mgminsights

Innovation Implemented.

A12

Frequently Asked Questions (FAQ)

Legal notice

mgm technology partners GmbH
Taunusstr. 23
80807 München
Tel +49 89 / 358 680-0

Jurisdiction and place of performance: Munich

www.mgm-tp.com

About this document

This FAQ answers a number of frequently asked questions about the **Enterprise Low Code Platform A12**. A structured introduction to A12 is provided in the whitepaper "A12 - Low Code for Individual Enterprise Software".

The list of questions will be successively expanded. Do you have further questions? Please feel free to contact us or arrange a demo appointment.

Inhalt

1. Low Code

1.1. What does "Low Code" mean? What is a "Low Code Platform"?

The Buzzword Low Code or Low Code Platform was coined in 2014 by the market research company Forrester. It refers to a range of very different approaches, which stand in the tradition of model-based software development. One common feature is the principle of generating program code based on models. This is exactly what the term low code refers to. It is about getting an executable software with less manually written code. The software development process should be accelerated, become more efficient and be less resource-intensive. (Strictly speaking it should be called low coding... it is not that the finished program consists of less code... only less code was written by hand)

A12 existed as a model-based architectural approach even before the Buzzword Low Code appeared. Nevertheless, today we call A12 an Enterprise Low Code platform. The new term conveys very well the central idea that we can develop software faster and more efficiently through the use of models and code generation, and make it easier to adapt and maintain.

1.2. How do model-driven software development and low code go together?

Low Code is a new term for approaches that have been practiced in a similar form for decades. This includes in particular model-driven software development. To realize applications with less manually written code, a part of the code has to be generated or interpreted. The prerequisite is a previous modeling, typically in a domain-specific language.

1.3. What is an "Enterprise Low Code Platform"? What does mgm understand by that?

There are big differences between the existing low code approaches. Some vendors provide platforms as closed ecosystems where users can click together and publish small apps. This modular principle has the advantage that results are visible very quickly. Training is minimal, hardly any previous knowledge is required. However, it also has the disadvantage that only limited complexity can be mapped. A12, on the other hand, is designed as an open platform that remains very close to a developer's modern toolbox. Depending on the problem, different approaches can be used flexibly.

If we imagine the software development process as a production line, different stations require different degrees of variability. Sometimes prefabricated building blocks that are reused one to one are sufficient. Sometimes we have to parameterize and configure the building blocks to map the necessary variability. If certain building blocks are needed

again and again in many different configurations, we can develop domain-specific languages. This makes even greater variability manageable. But then there are always cases that are highly complex, but do not always come up again. Here an individual implementation is the best way. We assume that projects for business-critical enterprise applications always contain parts that have to be developed individually. This is because in many constellations this is simply the most efficient and sustainable way. Unnecessary abstractions lead to unnecessary complexity and various subsequent problems such as difficult maintainability.

We call an enterprise low code platform a problem-adequate approach that combines the development options outlined above. Low code, where possible and reasonable. Individual development, where necessary.

2. A12 in general

2.1. What is A12?

A12 is an **enterprise low code platform** for the realization of business applications in complex IT landscapes.

A12's **modeling platform** provides tools to quickly create and maintain parts of an application without programming skills. A12's **runtime platform** provides the flexibility to develop low code apps in combination with professional custom software development and system integration which evolve into fully integrated enterprise applications.

2.2. What does "A12" stand for?

A12 stands for "Alliance 2012". Originally, A12 was an initiative of mgm for cross-project collaboration. The commitment to the alliance also marks the starting point for the development of the platform and the focus on model-based software development. Although the platform has gone through several stages of evolution in terms of technology, we have remained true to the name.

2.3. So, who is A12 intended for?

A12 is intended for medium-sized and large companies and public authorities that require individual software. With A12 you can quickly create prototypes and simple applications that can grow into complex, fully integrated enterprise applications..

2.4. What types of applications is A12 designed for?

With A12, highly scalable, secure and robust web applications can be developed. These include underwriting platforms for industrial insurance, portals, form-based systems and specialized applications for the public sector, as well as online stores, marketplaces and integrated solutions for online, branch and mail order business. A12 demonstrates the particular strengths of the model-based approach, especially in business-critical areas where applications have a long service life, but need to be adapted again and again due to technical developments or changing regulatory requirements.

