Co-funded by the European Union

Project Nr. 2022-1-FR01-KA220-HED-000086863

Erasmus+ Project

lightcode

Strengthening the Digital Transformation of Higher Education Through Low-Code

INTRODUCTION TO LOW-CODE

University of Macedonia

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Erasmus+ Project
lightcode

Co-funded by
the European Union

PROJECT'S COORDINATOR

Paris Dauphine University
France

dauphine.psl.eu

PROJECT'S PARTNERS

University of Macedonia
Greece

www.uom.gr

University of Nis
Serbia

www.ni.ac.rs

Karmic Software Research
France

www.karmicsoft.com

REACH Innovation
Austria

reach-innovation.eu

University of Zagreb
Croatia

www.unizg.hr

Symplexis
Greece

symplexis.eu

TABLE OF CONTENTS

Author: Giota Digkoglou (University of Macedonia)

OVERVIEW

Low-code development is a modern approach to software development which uses visual tools, reusable components, and automation to reduce manual coding and accelerate delivery. Additionally, low-code development approach allows a broad range of users, including non-professional developers from business backgrounds, to participate in developing software. This helps address the shortage of skilled developers, improves collaboration between IT and the business, and increases responsiveness to business and market needs.

It is becoming increasingly challenging for companies to source suitable IT talent. Low-code environments have the potential to reduce development time by over 50% compared to traditional coding languages. As low-code dramatically simplifies the software development process, companies of all sizes that adopt this approach have the opportunity to boost developer productivity and speed.

In view of the continuing evolution of technology, it is becoming increasingly clear that in order to maintain competitiveness on a number of fronts, whether in terms of the skills of employees or the business itself, it is not enough simply to innovate. There is also a need to demonstrate adaptability. The low-code approach offers a potential solution in this regard, since it provides a forward-looking framework within which emerging technologies can be integrated seamlessly, thereby ensuring that the company remains at the forefront of industry trends.

Co-funded by the European Union

Learning Outcomes

- Understand the definition and historical development of low-code development (LCD), especially their evolution and increasing relevance in both commercial and educational settings.
- Identify low-code platforms for designing and implementing simple projects.
- Appreciate the significance of low-code in fostering technological democratisation.
- Understand the role of low-code in reducing barriers to software development, making it accessible for non-technical users, and its contribution to increasing business agility.
- Explore the impact of low-code development in education through enhancing digital literacy, promoting creativity, and facilitating interdisciplinary collaboration.
- Identify and discuss the challenges and opportunities presented by low-code development, including its potential to significantly alter how organisations and industries operate digitally.
- Gain hands-on experience through interactive exercises and quizzes, enhancing understanding of low-code's functionality and its role in simplifying the development process.

Prerequisites

No prior coding knowledge is required to join this course. A general level of digital fluency (such as being comfortable with everyday computer use, apps, and online tools) will be helpful.

Co-funded by the European Union

Some additional background knowledge may be useful, but it is not required:

- Basic awareness of how software is developed (planning, design, testing, deployment).
- A general idea of programming concepts such as variables or loops.
- Familiarity with simple user interfaces (e.g., how menus, buttons, or navigation typically work).
- Awareness of how information is stored in databases or spreadsheets.
- An understanding of how businesses often organise and improve processes.
- General problem-solving and critical-thinking skills.
- Any experience with digital tools or platforms (even at a basic level).

These are helpful starting points, not entry requirements. The course is designed to support all learners, regardless of technical background.

Multiple Choice Quiz

Question 1: What is the typical sequence of steps in a traditional software development process?

a) Design, Testing, Implementation, Deployment
b) Planning, Design, Implementation, Testing, Deployment
c) Testing, Planning, Deployment, Design
d) Implementation, Design, Planning, Testing

Question 2: Which programming structure is used to repeat a specific block of code a known number of times?

a) If-Else statement
b) For loop
c) Function

Co-funded by the European Union

d) Variable

Question 3: What is the main purpose of user interface (UI) design in software development?

a) To manage database interactions
b) To optimise backend processing
c) To facilitate user interaction through effective layout and design
d) To increase the speed of the application

Question 4: Which of the following best describes a database query?

a) A method to insert new data into the database.
b) A request for information from the database.
c) A way to modify the user interface.
d) A tool to speed up the server response.

Question 5: How are business processes typically optimised with software applications?

a) By decreasing the number of software users.
b) By automating repetitive tasks and streamlining workflows.
c) By increasing the complexity of tasks.
d) By reducing the interaction between different business departments.

Question 6: In programming, what is a function?

a) A named section of a program that performs a specific task.
b) A variable that stores data.
c) A control structure that directs the flow of the program.
d) A type of user interface element.

Co-funded by the European Union

Question 7: Which skill is crucial for designing solutions using low-code platforms?

a) Ability to perform complex mathematical calculations
b) Skill in manual drawing and sketching
c) Problem-solving and critical thinking
d) Advanced knowledge of server maintenance

Question 8: What is an advantage of having experience with any software development tool before learning low-code development?

a) It is necessary for using low-code platforms.
b) It helps understand the underlying principles of application development.
c) It replaces the need for traditional programming knowledge.
d) It fulfills academic requirements for computer science degrees.

Co-funded by the European Union

DETAILED EXPLORATION

Understanding Low-Code

Low-code was first defined in a 2014 Forrester Research paper, "New Development Platforms Emerge for Customer-Facing Applications" by Richardson C. et al. (Bock & Frank, 2021; Bucaioni, Cicchetti, & Ciccozzi, 2022; Di Ruscio, et al., 2022). Forrester characterises low-code platforms as: "products and/or cloud services for application development that employ visual, declarative techniques instead of programming and are available to customers at low- or no-cost in money and training time to begin, with costs rising in proportion of the business value of the platforms" (Richardson, et al., 2014). Since then, current literature has provided various definitions, which have been enhanced in recent years (Rokis & Kirikova, 2023).

