Project Plan

OrDinner

InfoSupport

Date	**:**	**19-09-2022**
Version	**:**	**1.0**
State	**:**	**Not done**
Author	**:**	**DB02 team 1 (OrDinner)**

Version history

Version	Date	Author(s)	Changes	State

Distribution

Version	Date	Receivers

Contents

1. Project assignment

1.1 Context

Our client is InfoSupport. They want us to make an application which can be used to order food and drinks by phone or tablet. The order will then be sent to the kitchen or bar where everything is going to be prepared.

1.2 Goal of the project

The goal of this project is to make the order process faster. If customers can order on their own device, they won't have to wait for a waiter, and the cooks will instantly receive the orders.

1.3 Scope and preconditions

Inside scope:	Outside scope:
1 4x front-end in React	1 Hosting
2 Back-end API in JAVA	2 Native-App
3 Relational database	3

Preconditions:
None, the company did not provide any specific preconditions for our project.

1.4 Strategy

We use scrum as our strategy, that means working in iterations, each iteration consists of two- to four-week sprints. (Peek, 2022). We are working in iterations, that consists of sprints, each sprint is three weeks. We want to update our project owners as much as we can, so they can see how the project is progressing. After every sprint we ask for feedback, so we can use that to make a planning for the next sprint.

1.5 Research questions and methodology

What is DOT Framework?
The DOT framework works by asking questions and answering them by using different methods to find a good and reliable answer.

What framework do we use for Java?
For our project we want to use a framework to make our coding process easier, so we decided to use spring boot.

Question 1: *Is spring boot the best option for our project?*
Method: *community research and available product analysis*
Answer:
Spring Boot framework helps:
- to reduce development, Unit Test and integration test time by providing some defaults.
- to increase productivity. (Posa, 2022)
"Spring boot provides opinionated 'starter' dependencies to simplify your build configuration", (Spring Boot, n.d.). For us this is helpful, so it configures Spring automatically and 3rd party libraries whenever possible. On the spring.io website it says that it is *"easy to build applications for developers." (Why Spring?, n.d.)*

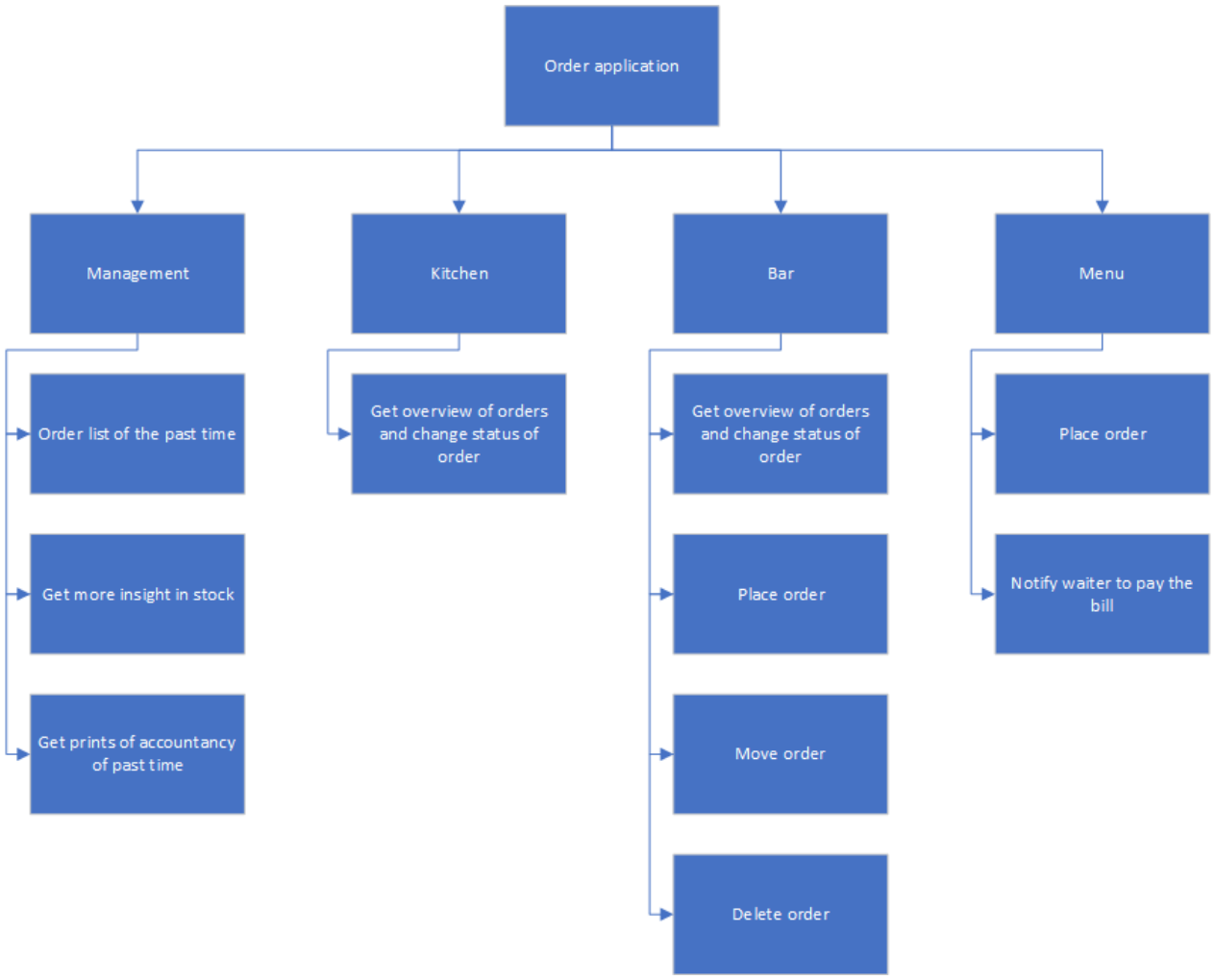
Question 2: *Which quality testing software are we going to use?*
Method: *Bad and good practices*
Answer:
We decided on four different quality testing software and compared on the pros and cons

