**Assignment-1: Project Plan**

**Note**

This file serves as a template.

You can add/remove/change its contents based on the characteristics of your project.

**Team**

Aniket Singh (Project Manager)

David Bouchedid (Engineer-I)

Nirajan Acharya (Engineer-II)

Jacob Steffen (Engineer-II)

Saravana Mauree (Customer Service Representative)

**Company name:** xxxx (Your team works like a startup compony, so give it a cool name).

**Software product name:** PenguinHealth

**Introduction**

This product is meant to be used by Youngstown State University’s students, faculty, and staff.

This software will be used as a daily health assessment tool. This tool is specifically designed for the pandemic we’re living in. The device uses the temperature of the body and questionnaires that ask about their recent activity, such as going to a bar, gym, grocery store. This questionnaire will also include some questions to see if the user feels any symptom of the COVID-19. This process will help determine if the user should be quarantined or isolated to protect themselves, their families, and the community.

The data will later be compiled to produce meaningful data, which can help students be aware of the number of people being quarantined or isolated. Compilation of data will help determine the population at high risk and low risk at the university.

This app will be a web-based application that can be accessed on any device, and it will be a dynamic web application. The software will be free to use and will be available to every student. Currently, YSU does not have any similar health assessment application or a web application to show the active cases. The application might include contact tracing features.

**Development model and method**

As a group, we decided that we would use a plan-driven method for our software engineering project. The model of software development would more likely be a combination of plan-driven methods and the agile approach. As a group, we have planned to start working on the technical requirements required for project development. We are planning to be more specific while creating the application. Repeatability, predictability, defined incremental process, extensive documentation, verification, validation, risk management, controlling and education, and monitoring are the aspects we will incorporate in during the developmental phases of our project. Methodologies like Personal software process, team software process, and united software process would help us to manage and organize the project in a good manner. As the title itself clearly explains the plan-driven methodology, so we will have to through methods with a well-made plan. We will have to break our project into a certain number of tasks. After that, each job is broken down into its composite activities. It would allow us to estimate total effort. This breakdown of the project could be analyzed for concurrencies and dependencies. As a result, our team size and composition would be great for the project. We will monitor our work throughout the development. That's why we have chosen plan-driven methodologies for our project.

**Phase-1: Requirements analysis**

1. Requirement elicitation:

You can do either a closed interview (with questionnaires), or an open interview (with free discussions).

For example, one student acts like a developer while other students act like customers.

1. Requirement specification:

After the interviews, you should write down the requirements in a formal document with tables or figures.

An example is given below.

1. Requirement validation:

You should verify that (1) there is no requirement conflict; (2) all requirements can be implemented.

1. Requirement change:

You may want to have a backup plan to deal with potential requirement changes.

Example: Requirements specifications in a tabular format.

|  |  |  |
| --- | --- | --- |
| **System requirements** | **Descriptions** | **Potential changes** |
| Hardware | Computers, servers, cloud hosts, cameras, I/O devices, embedded systems, etc. |  |
| Software | OSs, web browsers, graphic tools, databases, etc. |  |
| Network | High speed connection, routers, Wi-Fi, satellites, etc. |  |
| Others |  |  |
| **User requirements** | **Descriptions** | **Potential changes** |
| Inputs | Data is entered via keyboard, GUI (menu), files, tapes, cameras, network upload/download, etc. |  |
| Outputs | Output data types: numbers, strings, plots, graphics, etc.  Output formats: lines, records, tables, html, images, etc.  Output devices: terminal, web browser, smart phone, etc. |  |
| System response | A real-time system (show results immediately).  A delayed/offline system (show results later). |  |
| Easy-to-use | Do users need a special training to run the software? |  |
| Error handling | Users enter wrong data, users hit wrong keys or buttons, users close the program by accident (save or not save the results), etc. |  |
| Unwanted features | System crash, corrupting user data, incorrect computing result, a system hang or freeze, no help information, no error message, etc. |  |
| Others |  |  |

**Phase-2: Architecture design**

You **MUST** provide a diagram of the architecture design (see good and bad examples below).