The approach of low-code development (LCD) is designed to simplify the software/application development process and reduce the need for extensive coding skills (Woo, 2020). In other words, LCD tools allow for software development with minimal coding, as the name suggests. Thus, the LCD give the opportunity to programmers and practitioners to develop applications by focusing on design aesthetics and functionality, while reducing the amount of coding required (Waszkowski, 2019). However, code is typically generated automatically in the background and can be customized by IT staff (Wang, et al., 2021).

Another definition provided by Alamin, et al. (2021) indicates that LCD is a process that involves minimal manual coding, visual programming using a graphical interface, and model-based design for developing software. It embraces the principles of end-user software programming and allows practitioners with different backgrounds and levels of software development experience to participate in software development activities. (Background Note: These definitions are useful for students interested in research or theory.

Co-funded by the European Union

For practical use, it's enough to understand that low-code = visual development + minimal coding.)

The aim and justification of LCD tools is usually to assist organisations in constructing and deploying applications without requiring a highly skilled team of developers (Tariq, 2021). However, it is important to note that LCD tools are not exclusively used by non-programmers. Software developers also use LCD tools, such as Jenkins, to continuously build and test software releases (Lethbridge, 2021).

Low-code development enables individuals with limited programming skills to create and implement digital solutions that address specific business needs, thereby increasing business agility. This approach allows professionals or even end-users closer to the business to implement digital solutions to their business challenges, reducing the technical barriers of programming. Some authors refer to these individuals as 'Citizen Developers' (Everhard, 2019; Olariu, et al., 2015; Woo, 2020). Further analysis of Citizen Developers and their required knowledge and skills will be conducted in a following section. Briefly, in practice, Citizen Developers interact with a low-code platform using an application modeler featuring an intuitive graphical interface with several predefined elements to aid in the development process. Users can drag and drop these constructs into the graphical interface, which automatically generates code in the background. Thus, individuals lacking advanced coding skills can create and implement fully functional applications or digital solutions without relying on software developers (Sahay, et al., 2020).

LCD is gaining significant momentum, despite the ongoing debate about its origins. Market reports predict that by 2025, nearly 70% of all business applications will be developed using some form of low-code (Gartner, 2021), and over 90% of organisations will either be in use or considering the use of low-code platforms (MarketsandMarkets, 2020). If these predictions are

Co-funded by the European Union

correct, low-code will have a major impact on the digitalisation of various industries. It will shape the way in which operations and projects are managed.

Multiple Choice Quiz

Question 1: What year was low-code first defined by Forrester Research?

a) 2010
b) 2012
c) 2014
d) 2016

Question 2: Which of the following is NOT a characteristic of low-code platforms according to Forrester?

a) Visual, declarative techniques
b) High cost and extensive training required
c) Minimal manual coding
d) Availability at low or no initial cost

Question 3: What does LCD stand for in the context of software development?

a) Liquid Crystal Display
b) Low-Cost Development
c) Low-Code Development
d) Long-Code Development

Question 4: Which type of professional is especially empowered by low-code development tools?

a) Only experienced software developers
b) Only IT support staff

Co-funded by the European Union

c) Citizen Developers
d) Only graphic designers

Question 5: According to market reports, what percentage of business applications will be developed using some form of low-code by 2025?

a) 50%
b) 60%
c) 70%
d) 80%

True/False Exercise

1. Low-code development tools primarily use textual programming languages to create applications.

2. Low-code development allows for the customization of automatically generated code by IT staff.

3. Citizen Developers typically have advanced programming skills and can handle complex software development projects alone.

4. Low-code platforms reduce the barriers to software development by allowing users to employ visual and model-based design techniques.

5. Low-code development tools support real-time feedback, enabling instant visualization of changes.

The Importance of Low-Code

In the current digital era, it is evident that education needs to be approached differently (UNESCO, 2020). It is crucial to prepare students to be competent and confident in engaging with digital technologies, which may lead to identifying opportunities in STEM careers. The notion that everyone must learn to code to some extent (Resnick, 2013) is widely accepted, particularly in

Education 4.0. Technology is now ubiquitous in society, creating new opportunities and challenges for learning, as well as for post-education professional development (Salmon, 2019).

The ubiquity of technology in our lives underscores the necessity of integrating LCD into academic curricula. LCD offers a gateway for students to engage in software development without the steep learning curve traditionally associated with programming. By incorporating LCD into educational programs, we enable a broader spectrum of students to develop the digital competence required in today's job market. Through LCD, students can visualise complex processes, create functional applications, and participate actively in the digital transformation of industries. The hands-on experience with LCD tools not only solidifies their understanding of theoretical concepts but also instils a practical skill set that is both versatile and highly sought after. Fostering a mindset of innovation and adaptability, qualities imperative for students who will soon enter a workforce where technology and change are the only constants.

Specifically, LCD is quite appealing to the academia for many specific reasons. Its utility extends beyond mere application development, offering profound educational benefits and skill development opportunities for students. In more detail:

- Democratisation of Technology: Low-code platforms break down the barriers to digital innovation by allowing students from non-technical backgrounds to create and deploy applications. This democratisation fosters a culture of inclusivity and empowerment, where ideas can be transformed into digital solutions regardless of one's programming expertise. It encourages a wider range of students to engage with technology, thereby fostering a diverse environment of digital creators and problem solvers.