Name	*Pros*	*Cons*
Jetbrains Qodana	- *Works with IntelliJ* [1]	- *Early access for Javascript* [2]
PVS-Studio	- *PVS-Studio can detect possible vulnerabilities*[3]	- PVS-Studio does not test quality on front-end languages[4]
Crucible	- *Works with our project planner software Jira*	- *Has no plugin for IntelliJ*
SonarQube	- *SonarQube can be used for Front-end and Backend*[5]	

[1] "Qodana is closely integrated with the IDEs you love. You can fix found issues directly in IntelliJ IDEA", (Qodana: The code quality platform for your favorite CI by, 2021) [Source]
[2] (Qodana for JS | Qodana, n.d.) [Source]
[3] "PVS-Studio is a SAST (Static Application Security Testing) tool, which means it can detect potential vulnerabilities classified under the CWE (Common Weakness Enumeration).", (Khrenov, 2018) [Source]
[4] (PVS-Studio Is a Solution to Enhance Code Quality, Security (SAST), and Safety, n.d.) [Source]
[5] (Code Quality and Code Security | SonarQube, n.d.) [Source]

1.6 **End products**

In the PBS that is drawn down here, shows which products our application will have at the end of the project. As shown, it exists of 4 different products which all have other tasks to fulfil.



2. Project organisation

2.1 Stakeholders and team members

Name	Abbreviation	Role and functions	Availability
Maarten van Dorp	*MD/DV*	*Developer*	*Monday and Tuesday, sometimes Wednesday.*
Britt Brink	*BB*	*Developer*	*Monday and Tuesday, sometimes Wednesday.*
Janine van Saaze	*JS/SV*	*Developer*	*Monday and Tuesday, sometimes Wednesday.*
Bart Lamers	*BL*	*Developer*	*Monday and Tuesday, sometimes Wednesday.*
Cas Goorman	*CG/GC*	*Developer*	*Monday and Tuesday, sometimes Wednesday.*
Jeroen Uijtenbosch	*JU*	*Stakeholder/Product Owner*	*On request*
Sander Klijsen	*SK*	*Stakeholder/Product Owner*	*On request*
Samuil Angelov	*SA*	*Stakeholder*	*Monday and Wednesday*

2.2 Communication

For our project we communicate with our PO's by mail or Teams, we thought this would be a smarter idea because we otherwise must travel for two hours. We want to contact them weekly. After every three weeks we give them a demo with the progression of our application. All the feedback given after the demo will be noted in Jira.

3. Activities and time plan

3.1 Phases of the project

In this project we have five phases. Each phase is explained here:

1. **Initiation Phase**
   In this phase we create a vision of our project. We start with some schematics and asking for feedback from the PO to understand what is important for this project. We also assign roles to the group members like Scrum Master.

2. **Planning and Estimation Phase**
   The second phase of this project we are creating plans for the sprints. After each sprint we evaluate our progress and combine each input into our project. We use a backlog to keep track of our progress.

3. **Implementation Phase**
   In the Implementation phase we implement the sprint. We update the backlog during this phase. We also give each other updates and reviews of the work plans or possible concerns.

