn how to communicate effectively, manage conflicts, and lead projects by working in teams.

The proposed approach is a comprehensive method that prepares undergraduate students for a successful career in decentralised applications. Thus, the contribution of the paper can be summarised as a novel teaching methodology encouraging teamwork for decentralised applications that are a pillar component of multi-component heterogeneous systems. This approach covers the technical aspects of decentralised technologies and focuses on developing soft critical skills essential for the industry's success. By incorporating traditional web application approaches and an iterative approach, students receive a well-rounded education that equips them with the knowledge and skills necessary for a successful career in this exciting and rapidly-growing field. Based on

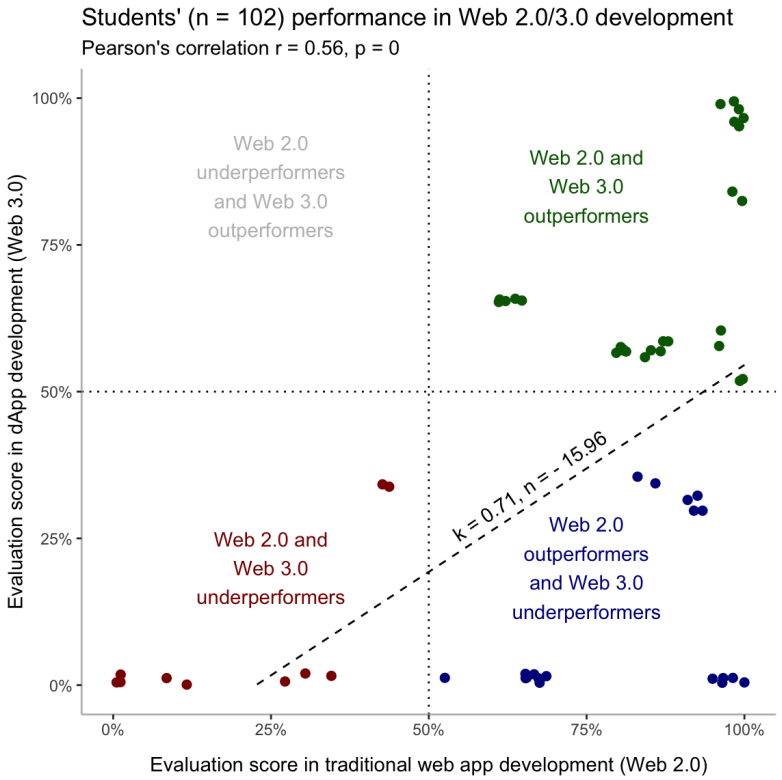

Figure 5: Best students in traditional Web development also outperform in developing decentralised applications.

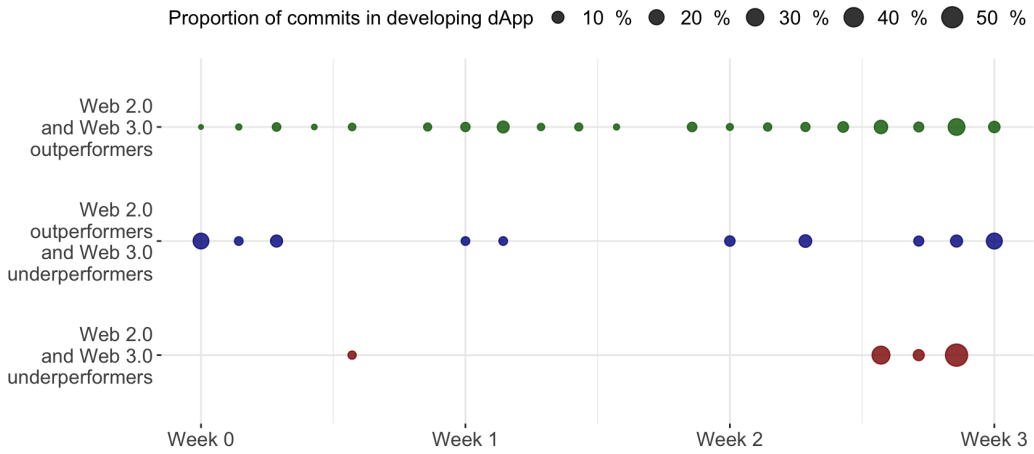

Figure 6: Utilisation of available time for dApp development.

an analysis of various development teams in our study, the best-performing groups tackle the problem in a more iterative approach and are also more successful.

