Cake it

1. Introduction to the project

"Cake it" is a cake baking mobile application that enables any cake specializing bakery's counter staff, cake decorators, to take user's order and bake for them a perfect cake and deliver it to their house in minutes. This application provides all type of cakes like birthday cakes, anniversary cakes, wedding cakes and many more. It is easy for the user to choose and find the most suitable cake for them. The user can either choose from available cake that provided by different bakeries or they can design a new cake for them self's, "cake it" allows the user to choose the size of the cake, color of the cake, and flavor of the cake etc. In this project we will produce a detailed quality plan and different quality aspects related to this application. The quality plan will include project scope, constraints, functional requirements, integration requirements, and the quality goals.

2. Purpose of quality plan for project

A Software product must achieve excellence in all of its aspects from its analysis phase to its deployment and maintenance. Quality plan is the key to gain not only the high quality for the product but also customers' satisfaction, defects decrease and much more.

The quality plan sets out the desired product qualities and how these are assessed and define the most significant quality attributes. It should define the quality assessment process. Quality plan provides essential elements such as quality goals, review activities, software tests, planned acceptance tests for externally developed software, configuration management plans, tools, procedures and data for version releases to help in assessing and testing the software product to reach the desired quality.

A software product's quality has an impact on project's quality, procedures, criteria, roles, responsibilities and even the workflow.

3. Project scope

"Cake it" is an application that provides a service to deliver chosen and designed cake into a specific location chosen by the customer it might be home, university, workplace ...etc. The application is in agreement with a specific bakery cake store. it represents predesigned menu of cakes also, it allows the customer to create his own design by choose everything to make his cake from the flavor to the additions, number of layers, the shape of the cake and so on so.

This application will help different categories, such as people they do not have transport services, people who want to surprise someone without care about the long distance between them, and much more.

4. General constrains

1.The programming codes and scripts shall be written in Java programming language.

2. The system should not use more than 10% of the device battery in 2 hours' time.

3. Downloading the application shall not take more than 200MB of the device memory.

4. The total cost of the project must not exceed $40,000.

5. The project must be completed by the deadline (January 20 ,2021).

6.The system shall be able to handle up to 10000 concurrent users.

5. Functional Requirements

1. The user shall be able to view cake details (size, flavor, number of layers, topping, shape).

2. The user shall be able to view cakes menu.

3. The user shall be able to set delivery information (location, date, time).

4. The user shall be able to register.

5. The system shall be able to send a conformation email to the client after the payment.

6. System and Integration Requirements

1.User authentication shall be managed centrally, allowing administrators to control access to the system.

2.The database management system shall have the ability to perform data manipulation (edit, delete, insert) using standard SQL directives.

3.The system should be able to operate in IOS and Android platforms.

4.All data modifications shall be recorded with the detailed information about who and when it modified.

7. Quality goals

ID	Requirements	McCall's Quality Factors
NR1	Time to detect and fix problems should not exceed 50 minutes.	**Maintainability**
NR2	The system shall not shut down for maintenance more than once in a 72-hour period.	
NR3	The size of a software module will not exceed 300 statements.	
NR4	New releases of the system shall not affect database's content and all customer's settings, that all customers information and preferences shall left just like it was before re- leasing.	

NR5	The system shall be able to handle up to 1000 concurrent users.	**Reliability**
NR6	The system shall be capable of restoring itself to its previous state within 15 seconds in the event of failure.	
NR7	The system defect rate shall be less than 1 failure per 1000 hours of operation.	
NR8	The system shall be developed to be integrated with Google Maps system, so that customers can specify their locations.	**Interoperability**
NR9	The system shall support IOS and Android operating systems.	**Portability**
NR10	The system should be authenticated, user data requests will use SSL/TLS over port 443 to encrypt requests	**Security**
NR11	The average time for the user to learn how to use the system shall be less than 1 hour.	**Usability**
NR12	All system's functionalities shall be available after no more than one clicks from the home page.	
NR13	The system should not use more than 10% of the device battery in 2 hours' time.	**Efficiency**
NR14	Scrolling one page up\down in the menu section page shall take at most 10 milliseconds to load.	
NR15	The Online Payment System shall be available for use between the hours of 6:00 a.m. and 11:00 p.m. CST.	**Availability**
NR16	The Online Payment System shall achieve 100 hours of MTBF (mean time between failure).	
NR17	The system shall be implemented for the future usage of multiple languages.	**Flexibility**
NR18	All the system component shall pass the testing plan in no more than one week.	**Testability**

NR19	The system shall be able to secure 400 active customer payments, that is no payments shall be lost.	**integrity**
NR20	The system shall not accept users' passwords that are less than 10 characters.	