4. **Reviewing Phase**
   Each sprint we review each other's progress and discuss the things we must discuss. After reviewing we brainstorm on problems to solve them. As a group it's easier to solve the problems because we are with more people then solving it alone.

5. **Releasing Phase**
   In the last phase, we deliver the product to our PO's and after that we are going to have a retrospective meeting to analyse the performance of each person in our group.

3.2 Time plan and milestones

We decided to work in the Agile method, we will set values and principles and based on these beliefs we will make decisions about how to do the work of developing software.

Each sprint will be 3 weeks, we chose three weeks to have enough time to make progression and process the given feedback by the PO’s. After each sprint we give a demo. We have a scrum master that leads the group. Every morning we have a stand-up to discuss the planning of the day, we close the day by showing the progression and hold backs. Below you can see in the table the start and finish date of our sprints, how they will look like:

Sprints	Start date	Finish date
Sprint 1	12-09-2022	05-10-2022
Sprint 2	05-10-2022	02-11-2022
Sprint 3	02-11-2022	23-11-2022
Sprint 4	23-11-2022	07-12-2022
Sprint 5	07-12-2022	11-01-2023

4. Testing strategy and configuration management

4.1 Testing strategy

In our project we choose for unit-testing. Because we can make specific test per user-story. Al these “units” needs to fully function before merging with the rest of the project.

Justification: We test every user story to make sure that every story can be finished properly, otherwise the application will break.

Conclusion: We are going to use SonarQube because in our DOT framework research it had the least cons and it is open source.

4.2 Test environment and required resources

Our project is mainly tested locally on our computers. Every branch for a specific feature will be tested locally and if it works it will be merged with the staging branch. When the staging branch works locally, were going to test the code in our live environment. For each branch with a specific feature, we will make unit-test for that feature. This Unit-test will be automatically checked in the pipeline. If a test fails, the branch will not be merged.

For our live environment we make use Buddy.Works for the CI/CD pipeline.

With Buddy.Works we have a “faster and easier alternative to CircleCI”. (Buddy, n.d.) [Source]

4.3 Configuration management

Our project uses version control via a shared GIT repository. For each Epic we make a milestone and we make use of different branches. Each feature / user story gets its own branch that makes it easier to track down specific bugs. We have a “main” and a “staging” branch, the staging is used to merge all the user stories in one branch to test if the whole project is still working as supposed to. For the “main” branch we use CI/CD so we can test each feature immediately in the live environment.

Before pushes and mergers to the main and/or staging branch the code must be checked by at least two members of the group.

5. Risk

5.1 Risk and mitigation

Risk	Prevention activities	Mitigation activities
1 People using the QR link from outside the restaurant	Ask for a code that changes every day before you can make an order	Only make it work when using the restaurants network.
2 App crashes when orders are made at the same time	Make a queue for orders.	Make a fallback error, send log to administration. Start app again.
3 The kitchen staff accidentally set food on “done”	Get notification before being set to ‘done’	Give option to set the food back to another status

Library:

Qodana: The code quality platform for your favorite CI by. (2021, 21 oktober). JetBrains. Geraadpleegd op 26 september 2022, van https://www.jetbrains.com/qodana/

Khrenov, S. (2018, November 26). *Everything You Wanted to Know about PVS-Studio and Dared to Ask*. PVS-Studio. Retrieved September 27, 2022, from https://pvs-studio.com/en/blog/posts/0593/

Qodana for JS | Qodana. (n.d.). Qodana Help. Retrieved September 27, 2022, from https://www.jetbrains.com/help/qodana/qodana-js.html

Posa, R. (2022, August 3). *Spring Boot Tutorial*. DigitalOcean Community. Retrieved September 27, 2022, from https://www.digitalocean.com/community/tutorials/spring-boot-tutorial

Spring Boot. (n.d.). Spring. Retrieved September 27, 2022, from https://spring.io/projects/spring-boot#overview

Why Spring? (n.d.). Spring. Retrieved September 27, 2022, from https://spring.io/why-spring

PVS-Studio is a solution to enhance code quality, security (SAST), and safety. (n.d.-b). PVS-Studio. Retrieved September 27, 2022, from https://pvs-studio.com/en/pvs-studio/

Code Quality and Code Security | SonarQube. (n.d.). Retrieved September 27, 2022, from https://www.sonarqube.org/

Buddy. (n.d.). *Smarter CircleCI alternative for dynamic teams | Buddy CI/CD*. Buddy: The DevOps Automation Platform. Retrieved September 27, 2022, from https://buddy.works/compare/circleci-alternative