

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **ID** | **Description** | **Remarks** |
| 1 | **Create Release YYMMDD Branch**  Tech Lead will create Release YYMMDD branch in Local GitHub. This branch will be used to merge changes from Developers. Naming standard for this branch will be Release# (YYMMDD) |  |
| 2 | **Create YYMMDD Feature Branch**  Each Developer will fork feature branch from Tech Lead’s Release YYMMDD Branch.  **Developer will be responsible for below tasks**   * + - Implement task according to design and best practices.     - Fix bugs and issues.     - Create, fix, and maintain unit and integration test.     - Create, fix, and maintain database scripts     - Ensure components, plugins, tools, and dependencies are approved technology. | Developers should have the application running in their local machine to Test the changes. |
| 3 | **Developer’s Pull Request**  After Development is completed, Developer will raise Pull request in GitHub for Tech Lead to Review the code changes. **Each Developer should provide comments (SCR#) on the code changes**.   |  |  | | --- | --- | |  |  | |  |
| 4 | **Tech Lead Code Review**  Tech Leadwill review the code and approve/Reject it. Tech Lead will provide required comments for approving or rejecting the Pull request. Both the coder and approver will be responsible for the set of code.   * + - If code is rejected by Tech Lead during Review, Developers needs to review the comments provided by Tech Lead and make suggested modification. After required changes, Developer can raise another Pull request for Tech Lead.     - If code is approved by Tech Lead during Review, it will automatically merge by GIT to the Tech Lead’s Release YYMMDD branch or manually merged by Tech Lead owner to Release YYMMDD branch. |  |
| 5 | **Tech Lead’s Pull Request**  Tech Lead will raise Pull request in GitHub for code Review/Merging the changes to Release/Developer Branch. If there are multiple Teams working on the Project. Tech Lead can assign the Code Review request to other Tech Lead working on the Project. Project Tech Lead will be responsible for reviewing the code changes.   * + - If code is approved by Project Tech Lead during review, it will automatically merged by GIT to Release / Development Branch.     - If code is rejected by Project Tech Lead during review, it will go back to respective App Team Lead | SonarQube will help us automate code review and able to regularly check our code against the standards and best practices at early stage.  SonarQube will be integrated as part of the CI step. |
| 6 | **Develop Branch**  This is the Branch where Continuous Integration (CI) for Dev-Int Environment is linked. Any changes in this branch will trigger the CI - Build, Test, and Release for Dev-Int Environment. |  |
| 6a | **Development Environment Build**  Once code changes are detected in *Develop* Branch, CI will trigger and will run series of build steps defined in Build definition. Basic build steps are:   * + - Build the solution. If one test fails it will return error and will not proceed to Test build step.     - Run Unit and Integration Tests. Automated tests scripts are executed against the build. If one test fails it will return error and will not proceed to Quality Gates.     - Validate Quality Gates. This includes code coverage, sonarqube quality gates, etc. If quality baseline is not met it will return error and will not proceed to release.   Build Trigger is configured as Continuous Integration (CI) which means it will run the build process every time *code changes* is detected in Release/ *Development* Branch. | Email is sent out on Build success/failure  **Tests**   * Unit Tests * Integration Tests * SonarQube   \*- these tests can be scheduled tests to run nightly, weekly, or depending on the needs. |
| 6b | **Development Environment Deployment**  Once Build is successful it will deploy the build to development environment. In this environment we don't need approval.  **Basic deployment steps are:**   * + - Deploy the artifacts created by the Build step.     - Run Application Availability Test (AAT)     - Performance test using Blazemeter (Optional)   In development environment releases we don't need *approvals from QA Team* because this environment is for developers  This Step is configured as Continuous Deployment (CD) which means it will run the release process every time *build artifact changes* are detected in the Build. | Email is sent out on Deployment success/failure  UI dependent tests will be executed after artifacts is deployed to the server(s).  **Tests**   * AAT Test. * Performance/Load Tests (Optional)\*   \*- these tests can be scheduled tests to run nightly, weekly, or depending on the needs. |
| 7 | **Development Environment**  This environment is for developers to check/verify the fix in real time. Every time the developer check in a code in Development branch it will automatically deployed in this environment.  This environment is also use to troubleshoot CI/CD issues and bugs. | The server’s setup in this environment is not production grade.  Owned by developers but accessible to all to do quick checks and validation |
| 8 | **Tech Lead’s Pull Request**  Tech Lead will raise Pull request in GitHub for code Review/Merging the changes to Integration Branch. | Pull Request will be assigned to [IR-Configuration-Management](https://github.aig.net/orgs/LR-Individual-Retirement/teams/ir-configuration-management) |
| 9 | **SCR Review**   * + - CM Team will review the SCR status in SBM.     - If SCR is approved, CM Team will merge the changes to Integration Branch. |  |
| 10 | **Integration Branch**  This is the Branch where Continuous Integration (CI) for QA Environment is linked. Any changes in this Integration branch will trigger the CI - Build, Test, and Release for QA Environment. |  |