Co-funded by the European Union

- Enhancing Digital Literacy: As digital literacy becomes as fundamental as reading and writing, low-code platforms serve as an excellent tool for teaching these essential skills. Students learn to conceptualise, design, and implement digital solutions, gaining a practical understanding of how software impacts various aspects of society and business. This hands-on experience is invaluable, providing students with a skill set that is increasingly important in a digital-centric world.

- Fostering Creativity and Innovation: By removing the complexity of traditional coding, low-code platforms allow students to focus on the design and functionality of their applications. This shift encourages students to think more about user experience, design thinking, and the application's impact, which are crucial aspects of technology development. The ability to quickly iterate and experiment with different ideas on low-code platforms also fosters a culture of innovation and continuous improvement.

- Collaboration Across Disciplines: Low-code platforms can be a unifying tool for students from various disciplines to collaborate on projects. For example, a team comprising business, design, and sociology students can work together to create an app that addresses a specific social issue. This kind of interdisciplinary collaboration mirrors the real-world scenarios where diverse teams come together to solve complex problems, thereby preparing students for the collaborative nature of the modern workplace.

- Rapid Prototyping and Problem Solving: The ability to quickly build and modify applications on low-code platforms allows students to engage in rapid prototyping. This process encourages a problem-solving approach where students can immediately see the results of their work, test different solutions, and learn from their mistakes in a risk-free

Co-funded by the European Union

environment. It is a dynamic form of learning that emphasises practical experience and learning by doing.

- Skill Development for the Future Workforce: Low-code platforms help students develop not just technical skills, but also soft skills such as problem-solving, critical thinking, and project management. As students design and iterate on their digital solutions, they learn to identify and solve problems, manage projects from conception to completion, and think critically about the design and user experience of their applications.

- Understanding of Digital Transformation: By using low-code platforms, students gain insights into the processes and benefits of digital transformation across various industries. They learn how businesses leverage technology to innovate, improve efficiency, and stay competitive. This knowledge is crucial for students, as it prepares them to be contributors to digital transformation efforts in their future careers.

- Preparation for Diverse Roles: As the industry increasingly adopts low-code and no-code solutions, there's a growing need for professionals who can use these tools to drive business innovation and efficiency. Students with experience in low-code platforms are well-prepared for these roles, which may include business analysts, project managers, or digital transformation specialists, even if they don't hold traditional programming roles.

Co-funded by the European Union

Scenario-Based Questions

Scenario 1: A sociology student, Mia, wants to create an app to gather data on community resource usage. She has no prior coding experience.

Question: How can low-code development platforms assist Mia in her project, and what unique perspective might she bring to the application development process?

Scenario 2: During a group project, students from business, design, and computer science disciplines are tasked with creating a prototype for a start-up idea. They need to work collaboratively to meet a tight deadline.

Question: How can a low-code platform facilitate their collaboration, and what roles might each student assume in this interdisciplinary team?

Scenario 3: Alex, a student, has come up with a concept for a mobile app that helps users reduce their carbon footprint. He needs to quickly iterate on his design based on user feedback.

Question: What features of low-code development will allow Alex to prototype, test, and refine his application swiftly?

Scenario 4: A group of students is involved in a capstone project that requires them to demonstrate not only technical skills but also project management and problem-solving abilities.

Co-funded by the European Union

Question: In what ways can engaging with a low-code platform during this project enhance these students' soft skills, and how might this experience benefit them in their future careers?

Components of Low-Code Platforms

In accordance with the aforementioned definitions and explanations, LCD is inextricably linked with LCD platforms. Hence, it could be asserted that low-code development is a tool-based approach (Rafi, et al., 2022). LCD platforms encompass application development, deployment, lifecycle management, and platform management features (Di Ruscio, et al., 2022). These platforms are either cloud-based (delivered through the Platform-as-a-Service (PaaS) model) or on-premises-based. LCD is accomplished on these platforms using visual tools, predefined components, and their customisation and configuration (Rokis & Kirikova, 2022; Sahay, et al., 2020).

A LCD platform can be defined as a set of interconnected components that are organised into distinct layers. (Background Note: This is mainly relevant for computer science students who want to understand platform architecture. For most learners, it is enough to know that low-code tools connect user interfaces, data, and deployment environments.) From an architectural perspective, the classification distinguishes the following layers (Sahay, et al., 2020):

- The **application layer**, which is comprised of a graphical environment, which incorporates toolboxes and widgets for the definition of user interfaces. Additionally, it encompasses mechanisms for authentication and authorisation. Users of the platform engage with this layer, employing its modelling constructs and abstractions to develop applications and define their behaviour in accordance with predefined specifications.

Co-funded by the European Union

- The **service integration layer**, which is responsible for facilitating the integration of disparate services, including application programming interfaces (APIs) and authentication mechanisms.

- The **data integration layer**, which is a dedicated component that enables the integration, operation, and manipulation of data from a multitude of disparate sources.

- The **deployment layer**, which represents the phase during which the developed application is deployed on either a dedicated cloud environment or on-premise. In collaboration with the service integration layer, the containerisation and orchestration of the developed application are performed.

Among the most significant functionalities and capabilities of LCD platforms are visual interfaces and modelling, which can be further categorised into the following specific directions (Kirchhof, et al., 2023):

Visual Interfaces

- **Drag-and-Drop Functionality**: allows users to easily build applications by dragging and dropping components onto a canvas, eliminating the need for extensive coding and making the development process more accessible to non-programmers.

- **Visual Modelling Tools**: enable users to create a visual representation of the application's workflow and data models. This approach simplifies complex processes and enhances understanding and collaboration among team members.