A relatively rough design is acceptable, because you can add more details in Assignment-2.

It is a good idea to find and study the architecture patterns (good examples) in your project field.

An architecture design shows the structure of a system, which is often presented as a graphical model.

For a product implemented with procedural languages (C/Fortran), a **process flowchart** is popular.

If a project uses OOP languages (C++/Java), a **class/object hierarchy** diagram is commonly used.

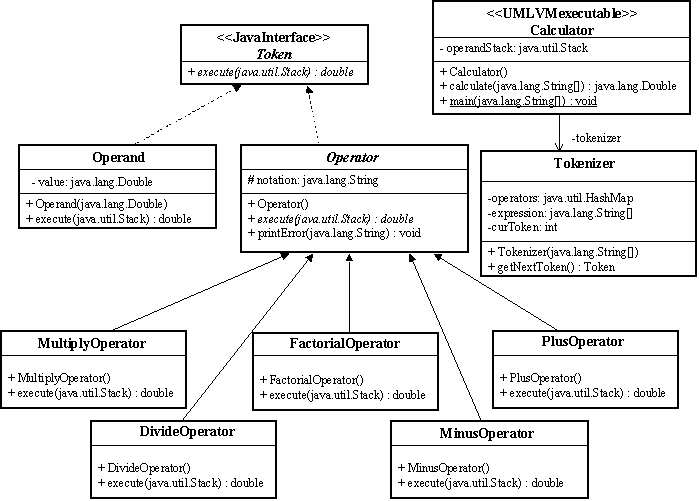
The above models are often displayed from either a static or a logical perspective.

If necessary, an architecture model can also be examined from a dynamic process view.

Example: The architecture model of a Calculator.

*Madhu, Manikya & Eadara, Babu & Malinowski, Adam & Suzuki, Junichi. (2020).*

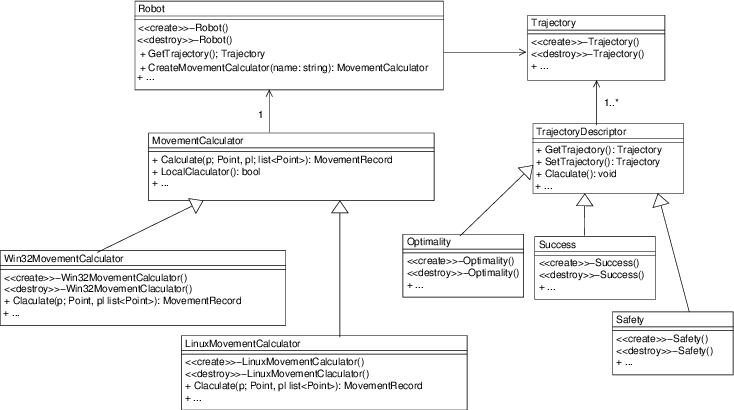
*Matilda: A Distributed UML Virtual Machine for Model-Driven Software Development.*



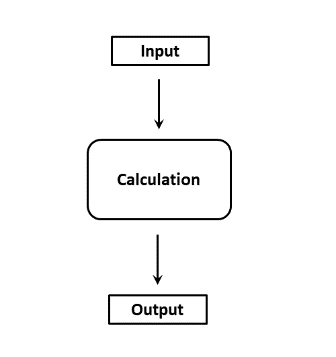
Example: The architecture model of a Robot simulator.

*Jimenez, J & Rano, Inaki & Minguez, Javier. (2006).*

*Advances in the Framework for Automatic Evaluation of Obstacle Avoidance Methods.*



A bad example: The architecture model of an online transaction system.



**Phase-3: Program design and implementation**

Provide a brief description of the program structure and coding strategy.

A relatively rough design/strategy is acceptable, because you can add more details in Assignment-2.

It is recommended to learn from the program design patterns (good examples) in your project field.

Basic activities in this phase:

1. Program design:

This is *similar to an architecture model*, but with more program information to facilitate coding work.