2.5. How does A12 differ from other low code platforms?

A12 combines a low code approach, where business experts can create an application without programming skills, with professional custom software development and system integration.

The focus of A12 is not on easily click-together apps for temporary use. A12 rather provides an answer to the question of how applications can become fully integrated, business-critical enterprise applications in the long term.

A12 is not a Platform as a Service (PaaS) service where simple applications can be clicked together and deployed. It is not a closed ecosystem. As an enterprise low code platform A12 is only used in professional, individual software development projects.

Another special feature is that A12 projects can submit direct requirements to the A12 base - analogous to the requirements for your own project software. This means that the projects are much more involved, they can influence the further development of A12 very directly.

With the Plasma Design System, A12 is also specifically designed to meet the requirements of mature business applications in the UI/UX area.

2.6. What are the advantages of A12?

A central advantage of A12 is the separation of business-related models and technology. Especially business-critical software, which has a long service life, benefits enormously from this. Business content, which is subject to constant change, can be modeled much faster and with less effort in the software. Technical innovations can be realized without having to consider all the business-related content of the application. For example, a new technology can be introduced in the design and realization of the user interface, in persistence or in server processing.

Independent handling of business content

Specialist experts and business analysts can use modeling tools to independently map the domain-oriented core of the software and maintain it over the long term.

- Adaptation of business aspects without programming knowledge
- Fast implementation of business-related changes
- Extensive code generation of modeled aspects
- Detached innovation of technology

Open platform instead of closed ecosystem

A12 is designed as an open system, which allows the greatest possible flexibility for the development as well as the long-term maintenance and further development of the software.

- Low code, individual software development and system integration from one source
- Flexible use of modular runtime components
- Consequent use of Open Source technologies
- Full control over operation - on-premise or in (private) cloud
- Possibility to make requirements directly to the A12 base

Future-proof platform for long-lasting software

The systematic separation of business and technology makes it possible to retain the business core even in the case of technological leaps.

- Detached innovation of technology through model-based approach
- Data First" principle for sustainable domain-oriented modeling
- Careful technology selection and use of industry standards
- Continuous development of the technical foundation

2.7. Does A12 lower the cost of a development project?

Drastic speed advantages can be achieved in software development through the use of low code. The first executable version of an application and the Minimum Viable Product (MVP) of a business application are made available very quickly. How high the time savings and cost advantages are ultimately depends on how intensively the project relies on standard components. The more standard solutions from A12 are used, the less individual effort is required and the higher the cost benefits.

2.8. To what extent does the use of A12 bind me to mgm?

A12 is designed as an open system, which allows the greatest possible flexibility for long-term maintenance and further development of the software. There are no lock-in effects. Customers can develop and maintain a large part of the application - the modeled domain expertise - independently from the beginning. The share of individually written code is significantly lower compared to classic individual software projects. This also simplifies handovers - in case an internal development team or another service provider is to take over further support after project completion.

3. Use of A12

3.1. How is A12 used in actual practice?

A12 is used in individual software development projects. In this case, we speak of an *A12 project*. The goal of an A12 project is to create a secure, high-performance and scalable business application (*A12 application*) and bring it into production quickly.

3.2. What are the parallels and differences between an A12 project and a conventional software development project?

In an A12 project, too, a complete development environment and professional project management are indispensable. However, the use of the A12 platform provides a proven technical basis that already solves many typical challenges of enterprise software. Individual extensions can build on this. Another difference is a new form of division of labor: The low-code approach allows business analysts to model domain-specific content with the help of special tools. In parallel to the work on the source code, there is work on models.

3.3. How can I get A12?

The provision of A12 is always bound to a project in which mgm or an official A12 partner acts as contractor. Please feel free to contact us if you are interested.

3.4. Can my department use A12 independently?

If an A12 application has been created as part of a project, the department is able to adapt the application independently using modeling tools. However, A12 is not an app toolbox that involves clicking the application together and deploying it at the push of a button. The use of A12 outside of a software development project is not intended in this form.