| 11 | **QA Environment Build**  Once code changes are detected in *Integration* branch, CI will trigger and will run series of build steps defined in Build definition. Basic build steps are:   * + - If Build fails it will return error and will not proceed to Test build step.     - Run Automated AAT, Unit Test. If one test fails it will return error and will not proceed to Quality Gates.     - Validate Quality Gates. This includes code coverage, sonarqube quality gates, etc. If quality baseline is not met it will return error and will not proceed to deployment     - Performance test using Blazemeter   Build Trigger is configured as Continuous Integration (CI) which means it will run the build process every time *code changes* is detected in *master* Branch.  Build can be executed manually via Jenkins. | **SIO/AppGov Tests**   * Code Review scanning (SonarQube)\* * Veracode\*   \* These tests can be scheduled tests to run nightly, weekly, or depending on the needs. |
| 12 | **QA Environment Deployment**  Once Build is successful and approved, CM Team will deploy the build to QA environment using Jenkins.  **Basic deployment steps are:**   * + - Deploy the artifacts created by the Build step.   **Post Deployment**   * + - Run Application Availability Test (AAT)     - Performance test using Blazemeter.     - Regression Testing | This environment is owned by QA Team.  **Tests**   * Application Availability Test (AAT) * Performance Test (Optional)\*   \*- these tests can be scheduled tests to run nightly, weekly, or depending on the needs. |
| 13 | **QA Environment**  This environment is for QA to run different tests, manual tests, and application compliance test.  This environment is also use to reproduce issues and bugs reported by the users. | Owned by QA but accessible to all.  The build deployed in this environment is production grade but the infrastructure is not. |
| 14 | **QA Monitoring**  QA monitoring and logging tools are integrated to give the team deeper details of the issues and bugs that cannot be captured in the application logs. Monitoring tools can send emails to respective users if detects undesirable metrics and statistics. | Monitoring tools: AppDynamics, Splunk.  This is managed by CM team.  Everyone have view access to monitoring tools in QA environment |
| 15 | **Tech Lead’s Pull Request**  Tech Lead will check for QA Management approval for UAT Move and raise Pull request in GitHub for code Merge to UAT Branch. |  |
| 16 | **SCR Review**   * + - CM Team will review the SCR status in SBM.   If SCR is approved, CM Team will merge the changes to UAT Branch. |  |
| 17 | **UAT Branch**  This is the Branch where Continuous Integration (CI) for UAT Environment is linked. Build will promote from Integration Branch to UAT Branch. |  |
| 18 | **UAT Environment Build**  CM Team will promote/New Build\* the Build from QA to UAT.  \*New Build will only be allowed in exception cases. |  |
| 19 | **UAT Environment Deployment**  CM Team will deploy the build to UAT environment using Jenkins.  **Basic deployment steps are:**   * + - Deploy the artifacts created by the Build step.   **Post Deployment**   * + - Run Application Availability Test (AAT)     - Performance test using Blazemeter.     - Regression Testing | This environment is owned by QA/ABBS Team.  **Tests**   * Application Availability Test (AAT) * Regression Testing * Performance Test   \*- these tests can be scheduled tests to run nightly, weekly, or depending on the needs. |
| 20 | **UAT Environment**  This environment is for QA to run different tests, manual tests, and application compliance test.  This environment is also use to reproduce issues and bugs reported by the users. | Owned by QA/ABBS but accessible to all.  The build deployed in this environment is production grade both build and infrastructure. |
| 21 | **UAT Monitoring**  UAT monitoring and logging tools are integrated to give the team deeper details of the issues and bugs that cannot be captured in the application logs.  Monitoring tools can send emails to respective users if detects undesirable metrics and statistics. | Monitoring tools: App Dynamics, Splunk.  This is managed by CM team.  Everyone have view access to monitoring tools in UAT environment |
| 22 | **Tech Lead’s Pull Request**  Tech Lead will check for QA/ABSS Management approval for Production Move in SBM.  Tech Lead will raise Pull request in GitHub for code Merge to Master Branch. |  |
| 23 | **PCR Review**   * + - CM Team will review the PCR status in SBM.   If SCR is approved, CM Team will merge the changes to Master Branch. |  |
| 24 | **Master Branch**  Master Branch will go to Production. Only CM Team will have merge access to this branch. |  |
| 25 | **Production Environment Build**  CM Team will promote the Build from Integration UAT to Production. |  |
| 26 | **Production Environment Deployment**  CM Team will deploy the build to Production environment using Jenkins.  **Basic deployment steps are:**   * + - Deploy the artifacts created by the Build step.     - Run Application Availability Test (AAT) | This environment is owned by QA/ABBS Team.  **Tests**   * Application Availability Test (AAT) |
| 27 | **Production Environment**  This environment is available for business and customers. |  |
| 28 | **Production Monitoring**  Production monitoring and logging tools are integrated to give the team deeper details of the issues and bugs that cannot be captured in the application logs.  Monitoring tools can send emails to respective users if detects undesirable metrics and statistics | Monitoring tools: App Dynamics, Splunk.  This is managed by CM team.  Everyone have view access to monitoring tools in UAT environment  Health Check |

* Each Error Marked with Red. Developer& Tech lead will be responsible for foxing the error. Once the error is fixed New Pull Request needs to be raised.
* SonarQube & Veracode scan will not be available for June Release.
* App Dynamic Availability in QA Environment will be based on License availability.
* **Process Change** - NO Release Note will be required for Code Changes. Release Notes will be required for DBCR changes as DBCR will remain on PVCS as part of June Release.