8. Planned reviews

Phase 1: Requirement gathering and Analysis

1. Requirement specification document.
2. Function Requirement correctness.
3. Nonfunctional requirement completeness.

Phase 2: Design

1. Preliminary Design Review
2. Detailed Design Review
3. Database Design Review

Phase 3: Implementation

1. Software source code
2. Post-Implementation review report
3. User Manuals

Phase 4: Testing

1. Test Plan Review
2. Software Test Procedure Review
3. Test Readiness Review

Phase 5: Maintenance

1. Add new functionality to the system.
2. Updating the system.
3. System audit.

***Walkthroughs*:**

Requirements gathering and analysis:

1. Requirements refinements contract: Are all the refinements approved by the customer?

2. Requirements specification document: Does each requirement consistent with other requirements in this document and with the stated goals of the software?

3. List of all "Cake it" application recommendations: Are all recommendations sufficient and applicable?

Software design:

1. Deployment diagram: Do all nodes and links in the deployment diagram denoted correctly?

2. Component diagram: Are all the needed 'Provided' and 'Required' interfaces included in the diagram?

3. Database design: Are the data in the database consistent?

Implementation:

1. Software source code: Does the source code implements all the functions? Do the functions contain required information to be executed?

2. Post-Implementation review report: does the report has been summarized and evaluate of all implementation activities?

3. User Manuals: Does the user manual provide clear and detailed steps procedure which designed for the users who use the system in their first time or occasionally users? Do the instructions map to the product in all respects?

Software testing:

1. Usability testing: Are all functions accessible and easy to perform from the user's first attempt?

2. Unit testing: Are all methods in the code validated?

3. Integration testing: Are all defects in the interfaces and in the interactions between in targeted components discovered?

Software maintenance:

1. Software detection and correction report: Does the report have clear and detailed descriptions for each detected software deviation and its corresponding correction?

2. Adding a new functionality to the system: Is the code able to accommodate to a new module?

3. A list of risk assessment that found in the application: Are all risks discovered?

9. Planned Verification Tests

1.Testing if the application permit customers to view only their own previously placed orders, not orders placed by other customers.

2.Testing if the customer performs "place order" command in their first attempt no longer than 2 minutes.

3.Testing if network transactions that involve financial information or personal information encrypted using BR-33.

4.Testing if the application calculates Value-added Taxes according to the KSA VAT regulations which is 5%.

5.Testing if Time between user event and application respond should not exceed 10.

10.Planned Validation Tests

1.Testing if the application can let the user make new order.

2.Testing if the application can let the user login.

3.Testing if the application can let the user logout.

4.Testing if the application can let the user update his/her account.

5.Testing if the application can let the user cancel his/her order before the delivery starts.

11. Planned acceptance Tests

There will be five acceptance testing:[3]

1. First acceptance testing will be after the requirement specification phase: The customer will attend a storyboard, and it will consist of sample application outputs and simulation of the application.

2. Second acceptance testing will be after designing the software: There will be a prototype that the user will interact with. The prototype must implement all the functional requirement provided by the customer, so that the customer can provide feedback.

3. Third acceptance testing will be contract acceptance testing: the developed software is tested against certain criteria and specifications which are predefined and agreed upon in a contract.

4. Fourth acceptance testing will be Regulation Acceptance Testing: it examines whether the software complies with the regulations, laws and safety standards. This includes governmental and legal regulations.

5. The last acceptance test is before releasing the product: check for any failure. by releasing the product to group of users to use this product and report any bugs or issues they faced and take their feedback to correct the product.

12. Planned configuration management.

Software Storage:

"Cake it" application will use DBMS as a storage and server. DBMS means there is a physical server on site. DBMS provides a framework for better enforcement of data privacy and security policies. DBMS provides Better data integration since it promotes an integrated view of the organization's operations and a clearer view. also, it becomes much easier to see how actions will be.[1]

Security and backups:

"Cake it" application will be using HTTPS protocol for data encryption. The principal motivation for HTTPS is authentication of the accessed application and protection of the privacy and integrity of the exchanged data. It protects against man-in-the-middle attacks. The bidirectional encryption of communications between a client and server protects against eavesdropping and tampering of the communication. For the backups, "Cake it" application will use MSP backup and recovery software. SolarWinds MSP (formerly LOGIC now) provides speedy backups, layers of network security and flexibility in database recovery and support. True Delta Technology tracks file changes at the block level, only copying what has changed and allowing for backups during business hours. It's compatible across multiple platforms including SQL Server and more.[2]

Version Control:

"Cake it" application will be using GIT tool for managing the work as well as the different versions. Since it capable of efficiently handling small to large sized projects like "Cake it" application and it has a cryptographic authentication to increase the security. Also, GIT offers GIT GUI where we can very quickly re-scan, state change, sign off, commit & push the code quickly with just a few clicks.