3.5. How does the cooperation between client and contractor work in an A12 project?

In principle, cooperation between the customer and the contractor is an important success factor in A12 projects. With regard to the division of tasks through the use of the Low Code platform, the following constellations are common:

COLLABORATION MODEL	DESCRIPTION
Contractor assumes overall responsibility	The contractor is responsible for the entire application. It takes on the requirements of the customer within the requirements analysis and in ongoing iterations. The Contractor's project team is responsible for creating the corresponding A12 models and for all development, test and release activities.
Task sharing: modeling / development	The customer is responsible for the domain-oriented modeling. The contractor is responsible for development and technology. The prerequisite for this model is that the business experts/business analysts of the customer are trained in A12 modeling. Since the models represent the business domain, this division of tasks makes a lot of sense. The contractor provides support for modeling questions and is responsible for the entire development and technical tasks.
Cooperative development	If the client has its own development team that masters the A12 technology stack, it is possible to involve this team in the development of the A12 application. The prerequisite is the further training of the team members with training courses on development with the A12 platform. From the client's point of view, the advantage of this model is greater independence from service providers and the possibility of developing future additions to the application independently.

4. Modeling

4.1. What modeling tools are available?

The Simple Model Editor (SME) bundles all modeling functionalities of A12. The web-based tool allows easy management of all A12 models - both in the tree view of a workspace explorer and visually through the integrated Model Graph Diagram Editor. For editing the individual model types, the tool contains specialized editing modules.

4.2. Which A12 models are available?

KATEGORIE	BEZEICHNUNG	BESCHREIBUNG
Data Model	Document Model	A12 document models contain field definitions and associated validation rules in a hierarchy of groups. Validation rules range from simple constraints - e.g., the definition of mandatory fields - to complex patterns and conditions across multiple fields.
	Relationship Model	Relationship models describe links between documents. They model the relationship properties and constraints .
UI Model	Form Model	Form models define the structures and contents of online forms. A12 forms consist of common UI elements such as input fields, buttons, labels, checkboxes, etc. The modeling tools provide powerful ways to organize these elements.
	Overview Model	Overview models offer various possibilities for tabular presentation of data.
	Tree Model	Tree models allow data structures to be displayed and edited hierarchically.
Workflow	BPMN 2.0	A12 supports modeling of business processes in the BPMN (Business Process Model and Notation) standard. BPMN models interact seamlessly with A12 models.
App Model	App Model	An app model defines the framework of the application and acts as a kind of container for all other models.
Output Model	Print Model	The Print Model can be used to create print templates for the generation of accessible PDFs.

4.3. What can I model with A12 and what not?

The current modeling scope includes the entire domain-related content and parts of the user interface. The decision as to what can be modeled in A12 is based on the added value that the modeling capability would bring. In complex business applications, there are always aspects that can be developed individually faster, more efficiently and more sustainably.

With the data models and the rule language for validations and calculations, very complex technical relationships can be mapped. Trained business experts and business analysts are thus able to use the modeling tools to independently define the business aspects of an application - without being tied to developers. Only in exceptional cases - such as calculations with very complex formulas - can it be more convenient to map the calculations directly in the code and not via the modeling tools.

The modeling of the user interface is currently limited to those areas in which model-driven components are used.

MODELABLE	INDIVIDUALLY IMPLEMENTABLE
Domain expertise - data models with validation rules and calculations	complex algorithms (e.g. generic premium calculator in the insurance environment)
Frame of an application including placement of model-driven engines	placement of widgets of the Plasma design system
Forms, including repeatable structures	definition or adaptation of design elements
Tabular overviews of data sets	
Tree-like overviews of data sets	
Relationships between different model-driven components	
Accessible PDF documents	

4.4. Is there a training program from mgm for the modeling tools?

Yes, the Business Professional Services Team of A12 offers training in the use of the modeling tools. They are primarily aimed at business analysts, who take over parts of the modeling in projects. In addition to face-to-face trainings, which can be individually designed according to previous knowledge, an e-learning module is available for the introduction to data modeling.