- **Real-time feedback**: is also provided through visual interfaces, allowing developers to see the effects of their changes instantly. This

Co-funded by the European Union

prompt feedback loop facilitates rapid iterations and adjustments, thereby significantly reducing development time.

- **Pre-built templates and UI components**: can be tailored to enhance the user experience. These tools allow developers to concentrate on usability and design aspects without being weighed down by backend complexities.

Modelling

- **Business process modelling**: allows users to design and automate business processes. This feature is critical for aligning application functionality with organisational workflows and goals.

- **Data modelling**: allows developers to graphically define and manage data structures. This approach simplifies the management of data relationships and integrity, making it easier to design responsive and data-driven applications.

- **Application logic modelling**: enables visual definition of application logic, including conditions, actions and interactions between different application components. This not only speeds up the development process, but also ensures that the application logic is transparent and easy to modify.

- **Integration modelling**: provides visual tools for defining and managing integrations with external systems and services. This feature simplifies the process of connecting the application to APIs, databases or other services, enabling a more seamless flow of data and functionality.

Co-funded by the European Union

What is more, according to Kirchhof, et al. (2023), the characteristics of LCD platforms can be divided into 4 main categories, each of which has specific features:

1. Development Features:

- **Language**: describes the language that can be used by developers to specify their application or to add hand-written code to it. If there is no domain-specific language (DSL), the language of the generated code is typically used. This feature does not relate to the language of the implementation of the low-code tools themselves.

- **Handwritten code support**: refers to the ability to embed custom code. Low-code tools may not cover all use cases due to the diversity of projects. This feature ensures that users will be able to meet their requirements, even if they fall outside of the normal use cases of the tool.

- **Hot Reload**: refers to the tool's ability to automatically apply changes to the data model and graphical user interface (GUI) without requiring manual rebuilding and reloading of the application. This feature is particularly useful during the early development phases when changes are frequent, as it allows for immediate reloading and easier testing of the changes.

- **Third-party plugins**: enable the integration of external services, such as Dropbox, ERP systems like SAP, or Salesforce, to connect generated applications with existing infrastructure.

- **Visual processor**: is responsible for graphically processing models and/or GUIs. This is typically achieved through drag-and-drop

Co-funded by the European Union

functionality and, in the case of GUIs, a '*What you see is what you get*' approach.

2. Database Technologies:

- **SQL database**: is a type of database that uses the Structured Query Language (SQL) to store data. This criterion is met even if external SQL databases can be integrated. SQL is widely used in industry, making it easier to integrate with existing SQL-based applications.

- **Graph databases**: store data in a graph structure, which can also include external databases. They are particularly useful for managing highly complex relationships between data, such as those found in social network applications.

- **Storing data in Key-Value or Document-oriented Databases**: is a key feature, which will also be in place if external databases can be integrated. These databases are commonly used for unstructured data and when horizontal scalability is required. Examples of such databases include Google Firestore, Amazon DynamoDB, and MongoDB.

3. Type of Target App:

- **Desktop**: refers to a native application created for desktop computers, such as Windows, macOS, and/or Linux. It excludes web applications unless they are installed as separate applications independent of the browser.

- **Web Application**: refers to when the low-code platform creates an application that runs in the browser.

- **iOS**: refers to the creation of a native iOS application, which can potentially be downloaded from the app store. This excludes responsive

Co-funded by the European Union

web applications. Responsive web applications adapt their interface to the size of mobile devices without being installed on the device.

- **Android**: refers to the creation of a native Android application using the low-code platform, which can potentially be downloaded from the app store. This excludes responsive web applications.

4. Features of Generated App:

- **Live Collaboration**: refers to the ability for multiple users to work on the same data simultaneously and view real-time changes made by others. Examples include Google Docs or Overleaf, where word processing users can see live cursors and changes made by other users working on the same document. Live collaboration can facilitate the synchronisation of users' work, particularly when there are a larger number of users.

- **REST API**: describes the property that the generated application is also operable via a REST API that follows the design principles of the Representational State Transfer (REST) architectural style, for example, it is possible to enter new data. Users can automate their tasks using REST APIs.

5. Deployment & Operation**:**

- **Serverless / cloud-native**: describes the ability to run applications in the cloud without the need to manage servers. Application management can be very complicated. Keeping this to a minimum is therefore desirable, especially for non-technical users. This feature refers also to if the tool provider itself acts as the application's host, thus eliminating the need for administration.

Co-funded by the European Union

- **Monitoring**: refers to the ability to technically monitor how the generated application runs, e.g. to view traffic or CPU load. In particular, it does not include services such as Google Analytics, which does not monitor how the application itself runs, but rather how the user behaves. Such monitoring can be used to identify and correct problems.

Some more modern software solutions, such as **the LightCode platform**, are by design not structured by layers and follow the Domain-Driven-Design approach. Nevertheless, the layered architecture presented here remains a useful conceptual reference and layers can be globally considered as frequent features.

Multiple Choice Quiz

Question 1: What is the primary deployment model for LCD platforms?

a) Only on-premises
b) Only cloud-based
c) Either cloud-based or on-premises
d) Neither cloud-based nor on-premises

Question 2: Which layer of a LCD platform is primarily responsible for user interaction and interface creation?

a) Application layer
b) Service integration layer
c) Data integration layer
d) Deployment layer

Co-funded by the European Union

Question 3: What functionality does the service integration layer in LCD platforms provide?

a) Data manipulation
b) User interface creation
c) Integration of external services like APIs
d) Deployment of applications

Question 4: What does the deployment layer in a LCD platform typically handle?

a) Building the application's frontend
b) Managing the application's database interactions
c) Deploying the application in a cloud or on-premises environment
d) Integrating user feedback into application design

Question 5: Which feature of LCD platforms is aimed at reducing the need for extensive coding?

a) Drag-and-Drop Functionality
b) Data modelling
c) Hot Reload
d) REST API

Question 6: What is the primary purpose of visual modelling tools within LCD platforms?

a) To provide real-time feedback
b) To simplify complex processes and enhance team collaboration
c) To store data using SQL
d) To manage application deployment

Question 7: Which of the following is not a main category of characteristics for LCD platforms?

