Project Plan

Key Watch

BAS World

Date	**16/09/2024**
Version	**0.2**
State	**Draft**
Author	**Nameless**

Version history

Version	Date	Author(s)	Changes	State
0.1	09/09/2024	Nameless	First draft	Complete
0.2	16/09/2024	Nameless	Second draft	Complete

Distribution

Version	Date	Receivers

Contents

1.   Project assignment

1.1   Context

For this project, the client is "BAS World," a leading company that specializes in buying and selling a wide range of commercial vehicles. Their inventory spans various categories, including trucks, machinery, containers, and much more.

With such a thriving business they want to analyze their customers behaviors on their website and process the hundreds to thousands of daily events into a readable dashboard. The team will collect this data, process it, store it and display meaningful analytics on a dashboard for the client to see and analyze.

1.2   Goal of the project

This project aims to develop a real-time event dashboard to monitor, process, and display events that capture important actions performed by visitors. This dashboard will offer insights into user interactions on the website, enabling BAS World to track and analyze event data in real time. There will be a simulated or stub system to generate and post events to a queue.

1.3   Scope and preconditions

| Inside scope:	Outside scope:
1 Create API or service that receives events	1 Deployment with AWS
2 A back-end system that stores and organizes data	2 Authorization and authentication
3 Front-End dashboard to displays event data in real time	3 Reactive display regarding different devices (primarily desktop)
4 Event Data Simulation	4 Modification of data

1.4   Strategy

This project will be following the Agile approach using the Scrum method. This means the project will work in short-term iterations known as sprints and have regular reviews and planning meetings. The focus of the Agile methodology is on continuous improvement and delivering functional increments of the product over time. This approach will break the project into manageable pieces and prioritize critical features. At the end of each sprint, there will be a demonstration of the project's progress to the Product Owner (PO).

1.5 Research questions and methodology

- JSON Schemas & storing JSON data in databases?
- Useful data and metrics to display in a dashboard?
- Popular dashboard designs, user features they include, etc.

1.6 End products
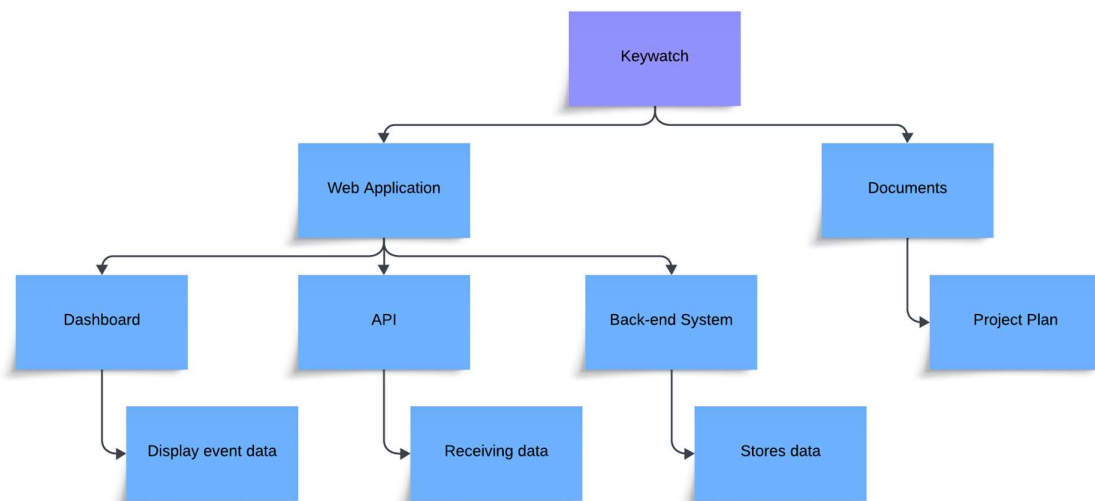
2. Project organisation

2.1 Stakeholders and team members

Name	Abbreviation	Role and functions	Availability
Rob Eijgelshoven rob.eijgelshoven@basworld.com	R.E	Role: Product Owner Function: Engineering Manager	Mail: Every Business Day Other: Every End of Sprint
Raja Gorentla r.gorentla@fontys.nl	R.G.	Role: Tutor Function: Group project teacher	Mondays from 9-12 Tuesdays from 13-16
Tymofii Andreiev	T.A.	Role: Developer Function: Student	Every Business Day from 9-16
Nuno Dias	N.D	Role: Developer Function: Student	Every Business Day from 9-16
Szymon Gancarz	S.G.	Role: Developer Function: Student	Every Business Day from 9-16
Duy Nguyen	D.N	Role: Developer Function: Student	Every Business Day from 9-16
Dick van de Meulenhof	D.M.	Role: Developer Function: Student	Every Business Day from 9-16
Addi Beenen	A.B.	Role: Developer Function: Student	Every Business Day from 9-16

2.2 Communication

R. Eijgelshoven

After each sprint, a feedback and question meeting will be arranged, to be held at the university.

R. Gorentla

After each week, a meeting with the tutor will be scheduled, taking place at the university.

Group Communication

Communication will be conducted online through Discord and in-person at the university, depending on the day.

3. Activities and time plan

3.1 Phases of the project

Phase 1 –Setup & Planning:
The first phase of any project is Setup and Planning. It involves creating the first draft of a Project Plan as well as setting up the basic infrastructure to start working as a group, notably a version control system like Git and a task scheduler like Jira. Finally, when working in AGILE methodology creating a backlog of tasks and assigning some to the first sprint.

Phase 2 –Documentation:
In this phase, the rest of the documentation will be completed. This includes a UML, coding guidelines, code comments, user stories and an architecture document. As well as revising old documents to keep them up-to-date and modify them according to any feedback from the client.

