



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

5. Understanding your needs according to your expertise

REACH Innovation



Dauphine | PSL
UNIVERSITÉ PARIS



KarmicSoft symplexis

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
Strengthening the Digital Transformation
of Higher Education Through Low-Code



Co-funded by
the European Union

PROJECT'S COORDINATOR

Dauphine | PSL 
UNIVERSITÉ PARIS

**Paris Dauphine
University**
France

dauphine.psl.eu

PROJECT'S PARTNERS



University of Macedonia
Greece

www.uom.gr



University of Niš
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

Symplexis
Greece

symplexis.eu

TABLE OF CONTENTS

UNDERSTANDING ONE'S NEEDS ACCORDING TO ONE'S EXPERTISE	4
PERSONALIZING LOW-CODE LEARNING	7
LOW-CODE ACROSS EXPERTISE LEVELS	13
INTEGRATING LOW-CODE IN DIVERSE FIELDS	16
BRIDGING KNOWLEDGE GAPS WITH LOW-CODE	18
CREATING A PERSONALIZED LOW-CODE ROADMAP	21
CONCLUSION	24
REFERENCES	25

UNDERSTANDING ONE'S NEEDS ACCORDING TO ONE'S EXPERTISE

This module is tailored to different disciplines and expertise levels. It helps students identify how low-code can specifically benefit them in their areas of applied studies since it provides an overview for applying low-code in various academic or professional contexts.

OVERVIEW

It can be a challenge to manage the complexity of low-code features to meet specific requirements and expertise. However, there are some important considerations that can help you match your needs with your skills when embarking on a low-code project.

Tailoring low-code features to individual skills and needs requires an understanding of the different skill levels and requirements of users within an organization. It is important to provide customized training and support to cater for beginners who need guidance on basic concepts and navigation, as well as advanced users who require in-depth knowledge of advanced features and integration techniques. Tailoring the use of low-code platforms to users' skills and specific tasks will increase your efficiency and productivity across the board.

By taking these aspects into account, the end users can provide a comprehensive understanding of their needs within a system, and the application can be developed and designed to fulfill those specific requirements.

Learning Outcomes

This chapter, developed by REACH Innovation, plays a crucial role in helping you understand how to align your learning journey with your existing expertise and future goals. In today's rapidly evolving digital landscape, mastering low-code development is not just a technical skill, it is a strategic advantage.

Organizations across industries are increasingly adopting low-code platforms to accelerate innovation, streamline processes, and empower employees who are not traditional programmers to create impactful solutions.

Real-world applications are provided to demonstrate why understanding your needs, skill level, and learning objectives is essential before diving deeper into low-code development.

This chapter will guide you through identifying your current competencies, assessing your learning priorities, and creating a personalized roadmap to make the most of low-code opportunities—whether you aim to optimize workflows, develop applications, or integrate systems in your professional context.

After completing this chapter, you will be able to:

1. Identify your current skill level and specific learning needs related to low-code development, using structured self-assessment tools.
2. Apply a self-assessment framework to design a personalized low-code learning path tailored to your goals.
3. Select relevant low-code features and tools that best match your academic or professional objectives.
4. Create a customized roadmap for skill development and future application of low-code solutions in real-world contexts.
5. Evaluate how low-code platforms can support various fields or career paths, considering efficiency, scalability, and accessibility.

These outcomes aim to ensure you are not only informed about low-code concepts but also equipped to take actionable steps towards integrating them into your studies or professional development.

Prerequisites

To get the most out of this chapter, it is highly beneficial to have completed the previous modules in the LightCode training course. These modules provide the foundational knowledge and context that will help you better understand and apply the concepts introduced here.

Having this background will enable you to fully appreciate the personalization strategies and self-assessment frameworks presented in this chapter. While no advanced programming expertise is required, the concepts from these earlier modules provide a strong base for tailoring your low-code learning journey effectively.

No advanced programming or software development experience is necessary for completing and understanding this chapter. It is designed to accommodate learners from diverse educational and professional backgrounds, making it suitable even for those with limited technical expertise.