Co-funded by the European Union

a) Development Features
b) Database Technologies
c) User Experience Customizations
d) Type of Target App

Question 8: What is the function of business process modelling in LCD platforms?

a) To design and automate business processes
b) To create visual interfaces
c) To deploy applications
d) To integrate services like Dropbox or SAP

Question 9: Which type of database is particularly useful for managing highly complex relationships between data, as used in social network applications?

a) SQL database
b) Graph database
c) Key-Value database
d) Document-oriented database

Question 10: What does the REST API feature in LCD platforms allow?

a) Visual modelling of data
b) Automatic reloading and testing of changes
c) Operation of applications via a specific architectural style
d) Cloud-based deployment of applications

Question 11: Which feature describes the ability to run applications without managing servers, typically in a cloud environment?

a) Serverless / cloud-native
b) Live Collaboration

Co-funded by the European Union

c) Visual processor
d) Handwritten code support

Question 12: What does the 'Hot Reload' feature in LCD platforms enable?

a) Immediate updates to the data model and GUI without manual rebuilding
b) Integration with third-party services
c) Synchronisation of live data among multiple users
d) Monitoring of application performance and usage

Applications and Opportunities

In recent years, numerous tools for rapid application development have been introduced. The following list comprises a variety of LCD platforms, presented in alphabetical order without the names of the companies. The URLs for each platform are provided for further reference. It is also worth noting that all of the following platforms are all fee-based, which is the reason for the existence of the LightCode project - to offer a LCD platform for educational purposes without any cost at all.

- **A12** is web app development framework. One of the most notable applications developed using A12 is the German federal tax reporting software, Elster.

- **AWS Amplify Studio** is tool for mobile and web application development. Builders can use Amplify CLI to integrate their applications with other AWS services. The GUI components provided can be customised using Figma, which is an interface design tool.

Co-funded by the European Union

- **Appian** is focused on process automation. In addition to enabling developers to create dashboards using a drag-and-drop visual editor, forms for data, and activity diagrams for processes, it also includes process features. Its strength is its ability to facilitate rapid development without compromising power or control, which makes it suitable for educational institutions seeking to develop robust applications for operations, research and learning management.

- **BettyBlocks** is a low/no-code platform that enables the building of web applications. The platform features a drag-and-drop visual editor for defining dashboards, online forms for defining the data model, and activity diagrams for defining actions. It provides integrations for various third-party services, including Twitter, Google Maps, and Slack.

- **Caspio** is a low/no-code tool for building business applications that focus on databases. The data can be visualised on dashboards, and a visual drag-and-drop editor is used for modelling data structures with class diagrams.

- **Claris FileMaker** is a database application that enables the development of high-performing, scalable, and custom applications. It integrates a database engine with a graphical user interface (GUI) and security features, allowing users to visually modify a database. Users can organise data into screens, layouts or forms, and manage contacts and projects.

- **DWKit** is a self-hosted or cloud platform for web and mobile application development. It allows the user to model complex business processes and handle complex business scenarios easily with simple drag-and-drop tools.

Co-funded by the European Union

- **Fliplet** is a tool for creating mobile and web applications. The drag-and-drop visual editor makes it easy to create GUIs and edit data using worksheets. Fliplet offers a variety of integrations and features, including push notifications, email notifications, SMS notifications, analytics, and single sign-on.

- **Google AppSheet** is a no-code tool that enables users to automate workflows. The platform features an activity-diagram-like workflow editor that allows users to create automations. AppSheet can be integrated with third-party apps and data sources such as Google Sheets, SQL databases, and Dropbox.

- **KissFlow** is a cloud-based platform that focuses on workflow and process automation, enabling business users to design, create, and customise business applications with little to no coding. It uses visual tools and custom scripting to develop and deliver complex applications at scale.

- **Mendix** is a platform for the development of web applications and mobile applications. It allows developers to define the graphical user interface (GUI) using a drag-and-drop visual editor, and the backend using class diagrams, which are called Domain Models, and activity diagrams, which are called Microflows. Additionally, it supports integration with other systems, such as SAP, through various third-party extensions.

- **Microsoft Power Apps** is a suite of applications, services and connectors, and a data platform that is a rapid development environment for the creation of custom business applications. It offers numerous integrations with other widely used Microsoft products in business scenarios. It is possible to integrate over 800 APIs using connectors.

Co-funded by the European Union

Power Apps is frequently combined with other services from Microsoft Azure, the company's cloud platform. With Power Apps, the users can quickly create apps that connect to their data stored in either the underlying data platform (Microsoft Dataverse) or various online and on-premises data sources (such as SharePoint, Microsoft 365, Dynamics 365, SQL Server, etc.). It can offer extensive business logic and workflow capabilities to convert your manual business operations into digital, automated processes.

- **Nintex** is a cloud-based no-code platform that allows users to manage processes and create workflows. It is a process management software that assists users in mapping, monitoring, and optimising business processes. The platform facilitates efficient collaboration among business professionals.

- **Oracle APEX** enables the development of enterprise web and mobile applications. It enables users to create a database-driven application, customise its user interface, and grant access to the application via URL. It provides a visual drag-and-drop editor for GUIs and integrates seamlessly with other Oracle technologies such as Oracle SQL databases.

