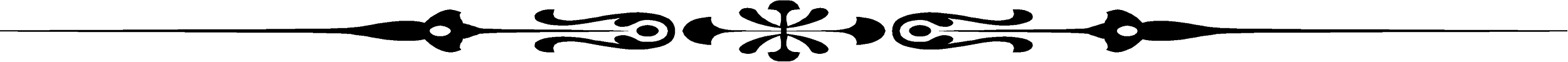
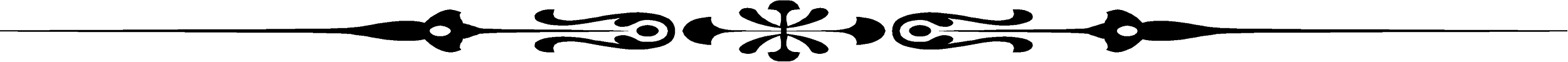
A black and orange square with green squares

Description automatically generated

**ĐẠI HỌC DUY TÂN**

**KHOA CÔNG NGHỆ THÔNG TIN**



**HỆ THỐNG TÍCH HỢP**

**NGHIÊN CỨU TRƯỜNG HỢP III**

**GIẢNG VIÊN: NGUYỄN MINH NHẬT**

**LỚP: SE 445 B**

**NHÓM: 07**

**THÀNH VIÊN**

|  |  |
| --- | --- |
| **Nguyễn Đình Phương** |  |
| **Phạm Anh Quân** |  |
| **Trần Như Thành** |  |
| **Huỳnh Đặng Ngọc Hoàng** |  |

**Đà Nẵng,**

**Contents**

[**I.** **Introduce** 4](#_Toc167704785)

[**II.** **Popular Low-code platforms** 4](#_Toc167704786)

[**2.1** **Overview** 4](#_Toc167704787)

[**2.2** **Comparison** 4](#_Toc167704788)

[**III.** **Interface and Use Cases** 5](#_Toc167704789)

[**3.1** **Interface** 5](#_Toc167704790)

[**3.2** **Use cases** 7](#_Toc167704791)

[**3.3 Display** 10](#_Toc167704792)

[**IV.** **Discussion** 13](#_Toc167704793)

[**4.1 Advantages and Disadvantages:** 13](#_Toc167704794)

[**4.2 When to Use Low-code:** 14](#_Toc167704795)

[**4.3 Challenges and Risks:** 14](#_Toc167704796)

[**4.4 Low-code Development Process vs Traditional Development:** 15](#_Toc167704797)

[**V.** **The Future of Low-Code** 15](#_Toc167704798)

[**VI.** **Conclusion** 17](#_Toc167704799)

[**6.1 Summary:** 17](#_Toc167704800)

[**6.2 Questions and Discussion:** 18](#_Toc167704801)

[**REFERENCES** 19](#_Toc167704802)

1. **Introduce**

Low-code is a modern software development approach that employs user-friendly drag-and-drop interfaces and configuration tools in place of traditional coding. This enables participation in the development process by individuals who may not have extensive programming skills.

1. **Popular Low-code platforms**
   1. **Overview**

The leading Low-code platforms:

OutSystems

Mendix

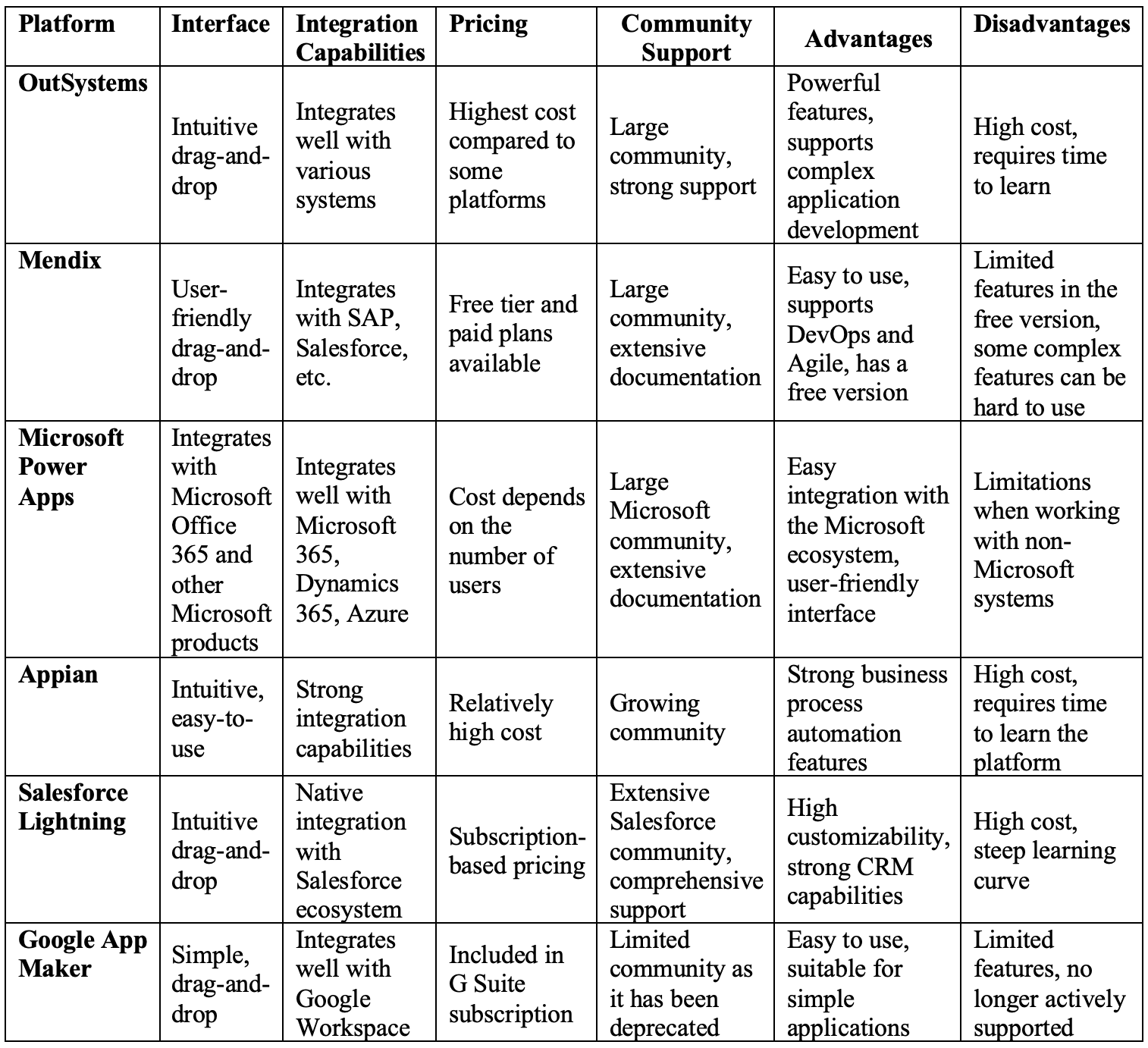
Microsoft Power Apps

Appia

Sales Lightning

Google App Maker

* 1. **Comparison**

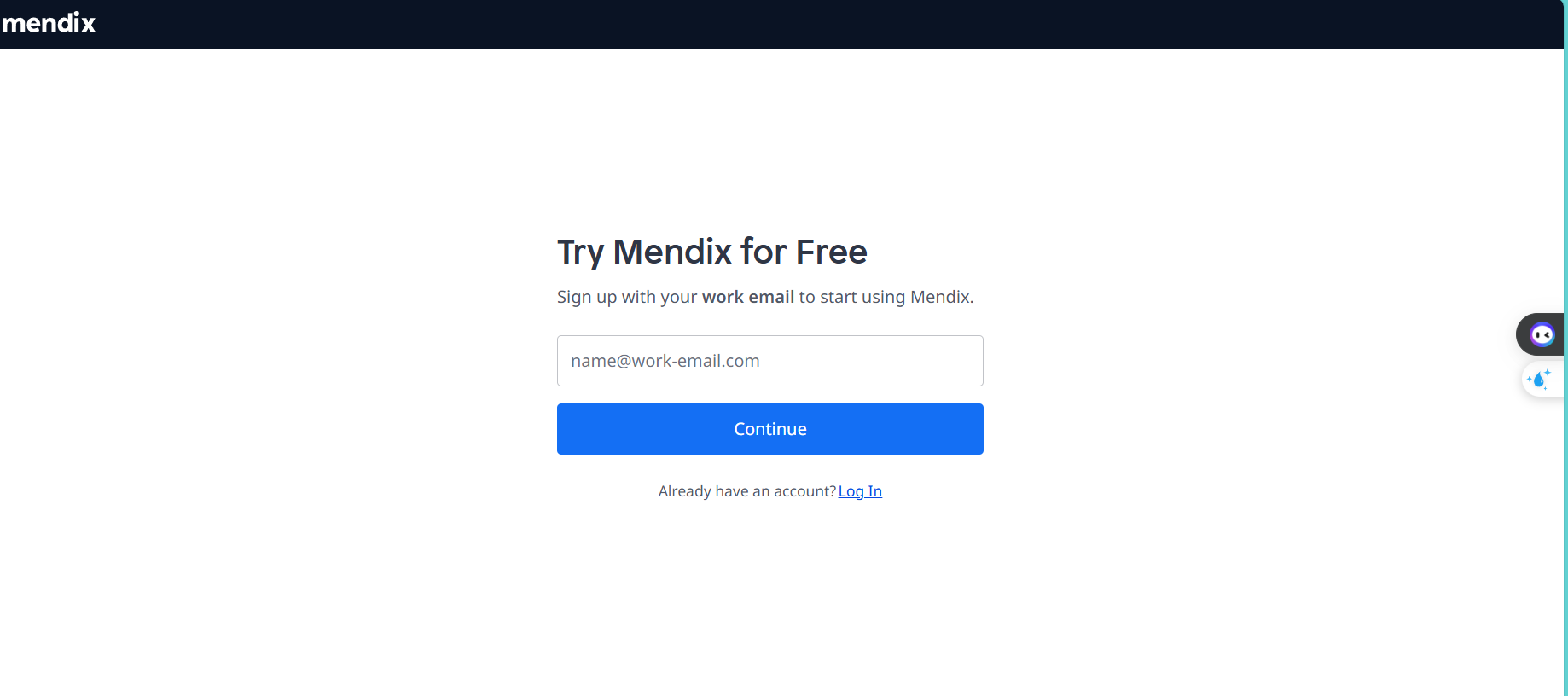


* **summary**:

Each low-code platform like OutSystems, Mendix, Microsoft Power Apps, Appian, Salesforce Lightning, and Google App Maker brings unique advantages and limitations, making them suitable for various business needs and scales. Selecting the ideal platform hinges on considerations such as technical prerequisites, financial constraints, and the specific operational environment of the organization.

1. **Interface and Use Cases**
   1. **Interface** 
      1. **communication :** Mendix studio
      2. **Set up the environment**