Structure of the Module

This module is designed to help you understand your unique learning needs and how low-code development can be tailored to your expertise level, career goals, and personal aspirations. To achieve this, it is organized into five key sections. Each section addresses a specific aspect of personalization in low-code learning, providing both theoretical insights and practical tools.

Below is an overview of these sections:

1. Personalizing Low-Code Learning: Learn how to tailor your learning path using self-assessment tools to match your goals and interests.

2. Low-Code Across Expertise Levels: Discover how low-code supports beginners and experts, and choose the right strategies for your level.

3. Integrating Low-Code in Diverse Fields: Explore real-world applications of low-code across sectors like education, healthcare, and business.

4. Bridging Knowledge Gaps with Low-Code: See how low-code helps overcome skill gaps and empowers learners with limited coding experience.

5. Creating a Personalized Low-Code Roadmap: Get practical steps to design your own roadmap for applying low-code in academic and professional contexts.

PERSONALIZING LOW-CODE LEARNING

Low-code development platforms have revolutionized the way software applications are built, enabling individuals with varying levels of technical expertise to create complex solutions rapidly. However, ensuring effective learning and skill development in low-code environments requires tailoring educational approaches to match the diverse skill sets and goals of learners.

This discussion explores strategies for customizing low-code learning experiences to accommodate individual needs, backgrounds, and aspirations.

Before embarking on personalising your own low-code learning path, it is essential to assess your existing skills and familiarity with programming concepts. Individuals come from diverse educational and professional backgrounds, influencing their learning preferences and aptitudes for low-code development.

Some learners may excel in visual learning through interactive tutorials and demonstrations, while others may prefer hands-on experimentation or structured coursework.

Before designing your personalized low-code learning path, you must identify specific learning needs. A self-assessment will help you to understand your strengths, weaknesses, and learning preferences.

Based on the assessment, you can customize your learning requirements, selecting relevant content and instructional methods. This will allow you to choose goals and progress at your own pace, focusing on areas relevant to your job roles and career aspirations.

You can break down the low-code learning curriculum into modular units, which will allow you to focus on specific skills or tools relevant to your goals. Beginners may start with basic concepts and gradually progress to advanced topics, while experienced developers can dive into specialized modules.

Tailoring your low-code learning will allow you to align with different career paths, whether it deals with app development, automation, or enterprise integration. You can also learn from industry-specific case studies and real-world projects to enhance your practical skills.

Tailoring low-code learning experiences to individual skill sets and goals is essential for maximizing learning outcomes and empowering learners to harness the full potential of low-code development platforms. By adopting personalized approaches, addressing diverse learning needs, and fostering continuous skill enhancement, organizations can cultivate a skilled workforce capable of driving innovation and digital transformation in the rapidly evolving tech landscape.

To illustrate personalized low-code learning paths, some hypothetical case studies were elaborated, as described in the sequel.

Case Study 1: Beginner to Intermediate Transition

Background:

Sarah, a marketing professional with no prior coding experience, expressed interest in learning low-code development to automate repetitive tasks and enhance her analytical skills.

Personalized Learning Path:

Foundation Building: Sarah started with introductory courses covering basic programming concepts, such as variables, loops, and conditional statements, using a low-code platform's visual interface.

Hands-On Practice: She engaged in hands-on exercises and mini projects to apply newly acquired skills, building simple automation scripts and data analysis tools.

Intermediate Modules: As Sarah gained confidence, she progressed to intermediate modules focusing on more advanced features and functionalities of the low-code platform, such as integrating external APIs (Application Programming Interface), creating dynamic user interfaces, and handling data transformations.

Real-World Applications: Sarah collaborated with her team to identify specific use cases within the marketing department where low-code solutions could streamline workflows and improve productivity. She developed custom applications tailored to address these business needs, gaining practical experience and demonstrating value to her organization.

Outcome: Sarah successfully transitioned from a beginner to an intermediate level in low-code development within six months. She implemented several automation projects, including lead generation forms, email marketing campaigns, and performance dashboards, resulting in significant time and cost savings for her team. Sarah's newfound skills not only enhanced her role in marketing but also positioned her as a valuable asset capable of driving digital innovation across departments.