4.5. Can A12 models be reused?

Yes, reuse is possible for all model types. It is particularly common practice with the A12 data models. Data models can be built up modularly. For example, lower-level submodels can be reused in several other models. Furthermore, special type definitions - for example for central country lists and legal forms - can be created that can be used in several models. In this way, cross-model aspects can be defined in a single place to avoid duplication and any potential inconsistencies.

5. Security

5.1. How is the security of the A12 platform ensured?

A12 follows the principle Security by Design. Security requirements are taken into account from the very beginning to prevent potential weaknesses. Security experts accompany all phases of development - from early requirements to architectural decisions and acceptance tests. In addition, the enterprise low code platform is continuously tested intensively with the help of the security test platform mgm ATLAS.

5.2. How are A12 projects secured?

The security measures required in each case must be defined individually according to the requirements of the project. Basic guidelines, best practices and recommendations for the secure use of A12 are summarized in the *A12 Security Guidelines*. They are available on the GetA12 documentation platform and outline what a secure standard configuration looks like based on the A12 Full-Stack Project Template. In addition to securing service endpoints and the approach for a logging strategy that takes into account the requirements of the General Data Protection Regulation, the documentation includes tips for a secure configuration of Keycloak as an identity provider and recommendations for the use of security headers.

5.3. What is mgm ATLAS?

ATLAS is a security toolset developed by mgm that integrates a number of tools such as OWASP Dependency Check, ZAP and sqlmap. It enables automated security tests and provides consolidated reporting. Among other things, ATLAS checks for known vulnerabilities in third-party components, detects configuration issues such as missing HTTP security headers, and tests how robust APIs are against attacks such as injection attacks. With the help of ATLAS, the A12 platform is continuously and automatically scanned for vulnerabilities. We also recommend the use of ATLAS in A12 projects.

5.4. My application must meet the security requirements of the "IT-Grundschutz-Bausteine" according to the German BSI. Is this possible with A12?

Yes, the software development lifecycle (SDLC) of A12 already takes into account the requirements of the CON.8 software development module. The relevant requirements must be updated in the development phase of the application on the basis of A12. Finally, there are specifications for operation (OPS module) that must be taken into account.

mgm already has experience in the successful implementation of the strict specifications of the “IT-Grundschutz-Bausteine” - both as a supplier and as an operator.

6. UI / UX

6.1. What is A12 Plasma?

A12 Plasma is a design system that mgm has developed specifically for business applications. It consists of UI/UX components, usage patterns and design guidelines that allow for consistent, efficient and attractive user interfaces.

The A12 Widget Showcase contains examples of all available plasma components.

In contrast to pure design languages such as Material Design, Plasma also takes into account the extended functionality typically required by business applications. This includes in particular aspects such as scalability and the handling of a high information density.

6.2. How does the UI modeling work in A12?

A12 uses special UI models for the design of the user interface. They, too, are guided by the idea of separating technology from content. UI models enable an abstract representation of the interaction structure - for example the structure of a form - without being hard-wired to a specific technical implementation. This has the advantage that the technical display details and the design can be developed separately from the UI models. This makes it much easier to implement a consistent and accessible user interface. The Plasma Design System is used for the actual realization.

6.3. Does A12 support accessibility?

Yes, the A12 platform is designed for building accessible web applications. Numerous UI components - including the model-driven engines for forms and overviews - are accessible out-of-the-box. However, in the project practice of individual software development, there are always additional aspects to consider. There are specific requirements that a Low Code platform per se cannot cover. For this purpose, the A12 team offers projects practical assistance in the form of a regularly updated guide. It contains, for example, background information on accessibility certification, design specifications, and requirements for modeling and development.

6.4. Can you build mobile applications with A12?

Yes, A12 is designed for developing responsive and device-independent web applications. They provide a first-class user experience across mobile devices such as tablets and smartphones.

6.5. What themes does A12 offer?

A12 provides four officially supported themes to choose from (*Default*, *Compact*, *Flat* and *Flat-Compact*), which are specifically designed to meet the requirements of business applications. *Default* and *Compact* themes follow a structure-focused design. *Flat* and *Flat-Compact* offer a content-focused design with more discreet navigation elements. In addition to these standard themes, extensions can also be used to create your own individual themes.