**Step 1**: Register: Visit the Mendix website and register for an account

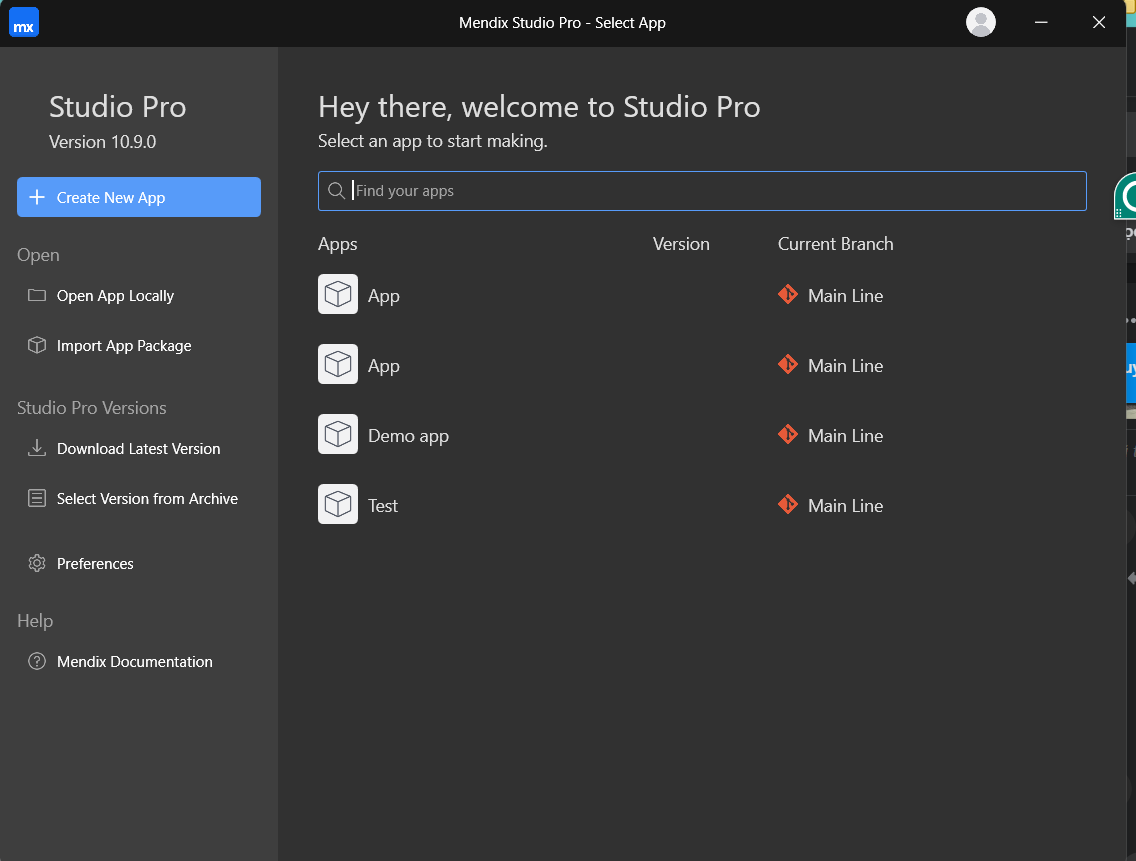


**Step 2**: Download Modeler: Download and install Mendix Studio Pro (Modeler) on your computer.

A screenshot of a computer

Description automatically generated

**Step 3**: Create a new project: Open Mendix Studio Pro and click "New Project". Choose a project template or start from scratch.