Case Study 2: Advanced Specialization in Enterprise Integration

Background:

John, an experienced software developer working in a large enterprise, sought to expand his expertise in low-code development to streamline integration processes and accelerate project delivery.

Personalized Learning Path:

Advanced Integration Concepts: John enrolled in advanced courses focusing on enterprise integration patterns, API management, and data synchronization techniques relevant to his organization's complex IT landscape.

Platform-Specific Training: He deepened his knowledge of a specific low-code platform used by his company, mastering its integration capabilities, connectors, and deployment strategies through intensive hands-on training.

Certification Preparation: John pursued certification in enterprise integration offered by the low-code platform vendor, completing specialized training modules and preparing for the certification exam to validate his proficiency in the subject matter.

Project-Based Learning: Leveraging real-world integration challenges within his organization, John embarked on project-based learning initiatives, collaborating with cross-functional teams to design and implement scalable integration solutions using low-code tools.

Continuous Learning: John remained actively engaged in the low-code community, attending webinars, participating in user forums, and contributing to knowledge-sharing platforms to stay updated on emerging trends and best practices in enterprise integration.

Outcome: John attained expert-level proficiency in low-code enterprise integration, earning recognition as a subject matter expert within his organization. He spearheaded several high-impact integration projects, including legacy system modernization, cloud migration initiatives, and cross-platform data synchronization, resulting in improved operational efficiency and enhanced business agility. John's advanced specialization in

low-code development positioned him as a key contributor to strategic IT initiatives, driving digital transformation and innovation across the enterprise.

- **Exercise & Quiz:** Self-assessment exercises to determine learning objectives.

Self-assessment exercises can be valuable tools to determine low-code learning objectives by helping learners identify their current knowledge, skills, and areas for improvement.

The following table can serve as a template to determine one's low-code learning objectives. The 3rd column shall be completed by the end-users.

Student Course Training Package

Skills Inventory	List the low-code platforms or tools you are familiar with.	
	Rate your proficiency level (e.g., beginner, intermediate, advanced) for each platform or tool.	
	Identify specific features or functionalities you feel confident using and areas where you lack proficiency.	
Project Reflection	Reflect on past projects or tasks you've completed in your organization.	
	Evaluate the complexity of the projects and how you could leverage low-code functionalities.	
	Identify any challenges or limitations you encountered during the projects/tasks.	
Learning Goals	Set specific learning goals based on your skills inventory and project reflection.	
	Determine the areas of low-code development you want to improve or expand upon.	
	Establish measurable objectives, such as mastering a particular feature, or completing a project with minimal assistance.	
Gap Analysis	Compare your current skills and knowledge with the requirements or expectations for low-code developers.	
	Identify any gaps or areas where you need to enhance your proficiency.	
	Prioritize the gaps based on their relevance to your learning goals and the demands of your projects or job role.	
Learning Preferences	Consider your preferred learning styles and methods (e.g., tutorial, hands-on projects, online courses).	
	Determine the resources and learning environments that best suit your preferences and schedule.	
	Explore different materials available for low-code development.	
Self Assessment	Review your performance and identify areas where you excel or need further practice.	
	Use the results to refine your learning objectives and focus on areas that require improvement.	
Peer Feedback	Seek feedback from peers or colleagues who are also engaged in low-code development.	
	Share your learning objectives and ask for input on areas where you could enhance your skills.	
	Consider collaborating on projects or participating in study groups to exchange knowledge and experiences.	

By engaging in this self-assessment exercise, you can gain valuable insights into your current proficiency level in low-code development and establish clear learning objectives to guide your skill development journey [1].

LOW-CODE ACROSS EXPERTISE LEVELS

Low-code platforms provide a visual development environment where users can create applications by dragging and dropping components. This approach significantly reduces the need for extensive coding skills. Users can build functional applications without deep technical expertise, making it accessible to a broader audience.

Low-code tools offer pre-built templates for common use cases. These templates encapsulate best practices and simplify the development process. Users can configure AI functionalities, database connections, and user interfaces without writing intricate code.

Low-code platforms abstract away complex logic, since users can define business rules, workflows, and data transformations using visual interfaces. This abstraction allows individuals with varying technical backgrounds to participate in application development.

