PRACTICAL WORK IN AGILE SCRUM PROJECTS

Work process / method

- Scrum is a management framework that teams use to self-organize and work towards a common goal. It describes a set of meetings, roles, and tools for an efficient delivery of the project.
- During the project, the first step is that we analyzed the project requirements and defined the project goals.
- We discussed the roles and responsibilities of each one of us, ensuring everyone has understood their contributions.
- To follow-up the workflow, we used project management tools and software like Trello. This tool helped us to manage our progress in the project.
- The good thing is we completed our project in a short period and the organization of our team leader was good. The not well part is the time limit for the project was short and the presence of the team members was not constant.
- The difficult part of the project is as this is our first group project, I felt difficult in understanding the roles of each team member and the concept of the project in the beginning.
- Being a team leader, I would make the team members come for the discussion without fail either in direct or online as I felt the members were not present it was difficult to convey our latest information. As we are totally new to this concept of working together on a single project, the time limit should have been extended for more practice in it as this method plays a vital part in the future workplace.

My role

- My role in this project is the front-end development of the web page. In the beginning I created the start side of the page.
- Later, after other part was done in the project, I again worked in the front-end for adding stylings and information in the home page and the checklist page.
- The other members in the team worked with controller, database, making todo list and adding property for Cognito userId, property of prio, showing prio in the checklist and few other functions.
- I would have chosen the database or developing controller or development for the home page. But I liked my role in this project, so I chose it.

Launch

- **AWS Code Pipeline -** AWS CodePipeline is a CI/CD service for building, testing, and deploying applications. We define a pipeline with four stages as source, build, test, and deploy. It integrates with other AWS services to automatically trigger actions, ensuring efficient and continuous delivery of your application. Users can access the application once the deployment stage successfully completes.
- **AWS Code Deploy -** AWS CodeDeploy simplifies application deployment to EC2 instances, Lambda functions, and ECS clusters. We define a deployment group, configure deployment settings, trigger the deployment process, and create the application. CodeDeploy handles the installation, validation, and scaling of the application, ensuring uninterrupted access for users.
- **AWS Code Build -** AWS Code Build is a managed build service that compiles code and runs tests. User should configure the build settings, connect to a source code repository, and generate build artifacts. These artifacts can be used to trigger deployments and launch the project, allowing users to access the application.
- **AWS Code Commit -** AWS Code Commit is a secure Git-based source control service where developers can commit code to repositories, manage versions, and collaborate. It integrates with AWS Code Pipeline for automated CI/CD pipelines, and user can deploy the application using AWS services. Users can access the deployed application through specified endpoints or URLs.
- **AWS Code Artifact -** AWS Code Artifact is a managed artifact repository for storing and sharing software packages. Upload artifacts, manage dependencies, and integrate them into your CI/CD pipeline. Users use the stored artifacts to deploy the application and configure networking components for user access. Code Artifact simplifies artifact management and ensures secure and efficient application deployment.
- **AWS Elastic Beanstalk -** By utilizing AWS Elastic Beanstalk, user can streamline the deployment process and focus on developing your application. Elastic Beanstalk handles the infrastructure provisioning, environment setup, and automatic scaling, reducing the operational overhead. With proper DNS configuration and monitoring, users can access the application using the assigned domain name, providing a seamless experience.

STEPS TO LAUNCH THE PROJECT

- Create a Code Commit repository to host our project's code.
- Configure Code Build to build our application by specifying build commands, testing frameworks and other build-related configurations we want. Code Build will automatically trigger the build process whenever changes are pushed to the repository.

- Create a Code Pipeline that orchestrates the release process. Define stages in the pipeline, such as source, build, test and deploy. Connect the pipeline to your Code Commit repository and Code Build project.
- Within the pipeline, configure the deployment stage to use Elastic Beanstalk. Define the necessary settings, such as the application name, environment, and deployment options.
- Once the pipeline is set, any changes pushed to the repository will automatically trigger the CI/CD process. Code Build will build the application and if all tests pass, Code Pipeline will deploy it to the Elastic Beanstalk environment. The application will be accessible to users through the provided Elastic Beanstalk URL.

AZURE SERVICES

- **Azure DevOps Pipelines (equivalent to AWS code pipeline) -** Azure DevOps Pipelines automates CI/CD for our project. Connect repository, define build/test stages, and deploy to target environment. Monitor performance, scale resources, and configure custom domain routing for user access. Azure DevOps Pipelines streamlines project launch and ensures a seamless user experience.

- **Azure DevOps Release Management, Azure App Service Deployment Slots(equivalent to AWS code deploy)** - By utilizing Azure DevOps Release Management and Azure App Service Deployment Slots, user can establish an effective release pipeline and mitigate risks associated with deploying changes directly to the production environment. Deployment slots allow for thorough testing and validation, ensuring a stable and reliable application. Finally, by configuring DNS and routing, users can access the application through the custom domain, providing a seamless experience.

- **Azure App Service(equivalent to AWS elastic beanstalk)** - By leveraging Azure App Service, user can quickly deploy and host the project with minimal infrastructure management. The built-in scalability options ensure that our application can handle user demand effectively. Configuring custom domains, SSL certificates, and monitoring tools further enhance the application's accessibility and performance. With Azure App Service, we can launch our project and provide users with a reliable and accessible application experience.

GOOGLE CLOUD PLATFORM SERVICES

- **Cloud Build and Cloud Source Repositories(equivalent to AWS code pipeline) -** By utilizing Cloud Build and Cloud Source Repositories, user can automate the build and deployment processes of our project. This allows for efficient development and seamless deployment to our target environment. With proper DNS configuration, routing, and monitoring, users can access the application through the custom domain, ensuring a smooth user experience.
- **Cloud Deployment Manager and App Engine Deployment(equivalent to AWS code deploy) -** By utilizing Cloud Deployment Manager and App Engine Deployment, user can streamline the deployment process and efficiently manage our application infrastructure. App Engine's auto scaling and load balancing capabilities ensure optimal performance and availability. With proper DNS configuration, routing, and monitoring, users can access the application through the custom domain, providing a seamless user experience.
- **App Engine(equivalent to AWS elastic beanstalk) -** By utilizing Google Cloud App Engine, users can focus on developing our application while leaving the infrastructure management to Google Cloud. App Engine handles deployment, scaling, and load balancing, providing a scalable and reliable platform. With proper DNS configuration, routing, and monitoring, users can access the application through the custom domain, offering a seamless user experience.