* 1. **Use cases**
     1. **Use case login**

**A diagram of a login

Description automatically generated**

* + 1. **Use case display information**

A diagram of a work flow

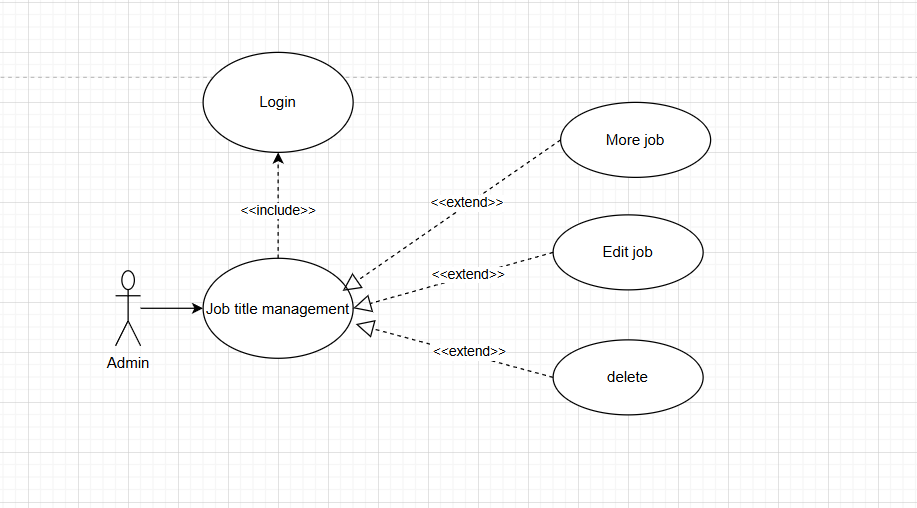
Description automatically generated

* + 1. **Use case manage staff information**

**A diagram of a company with Ice hockey rink in the background

Description automatically generated**

* + 1. **Use case job title management**

****

* + 1. **Use case shift manage**

**A diagram of a diagram

Description automatically generated**

### **3.3 Display**

**3.3.1 Login**

**A computer screen shot

Description automatically generated**

**3.3.2 Home**

**A screenshot of a computer

Description automatically generated**

**3.3.3 Dashboard**

**A screenshot of a computer

Description automatically generated**

**3.3.4 Staff**

**A screenshot of a computer

Description automatically generated**

**3.3.5 Add staff**

**A screenshot of a phone

Description automatically generated**

**3.3.6 Job**

**A screenshot of a computer

Description automatically generated**

**3.3.7 Add job**

**A screenshot of a computer

Description automatically generated**

**3.3.8 WorkShift**

**A screenshot of a computer

Description automatically generated**

**3.3.9 Add workshift**

**A screenshot of a computer

Description automatically generated**

1. **Discussion**

### **4.1 Advantages and Disadvantages:**

* **Advantages of Low-code:**
* **Accelerated Development**: Reduced development time compared to traditional methods.
* **Cost Efficiency**: Savings in development and maintenance costs.
* **Increased Productivity**: Focus on solving business problems rather than technical details.
* **User-Friendly**: Intuitive interfaces and drag-and-drop tools.
* **Flexibility and Customization**: Ability to expand and customize applications as needed.
* **Easy Integration**: Support for integrating with other systems and services.
* **Enhanced Collaboration**: Improved teamwork and communication between departments.
* **Rapid Prototyping**: Quick creation and modification of application prototypes.
* **Security and Compliance**: Ensures application security and regulatory compliance.
* **Support for Digital Transformation**: Facilitates strategies and deployment in digital transformation initiatives.
* **Disadvantages of Low-code:**
* **Limited Customization**: May not allow extensive customization for complex or unique requirements.
* **Scalability Challenges**: Issues scaling applications for large user bases or complex processes.
* **Vendor Dependency**: Reliance on platform vendors for updates, support, and compatibility.
* **Performance Limitations**: Potential performance issues, especially for demanding tasks or real-time operations.
* **Security Risks**: Rapid development may overlook robust security measures, leading to vulnerabilities.
* **Integration Complexity**: Challenges in integrating with legacy systems or specialized APIs.
* **Skill Requirements**: Although accessible, still requires skilled developers for customization and advanced tasks.
* **Vendor Lock-in**: Difficulty switching platforms due to data migration and proprietary technology dependencies.
* **Cost Considerations**: Initial savings can diminish with extensive customization or integration needs.
* **Regulatory Compliance**: Ensuring compliance with industry standards may be more complex with abstracted development processes.

### **4.2 When to Use Low-code:**

* **Rapid Prototyping and MVPs**: When you need to quickly create prototypes or Minimum Viable Products (MVPs) to validate ideas or enter the market swiftly.
* **Resource Constraints**: If there's a shortage of skilled developers or limited IT resources, low-code platforms empower citizen developers and reduce reliance on specialized technical expertise.
* **Standardized Business Processes**: For automating and streamlining standardized workflows and business processes that don't require extensive customization.
* **Cost Efficiency**: When cost reduction is a priority, as low-code development can minimize expenses associated with traditional coding and development cycles.
* **Innovation and Experimentation:** When you want to innovate rapidly and test new ideas without significant upfront investment in time and resources.
* **Agile Development**: In environments where agility and responsiveness to changing business needs are crucial, low-code platforms enable quick adjustments and iterations.
* **Digital Transformation Initiatives**: For accelerating digital transformation by quickly deploying new applications and enhancing existing systems with minimal disruption.
  1. **Criteria for Using Low-code:**
* **Time Constraints:** Ideal for projects requiring rapid delivery, such as Minimum Viable Products (MVPs) or quick prototyping.
* **Limited Developer Resources:** Useful when skilled developers are scarce or to lighten the workload of the development team.
* **Business Process Automation:** Effective for automating workflows and standard processes that do not demand highly customized solutions.
* **Cost Efficiency:** Prioritized for projects aiming to reduce development costs, leveraging the cost-effectiveness of low-code platforms.
* **Innovation and Experimentation:** Suitable for swiftly developing and testing new concepts, facilitating innovation without extensive resource commitments.