- **OutSystems**, also known as Service Studio, is being used for building mobile and web applications. It offers drag-and-drop visual editors. Class diagrams are used to define data structures. The GUI can be partially generated from these structures and then manually customised. Activity diagrams can be used to specify the behaviour of the application.

- **Pega** is focused on workflow automation. Processes are defined by developers using a drag-and-drop visual editor for business process models. The tool allows for the visualisation and analysis of processes through the use of artificial intelligence.

Co-funded by the European Union

- **Robocoder Rintagi** is an open-source platform that allows users to develop fully-customised mission-critical systems for financial institutions and enterprises. The platform features visual, drag-and-drop interfaces that enable non-developers and developers to work together to build business-critical applications with speed, ease, and high performance.

- **Skyve Foundry** is an open-source cloud-based platform for custom applications. It focuses on a data-centric design, allowing users to define data models that automatically generate user interfaces and application components, while also offering customisation and integration capabilities to cater to specific business needs.

- **Temenos**, also known as Kony, provides a low-code platform focused on digital banking. Users can create modern web apps, true native apps, chat apps, mobile devices, immersive experiences, and other digital experiences using a low-code approach. Temenos provides a long list of reusable components, integration links and visual tools that enable companies to deliver their products faster and keep their customers engaged.

- **SIB Visions VisionX** is a platform that allows users to visually develop entire applications, integrate with existing systems, and add custom code when necessary. It enables rapid development of web, desktop and native mobile applications by both business users and professional developers. These applications can range from simple replacements for paper processes or Excel sheets to user-friendly forms on ERP systems, dashboards, mobile apps, and even highly complex billing applications, customer portals, or trading systems. The user retains ownership of the generated Java code, which utilises only open-source libraries.

Co-funded by the European Union

- **Wavemaker** enables the development of mobile and web-based applications. It provides a drag-and-drop visual editor for GUIs and generates code for React Native. This code is not tied to the Wavemaker platform, which prevents vendor lock-in.

- **Wix Editor X** is a web design and development platform that combines no-code and low-code capabilities. It offers advanced design and layout features that enable the creation of complex website designs for any device, without requiring coding. However, Wix has stopped supporting Editor X and has transitioned to the Wix Studio Platform, a user-friendly drag-and-drop editor for web design. This platform allows users to easily customise their sites without requiring any coding knowledge.

- **Quickbase** is a no-code platform designed to enable dynamic teams to solve complex problems. It provides a database engine that combines relational and graph structures to facilitate both transactions and analytics. The platform offers drag-and-drop functionality, customisable templates and pre-built components, allowing users to build and deploy applications with simplified user interfaces.

- **Zoho Creator** is a platform that enables users to build multi-platform applications to meet their unique business needs. It offers a drag-and-drop interface, robust data management, automation workflows, and integration capabilities, making it a comprehensive tool for developing applications that streamline business processes and enhance productivity.

- Zvolv focuses on process automation. It allows users to model processes using activity diagrams and then partially automate them using bots that can either instantiate and configure predefined actions or use hand-

Co-funded by the European Union

written Python code. Zvolv also offers integrations with third-party tools such as Dropbox, SAP, or Slack, and data about the processes can be displayed in dashboards.

It is worth mentioning that some platforms are beginning to integrate artificial intelligence (AI) to more accurately predict user preferences. For instance, the low-code platform Mendix (Boston, MA, USA) has an assistant that uses machine learning to identify patterns in historical data and offer recommendations to users during app creation (Den Haan, 2018).

There is a high demand for low-code app development, which is understandable as mobile apps usually have short lead times and their development can be accelerated using LCD. However, it is important to note that different LCD platforms support the development of different types of applications. For example, one LCD platform can provide a method for developing web apps without coding, another platform can be leveraged to design interfaces for iOS apps without code, and another can connect to any service and provide an integrated set of links out of the box. Therefore, it is essential to select the appropriate LCD platforms for the project based on the requirements of the application type before development (Luo, et al., 2021).

Case Studies

Case Study 1: Health Monitoring Application

Scenario:

A healthcare startup wants to create a mobile app that allows patients to monitor their health vitals and share this data with their doctors in real-time. The startup has limited coding resources and needs to deploy a functional prototype quickly to secure funding.

Co-funded by the European Union

Open-ended questions:

- Which low-code platform from the list would be most suitable for creating this health monitoring app and why?

- Considering the need for real-time data sharing, which features of the chosen platform will ensure secure and efficient communication between patients and doctors?

- How can the startup leverage the low-code platform's integration capabilities to connect with existing healthcare databases or APIs?

Case Study 2: Educational Management System

Scenario:

An educational institution aims to develop a comprehensive management system that integrates student records, course scheduling, and online assessments. They seek a flexible solution that can adapt to changing educational needs and standards.

Open-ended questions:

- Which low-code platform would be ideal for developing an educational management system that requires frequent updates and customizations?

- How would the drag-and-drop functionality and data modeling tools of the chosen platform benefit the development of this system?

- What are the potential challenges in using a low-code platform for such a complex system, and how can they be mitigated?

Co-funded by the European Union

Looking Ahead

The future trends in low-code technology, particularly in the context of enhancing student employability and enabling academia to keep pace with rapid technological advancements, are poised to significantly influence educational outcomes and career readiness. A few important projections of how low-code can shape these aspects are the following:

- Bridging the Skills Gap: Low-code platforms can play a crucial role in addressing the skills gap in the technology sector by providing students with practical, hands-on experience in application development, even if they do not have a deep background in coding. This experience makes them attractive to employers looking for candidates who can contribute to digital projects without the need for extensive on-the-job training in traditional programming languages.