Technical experts can extend functionality by diving deeper into the codebase, while beginners can stick to the visual tools. Low-code platforms strike a balance between control and simplicity, empowering non-technical users while accommodating advanced users. Developers can dive into code when needed, but the platform ensures that basic tasks remain straightforward.

In summary, low-code platforms democratize application development by bridging the gap between technical and non-technical users. Whether you're a seasoned developer or a business analyst, these platforms offer flexibility and efficiency, empowering people to build meaningful software [2].

- **Example:** *Comparative study of low-code usage by beginners vs. experienced professionals.* Both beginners and experienced professionals can benefit from the low-code development. Nevertheless, its usage differs in many different aspects, as described in the Table below.

	Beginners	Experienced Professionals
Interest and Adoption	No-code and low-code tools are excellent starting points for beginners. They provide a high-level understanding of software development.	They appreciate low-code's efficiency and customization options.
Learning Curve	No-code tools offer a gentle learning curve, ideal for those new to programming.	They can quickly adapt to low-code platforms due to their existing coding knowledge.
Customization vs. Speed	Prioritize ease of use and speed.	Balance customization needs with development speed.
Use Cases and Complexity	Tend to start with simpler use cases, such as building basic web applications or automating simple business processes. They may be more cautious about tackling complex projects until they gain confidence and proficiency.	More likely to tackle complex projects and enterprise-level applications using low-code platforms. They may leverage advanced features for integration, customization, and scalability.
Productivity and Time-to-Market	Low-code platforms can accelerate their development process once they become familiar with the tools. They may prioritize speed and ease of development over customization and optimization.	Can leverage low-code platforms to rapidly prototype and deploy solutions, allowing them to focus on higher-level tasks and innovation. They may prioritize efficiency, scalability, and maintainability in their development process.
Challenges and Limitations	May encounter challenges related to understanding platform features, troubleshooting errors, and translating requirements into functional solutions. They may also face limitations in customization and flexibility.	May face challenges related to platform limitations, scalability issues, and integrating low-code solutions with existing systems. They may also encounter resistance from traditional development teams or stakeholders.
Feedback and Improvement	May rely on community forums, online tutorials, and support resources to overcome challenges and improve their skills. They may actively seek feedback from peers and mentors to enhance their proficiency.	May provide valuable feedback to low-code platform providers based on their real-world experiences and requirements. They may contribute to the evolution of low-code tools by identifying areas for improvement and innovation.

In summary, you will always find value in low-code development, regardless of your level of expertise . Beginners benefit from accessibility, while experienced developers appreciate the efficiency gains.

Overall, a comparative study of low-code usage by beginners versus experienced professionals can provide valuable insights into the diverse needs, preferences, and challenges of different user groups. By understanding these differences, low-code platforms can tailor their offerings to better meet the needs of their target audiences and support users at various skill levels.

- **Exercise & Quiz:** Activities exploring low-code tools at various complexity levels.

Low-code development is about empowering creativity and problem-solving. Below a few activities are listed and participants shall be able to select which of them should be accomplished, depending on their complexity for beginner, intermediate, or advanced levels.

Complexity Level	Activity
Beginner	Create simple workflows by connecting predefined blocks.
	Experiment with visually design web applications.
	Organize data visually and build simple applications without writing code.
	Create visually stunning websites.
Intermediate	Add custom logic.
	Explore tools that require integrating with APIs or databases.
Advanced	Leverage low-code platforms to complement their existing coding skills.
	Handle intricate business processes and build complex applications

INTEGRATING LOW-CODE IN DIVERSE FIELDS

Low-code platforms democratize application development, allowing domain experts and non-programmers to create tailored solutions. Their impact spans diverse sectors, driving efficiency, innovation, and agility [3]. We explore here some examples of how low-code development can be integrated in diverse fields.

- **Low-Code for Higher Education Institutions**

The transition to remote learning accelerated during the pandemic, impacting the quality of education. Low-code platforms can empower universities and colleges to swiftly adapt by creating intuitive applications for student management, course delivery, and administrative processes.

