System Requirements

# Sudershan Pothina

**Requirements Modelling** –

1. Outputs:
   1. Application artifact is built and stored into the artifact repository
   2. Application is deployed to the required environment
   3. Build statistics are sent to the Manager
   4. Deploy statistics are sent to the Manager
2. Inputs:
   1. Developer checks in the code that is ready to be deployed as a container.
   2. Developer requests for a deployment process to a specific environment
3. Processes: There are a number of processes for this application to be deployed correctly. The first step starts with developer checking in the code. The code is uploaded to the code repository and the build process is triggered. The build system will download the code and package it for deployment. Various security scans are performed to make sure that the application is safe to deploy.

After the application package is uploaded to the artifact repository for the next step of the process. Developer is sent a notification that the build is successful. The developer will make another request for the application to the deployed to an environment. Manager will get this request and approve it. The approval will trigger the deployment process and will check if the right resources are available. After all the resources are available, the application is deployed and notifications are sent to developer that the application is deployed successfully. Managers will be sent the deployment statistics every week containing the number of build requested each day and the amount of time it took for the process to complete. This report will also contain information on how many errors were encountered and the time it took to fix the process.

1. Performance: Performance of the build system and the application cannot be taken lightly. As this is a customer facing application the speed at which the application responds is monitored constantly. If the application is responding slower than usual or if the resources like processors and memory are insufficient then a notification is sent to the server administrator and necessary resources are increased.

The build process performance is also important. If there are any errors encountered during the build process Application administrator is immediately notified and follow up actions are taken to mitigate the issue. This will be treated as a high priority incident and will be fixed as soon as possible.

1. Security: Building and deploying the application securely is critical to protect the company from any cyber threats. Code is scanned for both code vulnerabilities and security vulnerabilities. Scan results are generated and analyzed. If there are any critical scans records found, the build process is stopped and the scan report is sent to security team for further analysis and remediation steps.

**Data Flow and context diagrams**

High level Context – Please refer data dictionary for detailed description 

Detailed Context – Please refer data dictionary for detailed description



**Data Dictionary**

1. Data Flow
   1. Name: automatic build and deploy process
   2. Description: process to describe how an automated build will take place
   3. Alternate names: None
   4. Origin: Developer will usually start the automation process by checking in code
   5. Destination: Different notifications will be send to the developers, managers and the server administrator and the artifact that was built will be deployed to a particular environment
   6. Volume and frequency: The frequency will be 5 to 10 times per day.
2. Data stores
   1. Code Repository
      1. Name: Code Repository
      2. Description: place to store all the versions of the code
      3. Alternate Names: None
      4. Volume and frequency: Will be accessed on a very regular basis when a commit process is initialized by the developer
   2. Artifact Repository
      1. Name: Artifact repository
      2. Description: stores the artifact that is built during the build process
      3. Alternate name: None
      4. Volume and frequency: Will be accessed 10 to 20 times a day. Twice as that of the number of the build
   3. Environment
      1. Name: Environment
      2. Description: Environment the application is deployed to
      3. Alternate name: none
      4. Volume and frequency: 5 to 10 times a day
3. Processes
   1. Save code to repository
      1. Name: save code to the repository
      2. Description: process to save code to the repository
      3. Process number: 1
      4. Process description:
         1. Input flow: Upload code to the repository
         2. Output flow: trigger build
         3. Description: process starts when a request to upload the code is received by the developer and after the process ends a build is triggered
   2. Build artifact
      1. Name: Build artifact
      2. Description: build an artifact from the code committed to the repository
      3. Process number: 2
      4. Process description:
         1. Input flow: Trigger build
         2. Output flow: build notifications, resource utilization notification, build errors and artifact upload
         3. Description: This process will start as soon as build trigger is received. It will build the code artifact and upload it to the artifact repository. This process will send notifications to the developer on the status of the build. This process will also send cpu and memory usage statistics to the Server administrators. This process will also send notifications to the managers and the System administrators.
   3. Deploy application
      1. Name: Deploy application
      2. Description: deploy and artifact to an environment specified
      3. Process number: 3
      4. Process description:
         1. Input flow: Upload artifact to the artifact repository
         2. Output flow: Deploy to an environment
         3. Description: This process will start when a new artifact is uploaded to the artifact repository and will be deployed to the environment mentioned in the process
4. Entities
   1. Developer
      1. Name: Developer
      2. Description: Developer that creates or modifies the code
      3. Alternate Name: None
      4. Input data flows: Build notifications
      5. Output data flows: Trigger build
   2. Manager
      1. Name: Manager
      2. Description: Manager of the development team
      3. Alternate Name: None
      4. Input data flows: Build and deployment statistics
      5. Output data flows: None
   3. Server Administrator
      1. Name: Server Administrator
      2. Description: server administrator who maintains the system
      3. Alternate Name: None
      4. Input data flows: resource utilization notifications
      5. Output data flows: Add more resources to the build platform
   4. Application Administrator
      1. Name: Application Administrator
      2. Description: application administrator who maintains the build process
      3. Alternate Name: None
      4. Input data flows: Build error notification
      5. Output data flows: None

