Project 1: Cloud-Based Image Recognition Service on Azure (100% of the mark)

Description: In this project, students will create a cloud-based image recognition service on Azure using Azure Cognitive Services and Azure Functions. The service will accept an image file as input and use Azure Cognitive Services' Computer Vision API to analyze the image and extract information such as objects, faces, and text. The service will then return the results in a JSON format.

Tasks High level:

1. Set up an Azure account and create a new Azure Function App.
2. Create a new function in the Azure Function App to handle image recognition requests.
3. Integrate the Computer Vision API from Azure Cognitive Services into the function to analyze the image.
4. Configure the function to return the analysis results in a JSON format.
5. Test the image recognition service by uploading various images and verifying the analysis results.

Optional Tasks High level:

1. Add authentication and authorization to the service using Azure Active Directory.
2. Store the analyzed image information in Azure Blob Storage or Azure Table Storage.
3. Create a simple web application that allows users to upload images and view the analysis results.

Skills Needed:

1. Understanding of cloud computing concepts and Azure services.
2. Experience with Azure Functions and Azure Cognitive Services.
3. Knowledge of programming languages such as C# or JavaScript.
4. Familiarity with RESTful API design and JSON data format.

Overall, this project will provide students with practical experience working with cloud-based services and API integration on Azure. It will also give them a chance to develop their programming skills and work with a real-world use case for image recognition.

Project 2: Global Web Service on Azure (100% of the mark)

Description: In this project, students will create a web service on Azure that can be accessed from anywhere in the world. The service will provide a simple API for users to retrieve information about a given city, such as its population, weather, and tourist attractions. The service will be deployed to multiple regions on Azure for global distribution and scalability.

Tasks High level:

1. Set up an Azure account and create a new Azure App Service.
2. Create a RESTful API for retrieving city information using a programming language such as C# or Node.js.
3. Integrate external APIs such as OpenWeatherMap and Google Maps to retrieve weather and tourist information for a given city.
4. Deploy the web service to multiple regions on Azure for global distribution using Azure Traffic Manager or Azure CDN.
5. Test the service by sending API requests from different regions of the world and verifying the response time.

Optional Tasks High level:

1. Add authentication and authorization to the service using Azure Active Directory or Azure App Service Authentication.
2. Store the city information in a distributed database such as Azure Cosmos DB for scalability and availability.
3. Create a simple web application or mobile app that consumes the API and displays the city information.

Skills Needed:

1. Understanding of web services and RESTful API design.
2. Experience with programming languages such as C# or Node.js.
3. Familiarity with external APIs and data formats such as JSON and XML.
4. Knowledge of Azure services such as Azure App Service, Azure Traffic Manager, and Azure Cosmos DB.

Overall, this project will provide students with hands-on experience in designing and deploying a scalable and globally accessible web service on Azure. It will also give them the opportunity to work with external APIs and learn how to integrate them into their own applications.

Project 3: Global City Information Web Service on Azure (100% of the Mark)

Description: In this project, students will create a global city information web service on Azure that allows users to retrieve information about cities around the world. The service will provide a simple web-based interface for users to search for a city and display its population, weather, and tourist attractions. The service will be deployed to multiple regions on Azure for global distribution and accessibility.

Tasks High level:

1. Set up an Azure account and create a new Azure Static Web App.
2. Use a low-code platform such as Microsoft Power Apps or Google AppSheet to create a simple web-based interface for users to search for a city and display its information.
3. Integrate external APIs such as OpenWeatherMap and Google Maps to retrieve weather and tourist information for a given city.
4. Deploy the web service to multiple regions on Azure for global distribution using Azure Traffic Manager or Azure CDN.
5. Test the service by accessing it from different regions of the world and verifying the response time.

Optional Tasks High level:

1. Add authentication and authorization to the service using Azure Active Directory or Azure App Service Authentication.
2. Store the city information in a distributed database such as Azure Cosmos DB for scalability and availability.

Skills Needed:

1. Understanding of web services and API integration.
2. Familiarity with low-code platforms such as Microsoft Power Apps or Google AppSheet.
3. Knowledge of external APIs and data formats such as JSON and XML.
4. Understanding of Azure services such as Azure Static Web Apps, Azure Traffic Manager, and Azure Cosmos DB.