7. Operations

7.1. What options does mgm offer for the operation of A12 applications?

For business critical software it is essential that sensitive data is stored in a trustworthy, secure environment and that smooth operation is ensured. The importance of maintaining full control over operations is something we see time and again with our customers in the e-commerce sector, for example. At times of high demand, such as during the Christmas business, the systems run at maximum load for a long time without downtime. To achieve this, the software must be both performant and scalable. On the other hand, it also requires sole control over the underlying infrastructure and the release versions used.

We offer the following options for the deployment of A12:

- On-premise operation in the company's own data center
- Operation in the private cloud of mgm, hosted in a data center in Germany
- Cloud operation with any cloud provider

7.2. Does A12 support Kubernetes?

Yes, A12 applications are designed to be deployed on Kubernetes clusters by default. Based on experience from several large software projects, we have selected a set of tools from the Kubernetes ecosystem that we recommend as the default stack. In principle, however, A12 applications can be operated with different technology stacks - depending on the specifications of the respective hoster.

7.3. Will A12 run on OpenShift clusters?

Although A12 applications do not run out-of-the-box on Red Hat OpenShift clusters, they can be run in such environments by modifying the configuration.

8. Technology

8.1. Can parts of the A12 platform be used separately?

Yes, A12 is modular and divided into different components. The cut of the A12 components is technically motivated. Each component has a clear scope and clear interfaces to the outside. The components can be used flexibly - even individually. For example, you can use the client and write the server yourself.

8.2. Which A12 components are available?

COMPONENT	INCLUDED IN PLATFORM LICENSE	DESCRIPTION
Client	Yes	Model-driven, client-side runtime component. Implements the UI/UX concept of the Plasma Design System and supports desktop, tablet and smartpho-ne. Main tasks are the orchestration of other UI components, especially the A12 engines, data retrieval and state management.
Engines	Yes	Model-driven UI components. Engines interpret data and UI models. They are based on the Plasma UI/UX concepts and use the widgets for rendering.
Widgets	Yes	Widget Library, based on Plasma UI/UX concepts. See also A12 Widget Showcase
Kernel	Yes	Bundles everything for the creation and processing of document models: modeling tools, language for validations and calculations, client- and server-side runtime components, Java and Typescript API.
Data Services	Yes	API for managing models and data. It also contains routines for client/server communication, validation, persistence and indexing.
User Management, Authentication and Authorization	Yes	Bundles solutions around authentication (Keycloak, OAuth 2.0, SAML, LDAP), authorization (Spring Security, RBAC, ABAC, custom logic) and user management .
Workflows	Yes	Integration of Business Process Model and Notation (BPMN) in A12; enables graphical modeling of server-side workflows and their execution
Simple Model Editor	Yes	Modeling tool for business analysts
Installer	Yes	Provides all current and compatible A12 products and tools in a pre-configured package for local installation - allowing business analysts to access a modeling and demo environment
ANTLR 4 Code Editor	Yes	ANTLR (Another Tool for Language Recognition) is a well-established parser generator used, among other

		things, for building domain-specific languages. The component provides a code editor that can be integrated into A12 applications for editing such languages - including syntax validation and highlighting as well as auto-completion.
Rocket.Chat Integration	Yes	Enables the integration of a live chat functionality into A12 applications - for example, to allow customers to chat with service staff. Uses the open source chat platform Rocket.Chat in the background.
Chatbot	Yes	The component can be used to bring chatbots into A12 applications. It uses the Rasa chatbot development platform. Can be optionally combined with the A12 Rocket.Chat integration.
Print Engine	Yes	Lets you generate accessible PDFs in A12 business applications. The Print Model Editor enables business analysts and subject matter experts to design the layout, design and content of PDFs. The Print Engine populates these templates of a print model at runtime with values from data fields of an underlying A12 document model.
Data Distribution	No	Transport layer for the secure and fast synchronization of data. The technical service is designed to distribute data between servers and clients and to propagate changes - especially in scenarios where clients are temporarily offline.
Notification Center	No	Bundles notifications to users in one central location - e.g. info on new tasks, news, appointments and reminders. Provides predefined notification types such as reminders and workflow events. With the help of an API, your own individual notification types can be added.