13. Quality Assurance Process Metrics

Metrics are numbers that tell you accurate measurements about how the process, the project and the product meet with the specified quality. Metrics provide base for you to suggest improvements. Usually measuring results with one metric is not a good enough strategy. A combination of metrics is used to measure the quality.
In order to measure the quality of the different aspects of the project, different metrics will be used to evaluate the 5 phases of software development lifecycle (SDLC). The metrics are:

1.Completeness Metric:

Completeness metric is related with number of unique requirements. This metric is used to assess whether the set of requirements is complete or not. Requirements are considered as complete if all pages are numbered, all figures and tables have captions and numbers, all references are present, all sections and subsections are numbered.

2.Structural Complexity Metric:

This metric is used to determine the structural complexity of the chosen design architecture using the modules, we will raise the number of modules that are subordinating another module in the design architecture to the 2nd power.

3.Development productivity Metric:

This metric is used to determine the development productivity by comparing the total working hours invested in the development of the software system with the total thousands of lines of software code.

4.Code Error Density Metric:

This metric is used to determine the code error density by comparing the total number of errors detected by code inspections and testing process with the total thousands of lines of software code.

5.Software System Failure Density Metric:

This metric is used to determine the failures density by comparing the total number of software failures detected during a year of maintenance service with the total thousands of lines of maintained software code.

1. Requirements gathering and analysis:

Measuring the Requirements gathering and analysis phase by using Completeness Metric.

Metric Name	Metric	What it does?	Data need to be collected?
(Q_{cm}) Completeness	**$Q_{cm} = R_{un} / R_t$** Completeness calculated by dividing the number of unique requirements on total number of requirements.	Used to measure "Cake it" total number of functions currently specified during the requirement analysis phase.	1. The number of unique requirements. 2. The total number of requirements.

2. Software design:

Measuring the software design phase by using Structural Complexity Metric.

Metric Name	Metric	What it does?	Data need to be collected?
($S_{(j)}$) Structural complexity	**$S(j) = f^2 out(j)$** Structural complexity is computed by, $f_{out(j)}$ means number of modules that are subordinating module raised to the 2nd power.	Used to measure "Cake it" overall structural design's complexity in terms of software modulus during the soft- ware design phase.	1. The number of modules that are subordinating a module.

3.*Implementation:*

Measuring the implementation phase by using Development Productivity Metric.

Metric Name	Metric	What it does?	Data need to be collected?
(DevP) Development Productivity	**DevP = DevH/ KLOC** Development Productivity calculated by dividing the Total working hours invested in the development of the software system on thousands of code lines.	Used to measure "Cake it" development productivity by using total working hours invested in the development of "Cake it" system in thousands of code lines.	1. Total working hour invested in the development of the software system. 2. Total thousands of lines of software code.

4.*Software testing:*

Measuring the software testing phase by using Code Error Density Metric.

Metric Name	Metric	What it does?	Data need to be collected?
(CED) Code Error Density	**CED = NCE / KLOC** Code Error Density calculated by dividing the number of code errors detected by code inspections and testing on thou-sands of code lines.	Used to measure "Cake it" code error density in thousands of code lines in the testing phase.	1. Total number of code errors detected by code inspections and testing. 2. Total thousands of lines of software code.

Cake it

5.*Software maintenance:*

Measuring the software maintenance phase by using Software System Failure Density Metric.

Metric Name	Metric	What it does?	Data need to be collected?
(SSFD) Software System Failure Density	**SSFD = NYF/KLMC** Software system failure density will be calculated by using the number of software failures detected during a year of maintenance service divided by Thousands of lines of maintained software code.	Used to measure "Cake it" system failure's density during 1 year of operation.	1. Total number of software failures detected during a year of maintenance service. 2. Total thousands of lines of maintained software code.

References:

[1]: Myreadingroom.co.in. What Is DBMS? Advantages And Disadvantages Of Database Management System(DBMS). [online] Available at: <http://www.myreadingroom.co.in/notes-and-studymaterial/65-dbms/462-advantages-and-disadvantages-of-dbms.html> [Accessed 1 April 20

[2]: Solarwindsmsp.com. *Types Of Database Backups | Solarwinds MSP*. [online] Available at: <https://www.solarwindsmsp.com/content/types-of-database-backups> [Accessed 1 April 2020].

[3]: https://usersnap.com/blog/types-user-acceptance-tests-frameworks/