Overall, this project will provide students with a low-code approach to creating a scalable and globally accessible web service on Azure. It will also give them the opportunity to work with external APIs and learn how to integrate them into their own applications.

Project 4: Globally Distributed Static Website on Azure (70% of the mark)

Description: In this project, students will create a globally distributed static website on Azure using Azure Blob Storage and Azure CDN. The website will provide information about a specific topic and will be available in multiple regions around the world for fast and reliable access.

Tasks High level:

1. Set up an Azure account and create a new Azure Storage Account.
2. Create a static website using a website builder tool such as Wix or Squarespace or using static site generators such as Hugo or Jekyll.
3. Upload the static website files to Azure Blob Storage and configure the storage account to host a static website.
4. Set up an Azure CDN profile and endpoint to serve the website content from the nearest CDN edge node to the user.
5. Test the website by accessing it from different regions of the world and verifying the response time.

Optional Tasks High level:

1. Set up Azure Functions to perform serverless computations on the website data, such as analytics or image processing.
2. Set up Azure Application Gateway to provide secure access to the website with SSL termination and WAF protection.
3. Configure Azure Traffic Manager to distribute traffic across multiple Azure regions for high availability.

Skills Needed:

1. Understanding of web technologies and website development.
2. Familiarity with website builders or static site generators.
3. Knowledge of Azure services such as Azure Blob Storage, Azure CDN, Azure Functions, and Azure Traffic Manager.

Overall, this project will provide students with practical experience in designing and deploying a globally distributed static website on Azure using infrastructure services and web services. It will also give them the opportunity to work with serverless computing and explore different Azure services for web applications.

Project 5: Predictive Maintenance Web Service on Azure (100% of the mark)

Description: In this project, students will create a predictive maintenance web service on Azure that predicts the likelihood of equipment failure based on historical data. The web service will be hosted on Azure Virtual Machines and will use Azure Machine Learning to train and deploy a predictive model. The web service will be accessible globally and will provide real-time equipment for health monitoring.

Tasks High level:

1. Set up an Azure account and create a new Azure Virtual Machine.
2. Install and configure web server software such as Apache or IIS on the virtual machine.
3. Create a web application using a web framework such as Flask or Django that allows users to input equipment data and retrieve predictions.
4. Collect historical equipment data from a dataset or a simulated environment and use Azure Machine Learning to train a predictive model.
5. Deploy the trained model as an Azure Machine Learning web service on the virtual machine.
6. Integrate the web service into the web application and provide real-time equipment health monitoring.
7. Deploy the web service to multiple regions on Azure for global accessibility using Azure Traffic Manager or Azure CDN.
8. Test the web service by accessing it from different regions of the world and verifying the response time.

Optional Tasks High level:

1. Use Azure Stream Analytics to collect and analyze real-time equipment data for predictive maintenance.
2. Use Azure Functions to trigger notifications or automate equipment maintenance based on the predictions.
3. Use Azure Kubernetes Service to manage and scale the web application and machine learning workloads.

Skills Needed:

1. Understanding of web technologies and web application development.
2. Familiarity with web frameworks such as Flask or Django.
3. Knowledge of machine learning concepts and tools such as Azure Machine Learning and Python libraries like scikit-learn.
4. Understanding of Azure infrastructure services such as Azure Virtual Machines, Azure Traffic Manager, and Azure Kubernetes Service.

Overall, this project will provide students with hands-on experience in building a predictive maintenance web service on Azure using infrastructure services, web services, and machine learning. It will also give them the opportunity to explore different Azure services and tools for building intelligent applications.

Project 6: Cloud-Based E-Commerce Website on Azure (100% of the mark)

Description: In this project, students will create a cloud-based e-commerce website on Azure that allows users to browse and purchase products online. The website will use Azure infrastructure services such as Virtual Machines and Azure SQL Database to host the web application and store product and user data. The website will be accessible globally and provide real-time inventory management and order processing.

Tasks High level:

1. Set up an Azure account and create a new Azure Virtual Machine.
2. Install and configure web server software such as Apache or IIS on the virtual machine.
3. Create a web application using a web framework such as Flask or Django that allows users to browse and purchase products online.
4. Create an Azure SQL Database to store product and user data and configure the web application to use the database.
5. Implement real-time inventory management by using Azure Service Bus or Azure Event Grid to update inventory data as orders are processed.
6. Use Azure Functions to trigger notifications or automate order processing based on the website activity.
7. Deploy the web application to multiple regions on Azure for global accessibility using Azure Traffic Manager or Azure CDN.
8. Test the website by accessing it from different regions of the world and verifying the response time.