- Encouraging Entrepreneurial Initiatives: As students gain proficiency in low-code platforms, they can develop entrepreneurial skills by initiating start-ups or tech ventures while still in higher education. These platforms lower the entry barrier to technology innovation, allowing students to prototype and test their business ideas rapidly, making them more prepared and confident to enter the entrepreneurial world post-graduation.

- Facilitating Internships and Real-World Projects: Educational institutions can collaborate with industry partners to provide students with internship opportunities where they use low-code platforms to work on real-world projects. This direct industry engagement allows students to understand the practical applications of their learning, making them more employable and better prepared to meet the demands of the workforce.

Co-funded by the European Union

- Enhancing Interdisciplinary Competencies: By enabling students from non-technical disciplines to create digital solutions, low-code platforms foster interdisciplinary competencies, preparing students for a workforce that increasingly values the convergence of technology with other fields. For example, a marketing student who can develop a customer engagement app or a healthcare student who can create a patient data management system brings a unique skill set to their respective industries.

- Supporting Continuous Learning and Adaptability: The rapid pace of technological change demands continuous learning and adaptability from the workforce. Low-code platforms can help students develop a mindset of lifelong learning and adaptability, as they get accustomed to rapidly mastering new tools and technologies, a skill highly valued in today's dynamic job market.

- Alignment with Emerging Industry Trends: As industries increasingly adopt low-code and no-code solutions, students familiar with these platforms will be well-aligned with emerging trends, making them highly desirable in the job market. Their ability to quickly adapt and create with low-code tools positions them as valuable assets to employers looking to leverage these technologies for digital transformation and innovation.

- Global Competitiveness: By mastering low-code platforms, students can compete in a global job market, where the ability to quickly deliver digital solutions is increasingly critical. Their low-code skills can open up international job opportunities, remote work, and collaborations with global teams, enhancing their career prospects and global competitiveness.

Co-funded by the European Union

True/False Exercise

1. Low-code platforms only benefit students with an existing background in coding and technology.

2. Gaining experience in low-code development can make students more attractive to potential employers who value digital skills.

3. Low-code platforms increase the barrier to entry into technology innovation, making it harder for students to launch start-ups.

4. Internships that involve low-code platforms do not provide real-world experience relevant to current industry standards.

5. Students from non-technical fields cannot benefit from learning low-code development because it does not apply to their areas of study.

6. Using low-code platforms can help students develop interdisciplinary skills that are valuable in today's diverse job market.

7. Continuous learning and adaptability are not necessary skills in the modern workforce, making low-code platforms less relevant.

8. Familiarity with low-code and no-code solutions aligns students with emerging industry trends and increases their marketability.

9. Mastering low-code platforms offers no advantage in global job markets or in securing remote work opportunities.

10. The rapid development capabilities of low-code platforms can hinder a student's ability to engage in thorough and detailed project management.

Co-funded by the European Union

CONCLUSION

The exploration of LCD presents a transformative potential for the educational landscape, particularly in enhancing the capabilities of faculty and students within the framework of the LightCode Project. This initiative, aimed at fostering innovative teaching methods and interdisciplinary approaches, aligns perfectly with the core benefits of LCD. Such tools and platforms simplify the software development process, making it accessible and engaging for users regardless of their technical background.

Low-code development democratises technology creation, allowing students and faculty from various disciplines to actively participate in building digital solutions. This accessibility is crucial for forming an open and innovative higher education system where all participants are empowered to contribute creatively to technology projects. LCD's intuitive design, drag-and-drop functionality, and pre-built templates allow users to focus more on the conceptual and design aspects of projects rather than on complex coding. This shift not only enhances digital literacy but also promotes a deeper understanding of how technology can be used to solve real-world problems.

For students, LCD approach serve as a critical tool in bridging the skills gap in the technology sector. The ability to quickly develop and deploy applications positions students as capable participants in technology-driven fields, ready to meet the demands of modern industries without requiring extensive coding expertise. Moreover, the project's emphasis on innovative teaching methods is complemented by LCD's suitability for rapid prototyping and iterative learning. Educators can incorporate these platforms into their curricula to teach critical thinking, problem-solving, and project management skills effectively. Students can immediately see the impact of their ideas, receive real-time feedback, and refine their solutions quickly, which enhances the learning process and ingrains a culture of continuous improvement and innovation.

Co-funded by the European Union

CORRECT ANSWERS

Multiple Choice Quiz (pages 6-8)

Question 1: b
Question 2: b
Question 3: c
Question 4: b
Question 5: b
Question 6: a
Question 7: c
Question 8: b

Multiple Choice Quiz (pages 11-12)

Question 1: c
Question 2: b
Question 3: c
Question 4: c
Question 5: c

True/False Exercise (page 12)

1: False
2: True
3: False
4: True
5: True

Multiple Choice Quiz (page 23-26)

Question 1: c
Question 2: a

Co-funded by the European Union

Question 3: c

Question 4: c

Question 5: a

Question 6: b

Question 7: c

Question 8: a

Question 9: b

Question 10: c

Question 11: a

Question 12: a

True/False Exercise (page 36)

1: False

2: True

3: False

4: False

5: False

6: True

7: False

8: True

9: False

10: False

Co-funded by the European Union

REFERENCES

Alamin, M. A., Malakar, S., Uddin, G., Afroz, S., Haider, T. B., & Iqbal, A. (2021). An empirical study of developer discussions on low-code software development challenges. *IEEE/ACM 18th International Conference On Mining Software Repositories (MSR)*, (pp. 46-57). Madrid.