### **4.3 Challenges and Risks:**

* **Customization Constraints:** Complex requirements may exceed the capabilities of low-code platforms. **Mitigation:** Select platforms that offer options for custom coding or extendibility through APIs.
* **Security Issues:** Concerns about data security and compliance could arise. **Mitigation:** Ensure the platform adheres to rigorous security practices and meets regulatory standards.
* **Scalability Challenges:** Low-code platforms may struggle to support large-scale applications effectively. **Mitigation:** Evaluate scalability features of the platform and conduct comprehensive testing.
* **Vendor Dependency:** Relying on a single platform can pose risks. **Mitigation:** Choose platforms that support industry standards and facilitate easy data migration.

### **4.4 Low-code Development Process vs Traditional Development:**

* **Low-code Development Steps:**

1. Define Requirements: Collaborate with stakeholders to outline application needs.
2. Design UI: Use intuitive drag-and-drop tools for user interface design.
3. Model Data: Visual creation of entities and their relationships.
4. Build Business Logic: Develop workflows using visual models.
5. Test and Deploy: Utilize built-in testing tools and deploy with ease.
6. Maintain and Update: Regularly update based on feedback and evolving requirements.

* **Traditional Development Steps:**

1. Requirements Gathering: Detailed documentation and analysis of project requirements.
2. System Design: Architectural planning, database schema design, and technology selection.
3. Development: Writing code for user interfaces, business logic, and database interactions.
4. Testing: Comprehensive testing phases including unit, integration, and user acceptance testing.
5. Deployment: Deploying the application to production environments.
6. Maintenance: Continuous updates, bug fixes, and enhancements based on ongoing feedback and changing needs.
7. **The Future of Low-Code**

* Future Trends:
* Increasing Adoption Across Industries: Low-code platforms are being adopted by a wide range of industries, from finance and healthcare to manufacturing and retail. This trend is expected to continue as businesses seek to accelerate their digital transformation efforts and reduce reliance on traditional, more time-consuming development processes.
* Empowerment of Citizen Developers: Low-code platforms empower non-technical users (citizen developers) to create and customize applications. This democratization of app development helps bridge the gap between IT and business units, fostering innovation and rapid problem-solving within organizations.
* **Integration with AI and Machine Learning** The integration of AI and machine learning with low-code platforms will enhance their capabilities, enabling users to build more intelligent and predictive applications. These integrations can automate complex tasks, provide actionable insights, and improve decision-making processes..
* Enhanced Collaboration and Agile Development: Low-code platforms facilitate better collaboration between IT professionals and business stakeholders. This leads to more agile development cycles, where feedback can be quickly incorporated, and applications can be iteratively improved.
* Focus on Security and Compliance: As low-code adoption grows, so does the need for robust security and compliance features. Future low-code platforms will likely offer enhanced security measures, including better access controls, data encryption, and compliance with industry regulations.
* Hybrid and Multi-Cloud Environments: Low-code platforms will increasingly support hybrid and multi-cloud environments, allowing organizations to build applications that can run seamlessly across different cloud providers and on-premises infrastructures. This flexibility is crucial for businesses looking to optimize their IT strategies.
* Expansion of Use Cases: The use cases for low-code platforms are expanding beyond simple application development. Future low-code platforms will be used for more complex workflows, automation, integration tasks, and even for developing enterprise-level applications.
* Evolution of Low-Code Development Skills: As low-code platforms evolve, so will the skills required to use them effectively. Training and certification programs will become more prevalent, helping users maximize the potential of these platforms and keep up with new features and best practices.
* Vendor Ecosystem Growth: The ecosystem of low-code platform vendors will continue to grow, offering a variety of solutions tailored to specific industry needs and use cases. This competition will drive innovation and improvements in the capabilities of low-code tools.
* Economic Efficiency: Low-code development offers significant cost savings by reducing the need for extensive coding and lengthy development cycles. This economic efficiency will be a compelling factor for many organizations, particularly in times of budget constraints.
* IoT (Internet of Things): As IoT devices become more prevalent, low-code platforms are evolving to support the development of applications that can connect and interact with these devices. This simplifies the creation of IoT solutions for monitoring, control, and data analysis.
* Blockchain Integration: Low-code platforms are starting to offer integration with blockchain technologies, enabling the development of decentralized applications (dApps) and smart contracts. This opens new possibilities for secure and transparent transactions in various industries.
* Potential to Transform Software Development:
* Accelerated Development Cycles: Low-code platforms enable rapid application development, significantly reducing the time required to build and deploy software.
* Reduced Development Costs: By minimizing the need for extensive manual coding, low-code platforms lower the overall costs associated with software development.
* Enhanced Collaboration: Low-code platforms facilitate collaboration between IT professionals and business stakeholders through visual development tools and user-friendly interfaces.
* Increased Accessibility: Low-code platforms democratize software development, enabling non-technical users (citizen developers) to create and modify applications.
* Seamless Integration: Modern low-code platforms offer robust integration capabilities, allowing for seamless connectivity with existing systems, databases, and third-party services.
* Agility and Flexibility: Low-code platforms support agile development methodologies, allowing for iterative improvements and quick adaptation to changing requirements.
* Improved User Experience: With pre-built templates and components, low-code platforms enable the creation of user-friendly and consistent interfaces.
* Streamlined Maintenance and Updates: Low-code platforms simplify the process of maintaining and updating applications, often providing automated tools for deployment and version control.
* Scalability: As low-code platforms evolve, they increasingly support scalable solutions that can grow with an organization's needs.
* Focus on Innovation: By handling routine development tasks, low-code platforms free up skilled developers to focus on more complex and innovative projects.