Phase 3 –Research:
In this phase, research is done to get more information about the topics and technologies necessary to complete the project. Research questions are made, and these get researched and documented so that the project can be finished successfully.

Phase 4 –Backend:
The next phase of the project is going to be backend – arguably, the most important part of the project. The plan for this phase is to start with event handlers to get the events. Then, database for saving all the data present in events and the system to transform all the data from database to logic are going to be created. Finally, everything is going to be covered with unit testing for preventing any issues happening in the future

Phase 5 –Frontend:
After the plan and the backend were finished, it is time to start showing up the events on the screen. For this, the plan is pretty simple: the first (optional) step is to create a prototype so there is a visual representation of what should be on the website. Then, the structure of the website will be created with HTML and CSS. Finally the JavaScript for talking with the Backend will be added.

Phase 6 –Testing:
This phase involves testing the project for bugs, usability and user friendliness. Because the plan is to use CI/CD there will be continuous testing for bugs and updates that break the application. As for usability and user friendliness this will be tested manually after significant updates to these aspects are made. It is important to test these early so that fixes can be pushed before any real damage is made.

Phase 7 –Deployment:
This final development phase is all about deploying the WebApp to the real world and readying it for real users. A docker file for easy development is essential as are compatibility and robustness. This will also involve a sprint to find and fix as many bugs as possible.

Phase 8 –Delivery:

The Delivery phase is meant to round everything up and handing the final product over to the client. This should be done with a presentation. All files will be laid out, bundled and handed over in a intuitive way.

3.2   Time plan and milestones

Sprint A:
- **Activities:**
  1. Form project and define roles
  2. Initial meeting with the client to understand requirements.
  3. Create a project name and logo.
  4. Create the project plan
  5. Create Jira/Trello to organize work
  6. Start on UML
  7. Start to work on Backend

Milestone: Sprint delivery

Sprint B:
- **Activities:**
  1. Formulate research questions
  2. Work on architecture document
  3. Proceed on UML
  4. Continue to work on Backend
  5. Start to work on Frontend

Milestone: Sprint delivery

Sprint C:
- **Activities:**
  1. Continue to work on Backend and Frontend
  2. Start on CI/CD
  3. Create cultural awareness report

Milestone: Sprint delivery

Sprint D:
- **Activities:**
  1. Continue to work on Backend and Frontend
  2. Create more test with CI/CD

Milestone: Sprint delivery

Sprint E:

- **Activities:**
    1. Make presentation
    2. Put a webapp into docker
    3. Test everything
    4. Record a video

Milestone: Sprint delivery

Phasing	Start date	Finish date
1 Sprint A	01-09-2024	11-10-2024
2 Sprint B	11-10-2024	08-11-2024
3 Sprint C	08-11-2024	29-11-2024
4 Sprint D	29-11-2024	20-12-2024
5 Sprint E	20-12-2024	17-01-2025

4. Testing strategy and configuration management

4.1 Testing strategy

<<Which testing strategy do you envision? E.g., on which levels will testing take place? Consider that you could choose unit, component, integration, system, or acceptance testing.

Justify your strategy, and also set goals where relevant. E.g., percentage code coverage for the relevant unit tests. For each of the planned tests, indicate what will be automated and what not.

Also think of quality testing setups like, e.g., Sonarqube.
>>

The testing strategy for **Key watch** will focus on ensuring that the system is both functional and reliable. Testing will be performed at multiple levels to ensure the quality and correctness of the system. The strategy includes:

1. **Unit Testing**:
   o **Objective**: Ensure that individual components, such as functions and methods, work correctly in isolation.
2. **Acceptance Testing**:
   o **Objective**: Ensure that the application meets the requirements and expectations of stakeholders. The acceptance criteria will be based on functional specifications.

4.2 Test environment and required resources

<< Describe the test environment. E.g., do you envision a DTAP (Development, Testing, Acceptance, Production) environment. Can you make use of a CI/CD environment, or will you develop your own?

It often helps to use a picture to visualize the test environment.

If you already know, describe which resources are required for realization and testing. Think of hardware, cloud environments and specific tooling required for development and testing.
>>

4.3 Configuration management

<< Describe the project approach with respect to version management (e.g. your GIT repository). This might include things like tooling, branching strategy, promotion-, release- and baseline strategy.

Also, when relevant, think of a mechanism to deal with change requests and problem reports.>>

Key watch will utilize a Git-based version control system. The approach to configuration management includes:
- **Branching Strategy**:
  o A **main branch** will be used for stable releases.
  o A **development branch** for ongoing work and feature additions.
  o Feature-specific branches for new development, which will be merged into the development branch after passing automated tests.
- **Version Control**:
  o Each feature release will be tagged in Git, with detailed release notes outlining new features, fixes, and known issues.

5. Risk and mitigation

Risk	Prevention activities	Mitigation activities
1 Loss of contact with Mentor	Frequent updates so time between loss of contact is minimal. Discuss short but also long-term plans with Mentor	Updates over email and with an alternative guide (teacher)
2 Sickness	Even distribution of work between members	Online work and redistribution of tasks between other members
3 Data Loss	Version Management with Git and cloud backup with Gitlab on Fontys Servers	Checking for latest version on members computers and re-merging
4 Expectations different from result	Frequent updates and sprint-end meetings to align/re-align goals and expectations with client	Reorganizing priorities and planning
5 Scope Creep	Frequent internal meetings (retrospective on sprints) to assess the feasibility of the project, timelines, and goals. MOSCOW & Phasing that prioritizes critical parts of project first	Reducing the scope on less critical parts of the project
6 Time Loss	Phasing that prioritizes critical parts of project first Thorough planning with some breathing room for bug fixing, testing and other problems that can be relied on if needed	Reorganizing priorities and planning