8.3. Where can I find the source code of A12?

The source code of A12 is currently only available to customers and partners in selected large projects.

8.4. On which specific technologies is A12 based?

The separation of domain knowledge and technology allows the technologies used to be exchanged as required. Currently the technology stack of A12 is composed as follows::

A12 PRODUCT	TECHNOLOGY	DESCRIPTION
Kernel	Java	
	Typescript	
	Groovy	
	Antlr	Parser generator

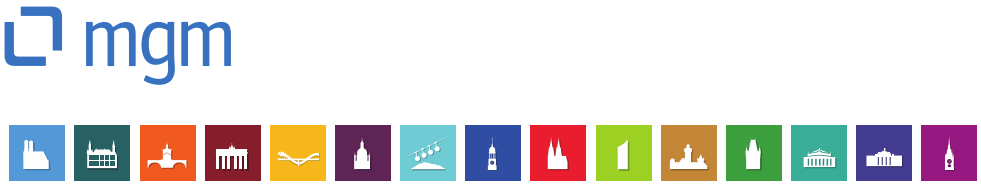
	StringTemplates	Template Engine
	JAXB	Mapping Java objects to XML
	Jackson	JSON processor for Java
Widgets	Typescript	
	React	Building UIs
	Styled components	CSS styling
	Recharts	Chart library
	DraftJS	Rich text editor
	React-Dnd	Drag and drop handling
	React-virtualized	Rendering partial data into DOM
	Redux	State management
UAA	Typescript	
	Redux	State management
	oidc-client-js	OpenIdConnect authentication protocol
	Java	
	Spring	Application framework for the Java platform
	Spring Boot	Auto configuration for Spring application
	Spring-security	Spring security approach for Authorization (SpEL - Spring Expression)
	KeyCloak	identity and access management
	OAuth2/OpenID	protocol for authentication
	SAML	protocol for authentication
	LDAP	protocol for accessing and maintaining distributed directory information services over an IP network
Services	Java	
	Apache solr	Search index
	WildFly	Application server
	Apache Tomcat	Application server
	Eclipse Jetty	Application server
	PostgreSQL	Database
	Oracle	Database
	H2	Local In-Memory-DB
	Spring Security	authentication, authorization
	Spring Boot	Auto configuration for Spring application
	NodeJS	Java runtime environment
	Typescript API	
Workflows	Kotlin	
	Spring	Application framework for the Java platform
	Spring Boot	Auto configuration for Spring application

	Camunda	Platform for BPMN workflow and DMN decision automation
	Typescript	Frontend
	React	Building UIs
	Webpack	JavaScript module bundler
	NPM	package manager for JavaScript
Overview Engine	Typescript	
	React	Building UIs
	Stylus	CSS preprocessor
	Recharts	Chart library
	DraftJS	Rich text editor
	React-Dnd	Drag and drop handling
	React-virtualized	Rendering partial data into DOM
	Redux	State management
Form Engine	TypeScript	
	JavaScript	
	TSLint	Analysing Typescript
	NodeJS	Java runtime environment
	NPM	package manager for JavaScript
	Lerna	Managing multi-package repositories
	Webpack	JavaScript module bundler
	React	Building UIs
	Redux	State management
	Marked	Markdown in expression language
	Jison	Expression language
	moment.js	JavaScript wrapper for the date object
Tree Engine	Typescript	
	React	Building UIs
	Stylus	CSS preprocessor
	Recharts	Chart library
	DraftJS	Rich text editor
	React-Dnd	Drag and drop handling
	React-virtualized	Rendering partial data into DOM
	Redux	State management
Chat Solution	A12 BAP Client	frontend
	A12 Widgets	frontend
	Rocket.Chat	Web chat platform
	NodeJS	Java runtime environment
	MongoDB	Data persistence