1. **Conclusion**

**6.1 Summary:**

* Accelerated Development: Low-code platforms significantly reduce the time required to develop and deploy applications, allowing businesses to respond quickly to market demands and customer needs.
* Cost Efficiency: By minimizing the need for extensive manual coding, low-code platforms lower development costs, enabling organizations to allocate resources more efficiently and invest in other strategic initiatives.
* Empowerment of Citizen Developers: Low-code democratizes software development, enabling non-technical users to create and modify applications. This broadens the pool of individuals who can contribute to software projects and fosters innovation.
* Enhanced Collaboration: These platforms facilitate better collaboration between IT professionals and business stakeholders, promoting a more inclusive development process that ensures applications meet business requirements and user expectations.
* Seamless Integration: Modern low-code platforms offer robust integration capabilities, ensuring new applications can operate within existing IT infrastructures and enhance overall system functionality.
* **Agility and Flexibility**: Supporting agile development methodologies, low-code platforms allow for iterative improvements and quick adaptation to changing requirements, helping organizations remain agile and responsive.
* Improved User Experience: Pre-built templates and components in low-code platforms enable the creation of user-friendly and consistent interfaces, enhancing end-user satisfaction and adoption rates.
* Streamlined Maintenance and Updates: Low-code platforms simplify application maintenance and updates, reducing the burden on IT departments and ensuring minimal downtime and disruption.
* Scalability: As low-code platforms evolve, they increasingly support scalable solutions that can grow with an organization's needs, ensuring long-term viability of applications.
* Focus on Innovation: By handling routine development tasks, low-code platforms free up skilled developers to focus on more complex and innovative projects, driving technological advancement and strategic advantages.

**6.2 Questions and Discussion:**

* Audience Interaction: Invite the audience to ask questions and discuss any aspects of the presentation. Encourage sharing of thoughts on their experiences with low-code platforms and how they see it fitting into their development processes.

**REFERENCES**

Further Reading: Provide a list of resources for the audience to explore more about low-code platforms and their applications:

* Mendix Official Documentation: Mendix Docs
* OutSystems Learning Portal: OutSystems Training
* Microsoft Power Apps Documentation: Power Apps Docs
* Appian Community: Appian Community