Tight budgets often hinder institutions from investing in software development. Low-code platforms offer a cost-effective alternative by allowing enhancements to existing systems without extensive coding. They bridge the gap between legacy systems and modern requirements [4].

- **Low-Code for Healthcare and Medical Research**

Low-code applications can streamline patient data management, appointment scheduling, and communication between healthcare providers. Customizable workflows can certainly ensure more efficient operations.

Low-code platforms can be used to build data collection tools, track patient outcomes, and manage research projects. The flexibility allows rapid adjustments based on evolving study requirements.

- **Low-Code for Human Resources and Talent Management**

Low-code platforms can facilitate candidate tracking, interview scheduling, and onboarding processes. Customizable forms and workflows can enhance efficiency.

HR departments can create personalized performance appraisal systems and training modules using low-code tools. These systems align with organizational goals and individual development plans.

- **Low-Code for Financial Services and Banking**

Low-code platforms can enable banks to build customized CRM systems for managing client interactions, leads, and sales pipelines.

Financial institutions can streamline loan processes by creating low-code applications that automate credit checks, document verification, and approval workflows.

- **Low-Code for Manufacturing and Supply Chain**

Low-code solutions can enhance inventory tracking, demand forecasting, and order fulfilment. Real-time visibility can ensure efficient supply chain operations.

Manufacturers can use low-code platforms to create quality control checklists, compliance audits, and incident reporting systems.

- **Low-Code for Marketing and Customer Engagement**

Low-code tools can help marketing teams design and execute campaigns, track analytics, and personalize customer interactions.

Customizable survey applications can collect customer feedback, improving product/service offerings.

For all those fields, case-studies of low-code solutions can be exploited within customized scenario-based exercises to apply low-code in different domains (e.g.: in fields like business, healthcare, and education).

BRIDGING KNOWLEDGE GAPS WITH LOW-CODE

Low-code development democratizes application creation, enabling a wide range of users to participate in the software development lifecycle. It bridges gaps, encourages innovation, and promotes collaboration across skill sets.

We present here some examples of how low-code can help overcome specific challenges and concretely add value on results, considering different users and different levels of collaboration.

- **Non-Technical Users**

- Low-code platforms empower Citizen Developers, allowing individuals with minimal coding experience to create functional applications. Drag-and-drop components, visual interfaces, and pre-built templates simplify the process.

- Non-technical users can tailor applications to their needs without writing complex code. They can focus on business logic, user experience, and data modelling.

- **Business Analysts and Domain Experts**

- Low-code tools enable business analysts to quickly prototype ideas. They can visualize concepts, gather feedback, and iterate faster.

- Domain experts can connect disparate data sources (e.g., databases, APIs) using low-code platforms. This bridges the gap between business processes and technical systems.

- **Developers with Limited Time or Resources**

- Low-code platforms handle boilerplate code, allowing developers to focus on critical functionality. This speeds up project delivery.

- Developers can create reusable components (e.g., custom widgets, connectors) within the low-code environment. These components enhance productivity and maintainability.

- **Legacy System Modernization**

- Organizations with legacy systems can use low-code to modernize incrementally. Instead of rewriting entire applications, they can build new features alongside existing ones.

- Low-code applications seamlessly integrate with legacy databases, APIs, and services. This preserves valuable data and functionality.

- **Cross-Disciplinary Collaboration**

- Low-code platforms provide collaborative workspaces where developers, designers, and business stakeholders collaborate, fostering cross-disciplinary problem-solving.

- Low-code tools use visual representations, making it easier for diverse teams to communicate and align on requirements.

Examples

Stories of individuals who used low-code to enhance their capabilities.

In this section, you will find some inspiring stories of individuals who harnessed the power of low-code development to transform their careers and solve real-world problems.

Jacqui Peck: From Audiologist to Low-Code Developer Analyst

- Background: Jacqui Peck had a clinical masters in audiology and worked as an audiologist for thirteen years.
- Discovery: While working in remote areas, she realized the need for better data collection and communication tools.
- Low-Code Journey: Jacqui taught herself how to build Apps, using a low-code platform. She created apps for data entry, training material access, and streamlined communication across her team.
- Impact: Her apps improved recordkeeping, reduced duplication, and empowered her colleagues. Now, she's a Low-Code Developer Analyst [5].