Bock, A. C., & Frank, U. (2021). Low-Code Platform. *Business and Information Systems Engineering, 63*(6), 733–740.

Bucaioni, A., Cicchetti, A., & Ciccozzi, F. (2022). Modelling in low-code development: a multi-vocal systematic review. *Software System Model*.

Den Haan, J. (2018, June 19). *Introducing AI-assisted development to elevate low-code platforms to the next level.* Retrieved from https://www.mendix.com/blog/introducing-ai-assisted-development-to-elevate-low-code-platforms-to-the-next-level/

Di Ruscio, D., Kolovos, D., de Lara, J., Pierantonio, A., Tisi, M., & Wimmer, M. (2022). Low-code development and model-driven engineering: Two sides of the same coin? *Software System Model, 21*(2), 437–446.

Di Ruscio, D., Kolovos, D., de Lara, J., Pierantonio, A., Tisi, M., & Wimmer, M. (2022). Low-code development and model-driven engineering: Two sides of the same coin? *Software System Model*, 437–446.

Everhard, J. (2019, January 22). *The Pros And Cons Of Citizen Development*. Retrieved from Forbes: https://www.forbes.com/sites/johneverhard/2019/01/22/the-pros-and-cons-of-citizen-development/?sh=4d04dfae84fd

Gartner. (2021, September 20). *Gartner Magic Quadrant for Enterprise Low-Code Application Platforms* . Retrieved from Gartner: https://www.gartner.com/en/documents/4005939

Co-funded by the European Union

Kirchhof, J. C., Jansen, N., Rumpe, B., & Wortmann, A. (2023). Navigating the Low-Code Landscape: A Comparison of Development Platforms. *2023 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)* (pp. 854-862). Västerås, Sweden: The Institute of Electrical and Electronics Engineers, Inc. .

Lethbridge, T. C. (2021). Low-code is often high-code, so we must design low-code platforms to enable proper software engineering. *International Symposium on Leveraging Applications of Formal Methods*, 202-212.

Luo, Y., Liang, P., Wang, C., Shahin, M., & Zhan, J. (2021). Characteristics and Challenges of Low-Code Development: The Practitioners' PerspectiveCharacteristics and Challenges of Low-Code Development: The Practitioners' Perspective. *15th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM).* Bari, Italy: Association for Computing Machinery.

MarketsandMarkets. (2020). *Low-Code Development Platform Market by Component (Platform and Services), Application Type, Deployment Type (Cloud and On-Premises), Organization Size (SMEs and Large Enterprises), Industry, and Region - Global Forecast to 2025.*

Olariu, C., Gogan, M., & Rennung, F. (2015). Switching the Center of Software Development from IT to Business Experts Using Intelligent Business Process Management Suites. *Soft Computing Applications*, 993–1001.

Rafi, S., Akbar, M. A., Sánchez-Gordón, M., & Colomo-Palacios,, R. (2022). DevOps Practitioners' Perceptions of the Low-code Trend. *ESEM '22: Proceedings of the 16th ACM / IEEE International Symposium on Empirical Software Engineering and Measurement* (pp. 301–306). New York: Association for Computing Machinery.

Co-funded by the European Union

Resnick, M. (2013, May 8). *Learn to code, code to learn*. Retrieved from EdSurge: https://www.edsurge.com/news/2013-05-08-learn-to-code-code-to-learn

Richardson, C., & Rymer, J. (2014). *New Development Platforms Emerge For Customer-Facing Application.* Forrester.

Richardson, C., Rymer, J. R., Mines, C., Cullen, A., & Whittaker, D. (2014). *New development platforms emerge for customer-facing applications.* Cambridge, MA, USA: Forrester.

Rokis, K., & Kirikova, M. (2022). Challenges of Low-Code/No-Code Software Development: A Literature Review. In E. Nazaruka, Ē. Sandkuhl, & U. Seigerroth (Eds.), *Lecture Notes in Business Information Processing.* Cham: Springer.

Rokis, K., & Kirikova, M. (2023). Exploring Low-Code Development: A Comprehensive Literature Review. *Complex Systems Informatics and Modeling Quarterly*(36), 68–86.

Sahay, A., Indamutsa, A., Di Ruscio, D., & Pierantonio, A. (2020). Supporting the understanding and comparison of low-code development platforms. *46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, (pp. 171-178). Portoroz, Slovenia.

Sahay, A., Indamutsa, A., Di Ruscio, D., & Pierantonio, A. (2020). Supporting the understanding and comparison of low-code development platforms. *2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, (pp. 171-178). Portoroz.

Salmon, G. (2019). May the fourth be with you: Creating education 4.0. *Journal of Learning for Development, 6*(2), 95–115.

Tariq, H. (2021). *Council Post: Low-Code Versus No-Code And The Future Of Application Development.* Forbes.

UNESCO (United Nations Educational, Scientific and Cultural Organization). (2020). Education in a post-COVID-19 world: Nine ideas for public action. *International Commission on the Futures of Education*.

Wang, Y., Feng, Y., Zhang, M., & Sun, P. (2021). The Necessity of Low-code Engineering for Industrial Software Development: A Case Study and Reflections. *2021 IEEE ISSREW*, 415–420.

Waszkowski, R. (2019). Low-code platform for automating business processes in manufacturing. *IFAC-PapersOnLine, 52*(10), 376-381.

Woo, M. (2020). The Rise of No/Low Code Software Development—No Experience Needed? *Engineering, 6*(9), 960-961.

Woo, M. (2020). The Rise of No/Low Code Software Development—No Experience Needed? *Engineering, 6*(9), 960-961.