**Decision Tables for process modeling:**

1. Build Process – Build process logical model on when an artifact should be build. The scan process can be bypassed if there is a production incident and manager signoff is available. The build cannot be uploaded if the build command fails.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Build artifact succeeded | Y | N | Y | Y | Y |
| Code scans succeeded | Y | Y | - | N | N |
| Production Incident | - | Y | Y | Y | N |
| Manger sign off | - | Y | Y | N | Y |
| Build | Y | N | Y | N | N |

1. Deploy Process – Deploy process will only succeed if all the conditions i.e., the artifact is available in the artifact repository, the environment has the necessary resources and there is an approval by the Manager.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact available | Y | N | Y | Y |
| Resources available | N | Y | N | Y |
| Manager approval | Y | Y | Y | Y |
| Deploy | N | N | N | Y |

1. Add more resources – Adding more resources to the build and the deploy process will be based on the following decision table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| CPU is low | Y | Y | - | - | N | N |
| Memory is low | Y | - | Y | - | N | N |
| Application needs special additional resources | - | - | - | Y | Y | N |
| Manager Approval | - | - | - | Y | N | Y |
| Add more resources | Y | Y | Y | Y | N | N |

**Object Modelling:** Identifying the list of objects

1. Employee :
   1. Name: Employee
   2. Attributes:
      1. Name
      2. Employee ID
      3. Experience
   3. Methods: None
2. Developer:
   1. Name: Developer inherits Employee
   2. Attributes:
      1. Reporting Manager – Manager who he is reporting to
      2. Programming Language – programming language that the developer uses
   3. Methods:
      1. Create or modify code – create new code or modify existing code of an application
      2. Commit code – commit the changed code to the code repository
      3. Fix code – fix broken code
      4. Review code scan results – reviews the scan results generated by the build process
      5. Document code – document the code usage for other developers
3. Manger:
   1. Name: Manager inherits Employee
   2. Attributes:
      1. Developers – list of developers who report to him
   3. Methods:
      1. Hire new employee
      2. Fire an employee
      3. Review build statistics
      4. Review deploy statistics
4. Server Administrator
   1. Name: Server Administrator inherits Employee
   2. Attributes:
      1. Servers Supported – list of servers supported
      2. Operating systems – list of operating systems supported
   3. Methods:
      1. Document Operating system installation – document the process of operating system installation
      2. Fix server errors – fix any errors encountered by the servers login
      3. Increase CPU – increase the number of processors
      4. Increase memory – increase the amount of memory of a system
      5. Build new server – build a new server
      6. Tear down a server – remove an existing server and repurpose the resources.
5. Application administrator
   1. Name: Application Administrator inherits Employee
   2. Attributes:
      1. Applications Supported – list of application supported
      2. Number of applications deployed – number of applications that have been deployed by this administrator
   3. Methods:
      1. Document tool installation – document the process of tool installation
   4. Fix build errors – fix any errors encountered by the builds
6. Build Artifact
   1. Name: Build Artifact
   2. Attributes:
      1. Build number – number of the build
      2. Application Name – name of the application being build
   3. Methods:
      1. Pull code – download code from code repository
      2. Run security scans – scan code for security and code vulnerabilities
      3. Build artifact – package the code into a deployable unit
      4. Send build status – notify the developers on the build status
      5. Send build statistics – notify managers weekly about the build metrics
7. Deploy Artifact
   1. Name: Deploy Artifact
   2. Attributes:
      1. Artifact name – name of the artifact being deployed
      2. Application name – name of the application being deployed
      3. Resources required – list of resources required for the application to be deployed
   3. Methods:
      1. Send deployment notification – send notification after deployment is complete
      2. Deploy application – deploy an application into a selected environment
8. Scan Code:
   1. Name: Scan code
   2. Attributes:
      1. Build id – id of the application build
      2. Build name – name of the application
      3. Scan id – id of the scan performed
      4. Scans results – list of warnings and critical errors
   3. Methods:
      1. Scan code – scan the code that is being built
      2. Send scan notifications to security – send the scan results to security

**Use case diagram –**

1. Build process



1. Deploy process



**Use case description –**

1. Build Process for an application
   1. Name: Build Process for an application
   2. Actor: Developer/ Manager/ Application Administrator
   3. Description: Defines the process of automatically building an application with build, scan, publish and notifications
   4. Successful completion:
      1. Build artifact is uploaded to the artifact repository
      2. Notifications are send to the developers
      3. Code has been scanned successfully
   5. Alternatives:
      1. Build error is sent to the Application administrator
      2. Vulnerabilities list is send to security
   6. Precondition: Developer commits code
   7. Post condition: Build artifact is available for deployment
   8. Assumptions: The code is build using container technology
2. Deploy process of an application
   1. Name: Deploy process of an application
   2. Actor: Developer/Manager/Server administrator
   3. Description: Use case to show the automatic deployment of an artifact to an environment
   4. Successful completion:
      1. Application is deployed to the respective environment
      2. Deployment statistics are stored
      3. Notifications containing the statics information is send to the Managers
   5. Alternative:
      1. Server Administrator is notified about the low resource availability
   6. Precondition: Developer requests for the deployment process
   7. Post conditions: the specific artifact in the request is deployed to the environment requested
   8. Assumptions: The environment that the application is deployed to will support container technology