Madison DaValle: Streamlining PR Processes

- Challenge: As a PR intern, Madison noticed the inefficiencies in PR request handling.
- Solution: Using a low-code platform, she built an app to streamline PR intake and communication.
- Collaboration: Madison partnered with IT experts and learned about app development.
- Result: Her low-code solution transformed the PR process, making it more efficient and organized [6].

More Success Stories

In [7], we find further success cases such as: *Compound* - A company that leveraged low-code to enhance its operations and achieve revenue growth; and *Keito* - A business that scaled its operations using low-code platforms, reaching impressive

annual revenue. Those stories showcase the ingenuity of individuals who embraced low-code development, even without traditional computer science backgrounds. Whether it is solving business challenges or changing career trajectories, low-code empowers creativity and problem-solving.

- **Exercise & Quiz:** Interactive tasks to identify and bridge personal knowledge gaps.

Let's put into practice your understanding of low-code development. Whether you're a beginner or an experienced professional, this exercise will help you bridge any knowledge gaps.

Problem-Solving:

- Pose real-world problems or scenarios related to your area of expertise or interest.
- Attempt to solve problems independently, applying your existing knowledge and skills.
- Identify any challenges or obstacles you encounter during the problem-solving process, which may indicate areas of weakness or gaps in your knowledge.

1. Concept Mapping:

- Visually organize your understanding of the problem.
- Start by brainstorming key concepts and relationships related to the topic.
- Identify any missing connections or gaps in your knowledge by examining the completeness and coherence of the concept map.

2. Low-code Approach:

- How would you approach the problem using low-code tools?

CREATING A PERSONALIZED LOW-CODE ROADMAP

Developing a personalized plan for learning and applying low-code involves tailoring your approach to your unique needs, preferences, and goals [8]. Here are some steps to consider:

1. Assess Your Current Knowledge:

- Reflect on your existing understanding of low-code development. Identify any gaps or areas where you'd like to improve.
- Consider your background in programming, software development, and related technologies.

2. Set Clear Objectives:

- Define what you want to achieve with low-code. Are you aiming to build specific applications, automate processes, or enhance existing systems?
- Establish measurable goals, such as creating a functional prototype within a certain timeframe.

3. Explore the LightCode Platform:

- Look into the features, ease of use, and community support.

4. Learn the Basics:

- Start with the fundamentals. Understand concepts like visual development, drag-and-drop interfaces, and component-based design.
- Explore tutorials, online courses, and documentation provided by the chosen low-code platform.

5. Hands-On Practice:

- Create small projects to practice using low-code tools. Build simple applications, forms, or workflows.
- Experiment with different components and explore their capabilities.

6. Collaborate and Learn from Others:

- Join low-code communities, forums, and social media groups. Engage with fellow developers and share experiences.
- Attend webinars, conferences, and workshops related to low-code development.

7. Customize Your Learning Path:

- Adapt your learning journey based on your interests. If you're more inclined toward web development, focus on web-based low-code tools.
- Explore use cases relevant to your industry or domain.

8. Stay Updated:

- Low-code platforms evolve rapidly. Keep up with new features, best practices, and updates.
- Subscribe to newsletters, follow industry blogs, and participate in relevant discussions.

9. Build Real-World Projects:

- Apply your knowledge by working on concrete projects. Start with small applications and gradually tackle more complex ones.
- Collaborate with colleagues or friends to gain practical experience.

10. Evaluate Your Progress:

- Regularly assess your progress. Are you achieving your objectives? Are there areas where you need additional learning?
- Adapt your plan as needed.

- **Exercise & Quiz:** Developing a personal low-code roadmap.

Now it is time for you to put into practice all you have learned so far. Develop your personal low-code roadmap, using the guide previously provided.

Especially, using a template for creating an individualized low-code learning and development plan.

CONCLUSION

For **understanding your needs according to your expertise**, it is important to reflect on how a personalized approach to low-code learning can transform your digital journey. By taking time to evaluate your current skills, interests, and goals, you position yourself to create meaningful applications and solve real-world challenges with confidence.