Optional Tasks High level:

1. Use Azure Machine Learning to provide product recommendations based on user behavior.
2. Use Azure Application Gateway to provide secure access to the website with SSL termination and WAF protection.
3. Use Azure Kubernetes Service to manage and scale the web application.

Skills Needed:

1. Understanding of web technologies and web application development.
2. Familiarity with web frameworks such as Flask or Django.
3. Knowledge of database concepts and tools such as Azure SQL Database and SQL queries.
4. Understanding of Azure infrastructure services such as Azure Virtual Machines, Azure Traffic Manager, and Azure Kubernetes Service.

Overall, this project will provide students with practical experience in building a cloud-based e-commerce website on Azure using infrastructure services, web services, and databases. It will also give them the opportunity to explore different Azure services and tools for building scalable and reliable applications.

Project 7: Serverless Web Application with API Gateway on Azure (90% of the mark)

Description: In this project, students will create a serverless web application on Azure that utilizes Azure Functions and API Gateway. The web application will allow users to upload and download files, and will use Azure Functions to process and store data. The API Gateway will be used to provide secure access to the web application and to manage API requests and responses.

Tasks High level:

1. Set up an Azure account and create a new Azure Function App.
2. Create an Azure Blob Storage account to store uploaded files and configure the Azure Function App to use it.
3. Implement Azure Functions to process file uploads and downloads and to store data in the Blob Storage account.
4. Use Azure API Management to create an API Gateway and configure it to route API requests to the Azure Function App.
5. Implement security measures such as authentication and authorization on the API Gateway to ensure secure access to the web application.
6. Deploy the web application to multiple regions on Azure for global accessibility using Azure Traffic Manager or Azure CDN.
7. Test the web application by uploading and downloading files from different regions of the world and verifying the response time.

Optional Tasks:

1. Use Azure Logic Apps to create automated workflows based on the data stored in Blob Storage.
2. Use Azure Event Grid to trigger Azure Functions based on events such as file uploads or updates.
3. Use Azure Active Directory to implement single sign-on (SSO) for the web application.

Skills Needed:

1. Understanding of web technologies and web application development.
2. Familiarity with Azure Functions and API Gateway.
3. Knowledge of Azure Blob Storage and Azure Logic Apps.
4. Understanding of Azure infrastructure services such as Azure Traffic Manager and Azure CDN.

Overall, this project will provide students with practical experience in building a serverless web application on Azure using web services, functions, and API Gateway. It will also give them the opportunity to explore different Azure services and tools for building scalable and secure applications.

Project 8: Deploying a Containerized Web Application on Azure (100% of the mark)

Description: In this project, students will create a containerized web application using Docker and deploy it on Azure using Azure Container Registry and Azure Kubernetes Service. The web application will use a containerized version of a popular open-source web application such as WordPress or Drupal, and will be accessible globally through Azure Kubernetes Service.

Tasks high level:

1. Set up an Azure account and create a new Azure Container Registry.
2. Create a Dockerfile to build a container image of the chosen web application and push it to the Azure Container Registry.
3. Create an Azure Kubernetes Service cluster and configure it to use the Azure Container Registry to pull container images.
4. Deploy the containerized web application to the Azure Kubernetes Service cluster.
5. Use Azure Traffic Manager or Azure CDN to provide global accessibility to the web application.
6. Implement scalability and high availability features for the web application using Azure Kubernetes Service.

Optional Tasks:

1. Use Azure Monitor to monitor the performance and health of the containerized web application.
2. Use Azure DevOps to automate the deployment process and manage the container lifecycle.
3. Use Azure Service Mesh to manage and secure communication between microservices running in the containerized web application.

Skills Needed:

1. Understanding of containerization and Docker.
2. Knowledge of Kubernetes and container orchestration.
3. Familiarity with Azure Container Registry and Azure Kubernetes Service.
4. Understanding of Azure infrastructure services such as Azure Traffic Manager and Azure CDN.

Overall, this project will provide students with practical experience in building and deploying a containerized web application on Azure using container services. It will also give them the opportunity to explore different Azure services and tools for building scalable and resilient applications.