Chatbot	Python	
	Rasa	Chatbot development framework
	Tensor-flow	Machine learning/differentiable programming framework
	Scikit-learn	Machine learning library
	Flask	Web framework
Client	Typescript	
	JavaScript	
	TSLint	Analysing Typescript
	NodeJS	Java runtime environment
	NPM	package manager for JavaScript
	Lerna	Managing multi-package repositories
	Webpack	JavaScript module bundler
	React	Building UIs
	Redux	State management
	Inversify	Configuration injection
Data Modeler	Java	
	Tycho	Building Eclilpse plugins
	RCP	Building Eclilpse plugins
	SWT	Widget toolkit for Java
	JFace	UI toolkit
	Jackson	JSON processor for Java
	JSONSchema	Validating the structure of json data
	Slf4J	simple facade or abstraction for various logging frameworks
	LOGBack	logging framework for Java applications
UI Designer	Java	
	Tycho	Building Eclipse plugins
	RCP	Building Eclipse plugins
	SWT	Widget toolkit for Java
	JFace	UI toolkit
	Jackson	JSON processor for Java
	JSONSchema	Validating the structure of json data
	Slf4J	simple facade or abstraction for various logging frameworks
	LOGBack	logging framework for Java applications
Simple Model Editor	A12	Front end
	Typescript	
	React	Building UIs
	Redux	State management

	Redux Saga	library used to handle side effects in Redux
A12 Installer	Typescript	
	React	Building UIs
	Redux	State management
	Redux Saga	library used to handle side effects in Redux
	Spring Boot	Auto configuration for Spring application
	H2 Database	Local In-Memory-DB
	Electron	Software framework to develop desktop GUI applications using web technologies
Plasma Design	Adobe illustrator	Creating graphical user interfaces
	Adobe XD	Creating screens and lo-fi prototypes
	Azure	Creating hi-fi prototypes
	PUG	Template engine – create reusable HTML
	BEM	Creating extendable and reusable CSS
Documentation	Asciidoc	User documentation
	Typedoc	Generating API documentation for TypeScript
	Javadoc	Generating API documentation for Java
QA, Testing & Security	Enzyme	Unit tests
	Cypress	Integration tests
	Testcontainers	Integration/system tests based on Docker containers
	JUnit 5	testing framework for java applications
	MockK	For Kotlin
	H2	Local In-Memory-DB
	QFS-Test-Suite	Automated surface tests
	PerfLoad	Load testing
	Selenium	Browser automation
	Mocha	JavaScript test framework
	TestCafe	Automating end-to-end web testing
	Sonarqube	Continuous inspection of code quality
	OWASP Dependency Check	Scanning for vulnerabilities
	TestRail	Managing and tracking testing
	JAX-RS	Integration tests
	jMeter	Functional behavior and performance tests
	TestNG	Unit, functional, end-to-end, integration tests
	Python	Orchestrating Security Test Suite
	Docker	Running Security Test Suite
	Sqlite, MariaDB	Persistent Storage for Licenses, Credentials, Configuration

	OWASP ZAP	Dynamic Application Security Testing
	Postman/Newman	REST client for API Testing
	OWASP DefectDojo	Security Reporting and Monitoring
	Xanitizer	Static Application Security Testing
	Chai	Assertion library for Node
	NYC	Test coverage reporting
	NPM audit	Security review of project's dependency tree
	Hamcrest	creating customized assertion matchers
Runtime	Docker/Docker-compose	defining and running multi-container Docker applications
	Kubernetes	managing containerized workloads and services
	Prometheus	systems monitoring and alerting toolkit
	Grafana	analytics & monitoring
	ELK (Elastic, Logstash, Kibana)	log management
	Ansible	Automating configuration management & application deployment
Development-Infrastructure	Jenkins	Automation of builds and deployment
	Artifactory	Managing code repositories
	GIT	Version control
	Bitbucket	Code Collaboration & Version Control
	Gradle	Build automation
	Maven	Build automation
	Webpack	JavaScript module bundler
	NPM	package manager for JavaScript



mgm technology partners
www.mgm-tp.com

mgm consulting partners
www.mgm-cp.com

mgm security partners
www.mgm-sp.com

mgm integration partners
www.mgm-ip.com