Software Quality Assurance Plan

**ChequeMeOut**

notsirkApps

**Prepared By**

|  |  |
| --- | --- |
| DOCUMENT OWNER(S) | Organization role |
| Kriston Sanders | Project Manager |

**Version Control**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author(s) | Change description |
| 1.0 | 6/8/20 | Kriston Sanders | Document created |
| 1.1 | 7/27/20 | Kriston Sanders | Updated coding standards |
| 1.2 | 7/30/20 | Kriston Sanders | Added naming conventions |

**Approvals**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Approving party | Signature |
| 1.0 | 6/10/20 | Kriston Sanders | Kriston Sanders |
| 1.1 | 7/27/20 | Kriston Sanders | Kriston Sanders |
| 1.2 | 7/30/20 | Kriston Sanders | Kriston Sanders |

| 1.0 Introduction |
| --- |
| * 1. **Purpose** * Reduce wait times, contact, and eliminate the need to split checks * Launch product that is fun to use and intuitive * Create product to highlight skills and impress potential employers * Be transparent about data collection/use * Accommodate transition between methods * Accommodate non-users * Low adoption cost   1. **Document Definitions** * Diner- customers of a restaurant or bar * Server- employee who interacts with the diners and tracks orders * Kitchen- employees who cook food * Owner- the owner of the bar or restaurant   1. **Intended Audience** * Development team * Recruiters   1. **Scope** * Customer mobile app for ordering and payment * Business applications for accepting and tracking orders and payments * Integrate with existing payment system (Square) * Secure data transmission * Store business and transaction data with Square or independent server   System will improve the dining out experience by reducing wait times, contact, and eliminate the need to split checks by offloading tasks to a suite of mobile applications. This system will bring in revenue through product licenses while bolstering my resume with a demonstration of my skills while also advancing them.   * 1. **References** * Test Strategy * Test Plan * Test Metrics Report * Requirements Testing Matrix |

| 2.0 Plan overview |
| --- |
| * 1. **Documents**   The following documents will be created and updated throughout the project:   * Design Document * Risks and Mitigations * Software Quality Assurance Plan * Test Plan * User guides:   + Diner   + Owner   + Server/Kitchen   1. **Standards**      1. **Coding** * Visual Studio used for development * Xamarin/C# used for coding * SQL database for data storage * Git code repository * Clean code principles:   + - No getters/setters     - “If” only as guard clause     - Composition over inheritance     - Objects over public statics     - Objects over enums     - Don’t tell an object what to do, ask it to do something for you (behavior)     - Avoid primitives- use fluent types instead   + Single responsibility principle * Open closed principle * Dependency inversion principle * Naming conventions:   + - Properties begin with underscore and use camel case nouns     - Methods use Pascal case and begin with a verb and end with a noun     - Variables use camel case nouns     - Classes use Pascal case nouns     - Constants use all caps nouns and underscores to separate words     - Unit tests use camel case and communicate property, state, expected output     1. **Documentation** * Version tracking * Signoff to finalize versions * Unit tests indicate intent   + XML commenting for classes, methods, and properties, stored in a separate file and referenced by the <include> tag within classes     1. **Testing** * TDD unit testing * Continuous Integration * System testing * Release testing * User testing   Refer to the Test Strategy and Test Plan for further details.   * 1. **Metrics**   The Test Metrics Report will be used to monitor quality throughout the project. A new report will be generated at the end of each iteration and a master report for the entire project will be maintained with each sprint as well. See the Test Metrics Report and the Test Strategy document for further details.   * 1. **Management**   Being a team of one, all tasks will be performed by one person including Developer, QA Manager, Test Manager, etc.   * 1. **Techniques** * Test-driven Development will force modular design, the coupling dependencies for more maintainable code * SOLID principles will drive development * Code Hot Spots will focus refactoring, to be done after completion of each feature   1. **Records/Reporting**   As this is a team of one, code reviews and inspections are not expected to be useful. Therefore, bugs and QA non-adherence will be addressed during refactoring/testing and tracked as PBIs in Azure DevOps. Test Metrics Reports and the Requirements Testing Matrix will also maintain records of plan conformance. |

| 3.0 Tasks, Activities, and outcomes |
| --- |
| * 1. **Product Assurance**      1. **Tasks/Activities** * Refactor code every time a requirement is complete with passing tests * Separate XML comments for classes, methods, and properties into a separate file * Write at least one test per requirement   + 1. **Outcomes** * Product meets all must-have requirements * Product follows SOLID principles * Product is extensible * Product has low coupling and high cohesion * Dependencies are extracted out with interfaces * Code is readable and self-documenting * Code is properly documented to clear up and ambiguity * Code is fully tested with no defects   1. **Process Assurance**      1. **Tasks/Activities** * Produce a Test Matrix Report for relevant iterations * Maintain the Requirements Testing Matrix to ensure test coverage * Track bugs/defects/plan nonconformance in the backlog   + 1. **Outcomes** * Iterations conform to this plan * Each iteration produces a Test Matrix Report where relevant * The Requirements Testing Matrix is always up to date * All bugs/defects/plan nonconformance tracked in the backlog |

| 4.0 Additional considerations |
| --- |
| * 1. **Service Terms**   The following service terms must be adhered to:   * iOS developer * Android developer * Square API * Azure DevOps * Any others where applicable, such as packages and other APIs or services |