Understanding and aligning low-code technology with your personal needs is key to making the most of its potential. This alignment empowers you to overcome skill gaps, leverage your strengths, and innovate within your academic or professional context.

Finally, this module contributes to the **Erasmus+ objective of inclusive digital transformation**, ensuring that learners from all backgrounds have access to practical digital skills. It also sets the stage for the next chapters, which will introduce hands-on tools like the **LightCode Platform** to help you turn your personalized roadmap into action.

REFERENCES

1. Prinz, Niculin and Christopher Rentrop, Melanie Huber. Low-Code Development Platforms – A Literature Review. AIS eLibrary. [Online] 2021.
https://aisel.aisnet.org/amcis2021/adv_info_systems_general_track/adv_info_systems_general_track/2 .
2. Combining Rigorous Requirements Specifications with Low-Code Platforms to Rapid Development Software Business Applications. Pedro Galhardo, Alberto Rodrigues da Silva. Requirements Engineering: Practice and Research, 2022.
3. Challenges of Low-Code/No-Code Software Development: A Literature Review. Karlis Rokis, Marite Kirikova. 2022.
4. Kelly, Andrew. 5 Reasons to Use Low-Code Platforms in Higher Education. www.bplogix.com. [Online] 2022,. <https://www.bplogix.com/blog/5-reasons-to-use-low-code-platforms-in-higher-education> .
5. Lamanna, Charles. 6 people who changed careers with low-code from Power Apps. [Online] 2022. <https://powerapps.microsoft.com/en-us/blog/6-people-who-changed-careers-with-low-code-from-power-apps> .
6. Valle, Madison da. I saw a problem and solved it with low-code. www.forbes.com. [Online] 2021. <https://www.forbes.com/sites/servicenow/2021/08/30/i-saw-a-problem-and-solved-it-with-low-code/> .
7. Wall, Pat. 7 Low-Code platform success stories. www.starterstory.com. [Online] 2024. <https://www.starterstory.com/ideas/low-code-platform/success-stories> .
8. Matthew L. Bernacki, Meghan J. Greene, Nikki G. Lobczowski. A Systematic Review of Research on Personalized Learning: Personalized by Whom, to What, How, and for What Purpose(s)? Educational Psychology Review. 2021, Vol. 33.

Further Exploration References

1. Calo, Emily. The must-have, low-code capabilities for every enterprise. PEGA. [Online] 2021. <https://www.pega.com/insights/articles/must-have-low-code-capabilities-every-enterprise> .
2. Martin, Joel. Using low-code in your SDLC can accelerate delivery by 85%. www.hfsresearch.com. [Online] 2023. <https://www.hfsresearch.com/research/low-code-agile-sdlc> .
3. Usman, Rafiq. Understanding Low-Code or No-Code Adoption in Software Startups: Preliminary Results from a Comparative Case Study. <https://link.springer.com/> . [Online] 2022. https://link.springer.com/chapter/10.1007/978-3-031-21388-5_27 .
4. Duarte Begonha, Andreas Kopper, Thanou Thirakul. Low-code/no-code: A way to transform shadow IT into a next-gen technology asset. www.mackinsey.com. [Online] 2022. <https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/tech-forward/low-code-no-code-a-way-to-transform-shadow-it-into-a-next-gen-technology-asset> .
5. Yajing Luo, Peng Liang, Chong Wang, Mojtaba Shahin, Jing Zhan. Characteristics and Challenges of Low-Code Development: The Practitioners' Perspective. www.arXiv.com. [Online] 2021. <https://arxiv.org/pdf/2107.07482>
6. Dilmegani, Cem. research.aimultiple.com/low-code/. research.aimultiple.com. [Online] 2024. <https://research.aimultiple.com/low-code/> .
7. Chatterjee, Shormistha. Comprehensive Guide to Low-Code Development in 2023. www.browserstack.com. [Online] 2023. <https://www.browserstack.com/guide/low-code-development> .

This document is licensed under the Creative Commons Attribution–NonCommercial 4.0 International License ([CC BY-NC 4.0](https://creativecommons.org/licenses/by-nc/4.0/)).