While the project-oriented teaching approach for decentralised applications is an effective method for undergraduate students, there is always room for improvement. Some suggestions for future improvements include:

- Incorporating industry partnerships: Collaborating with companies or organisations actively working in decentralised applications can provide students with access to real-world projects and opportunities to network with industry professionals.
- Integrating new technologies: As decentralised technologies continue to evolve, it is essential to incorporate the latest

developments into the curriculum to ensure that students are exposed to the most up-to-date information.
- Encouraging critical thinking: Challenging students to evaluate the benefits and limitations of decentralised technologies critically can help them develop a nuanced understanding of these complex systems.

In summary, by incorporating these suggestions, the project-oriented teaching approach for decentralised applications can be further improved and continue to provide students with an engaging and practical learning experience.

ACKNOWLEDGMENTS

The research and development reported in this paper received funding from the European Union's ERASMUS+ programme under grant agreement no. 621751-EPP-1-2020-1-BE-EPPKA2-SSA-B (**ESSA: The European Software Skills Alliance**) and from the Research Agency of the Republic of Slovenia under the research programme P2-0426 **Digital Transformation for Smart Public Governance 1/1/22-12/31/27**.

REFERENCES

[1] Dhaiouir, S.: The impact and potential of using blockchain enabled smart contracts in the supply chain application area. In: Cherfi, S., Perini, A., Nurcan, S. (eds.) Research Challenges in Information Science. pp. 680–687. Springer International Publishing, Cham (2021)

[2] Ohei, K.N., Brink, R.: Web 3.0 and web 2.0 technologies in higher educational institute: Methodological concept towards a framework development for adoption. International journal for Infonomics (IJI) 12(1), 1841–1853 (2019)

[3] Nakamoto, S.: Bitcoin whitepaper. URL: https://bitcoin. org/bitcoin. pdf-(: 17.07.2019) (2008)

[4] Zheng, Z., Xie, S., Dai, H.N., Chen, W., Chen, X., Weng, J., Imran, M.: An overview on smart contracts: Challenges, advances and platforms. Future Generation Computer Systems 105, 475–491 (2020).

[5] Ocheja, P., Agbo, F.J., Oyelere, S.S., Flanagan, B., Ogata, H.: Blockchain in education: A systematic review and practical case studies. IEEE Access 10, 99525–99540 (2022). https://doi.org/10.1109/ACCESS.2022.3206791

[6] Li, L., Wu, X.: Research on school teaching platform based on blockchain technology. In: 2019 14th International Conference on Computer Science Education (ICCSE). pp. 38–43 (2019). https://doi.org/10.1109/ICCSE.2019.8845353

[7] Ying Shang, Zexiang Li, Zheng Li, and Yongqiang Jiao. 2023. Blockchain Technology and Its Application in Higher Education. In Proceedings of the 14th International Conference on Education Technology and Computers (ICETC '22). Association for Computing Machinery, New York, NY, USA, 108–113. https://doi.org/10.1145/3572549.3572567

[8] Alkhajeh, A.: Blockchain and smart contracts: The need for better education. Rochester Institute of Technology (2020)

[9] Merlec, M.M., Lee, Y.K., In, H.P.: Smartbuilder: A block-based visual programming framework for smart contract development. In: 2021 IEEE International Conference on Blockchain (Blockchain). pp. 90–94 (2021). https://doi.org/10.1109/Blockchain53845.2021.00023

[10] Zhao, M., Liu, W., Saif, A.N.M., Wang, B., Rupa, R.A., Islam, K.M.A., Rahman, S.M.M., Hafiz, N., Mostafa, R., Rahman, M.A.: Blockchain in online learning: A systematic review and bibliographic visualization. Sustainability 15(2) (2023).

[11] Garg, A., Kumar, P., Madhukar, M., Loyola-Gonz´alez, O., Kumar, M.: Blockchain-based online education content ranking. Education and information technologies pp. 1–23 (2022)