You can extend the architecture design model by adding language-specific materials (C/C++/Java/Python, …).

Architecture design: A model that shows the “big picture” or “skeleton” of a system.

Program design: A model that has more programming details about classes/interfaces/functions.

1. Identify classes/interfaces:

Write down the contents of all classes/interfaces (this is the **most important and difficult** part).

For example, class definition, data members/attributes, methods, special members (references or pointers), parameters, initial states, interactions among objects, inheritance (single or multiple inheritance), polymorphism, templates, abstract class, compositions, associations, etc.

1. Implementation:

Start to write code.

1. Reuse existing software:

Specify what programs and what types of license will be used, especially for proprietary products.

1. Version management:

If multiple versions will be delivered by multiple developers, you should have a version management plan.

(you can merge it with 1) in phase 5).

**Phase-4: Testing**

Provide a brief description of the software testing procedure (200-300 words).

A relatively simple procedure is acceptable, because you can add more details in Assignment-3.

1. Unit testing:

Describe how each function/class will be tested (manually or automatically).

For automatic testing, please specify the tool to be used and the test suites to be designed.

1. Component testing:

Describe how to test interface, especially function calls, parameter passing, message passing and object interactions.

1. System testing:

Describe how to test the entire system, focusing on the user inputs/outputs requirements and system crash/freeze.

1. User testing:

Describe how to do Alpha-testing, Beta-testing, and Acceptance-testing.

**Phase-5: Configuration and maintenance**

Provide a brief description of the configuration and maintenance plan (200-300 words).

1. Version management:

As a group, we have planned to use Github and Git for version control. Git provides impressive control over code and the versions of the project. All the programmers will make sure to commit changes with descriptive messages that will later help to merge project with other programmers or go back to the working version if something breaks. We are aware that we will run into merge conflicts, so we must try to work on different tasks and inform other members about changes to make sure that the merge conflict is manageable. System building:

1. Describe which build tool will be used (make, cmake, Code∷Blocks, NetBeans, MS Visual Studio, etc.).

ng build since we would be using Angular for the most part.

1. Maintenance:

Maintenance would be controlled through Git, working version will be published and new features/maintenance will be done in a different branch if required.

**Development tools**

Provide a short description of the tools and platforms to be used.

As of now, we have planned to use HTML5, CSS3, Angular, TypeScript, and if we add backend then we might use Django and Java.

We have planned to use our own editors/compilers that we feel comfortable to use.

For compilation of data to make meaningful graphs of the survey, we might use SAS or R.

**Project management**

You should **modify** the management timetable below, according to the contents of your project.

However, two **deadlines** must be met:

--- Week 15 or week 16 for project presentation.

--- Week 15 or week 16 for submitting the project report and codes.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Weeks** | | | | | | | | | | | | | |
| **Activities** | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| Choose a project | **x** |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Requirement analysis |  | **x** | **x** |  |  |  |  |  |  |  |  |  |  |  |
| Architecture design |  |  | **x** | **x** |  |  |  |  |  |  |  |  |  |  |
| Program design |  |  |  | **x** | **x** | **x** |  |  |  |  |  |  |  |  |
| Coding |  |  |  |  | **x** | **x** | **x** | **x** | **x** |  |  |  |  |  |
| Unit Testing |  |  |  |  |  | **x** | **x** | **x** | **x** | **x** | **x** |  |  |  |
| System Testing |  |  |  |  |  |  |  |  | **x** | **x** | **x** |  |  |  |
| Documentation |  |  |  |  |  |  |  |  |  |  | **x** | **x** |  |  |
| Project presentation |  |  |  |  |  |  |  |  |  |  |  |  | **x** | **x** |
| Submit report/codes |  |  |  |  |  |  |  |  |  |  |  |  | **x** | **x** |

**Work assignments**

You should **modify** the assignment table below, according to members’ skills and the project features.

All members should know their assignments in an activity during each development phase.

For example, in requirement analysis, the manager and customer representative play a major role (**M**),

*and other members should make great efforts to support the activity*.

In a startup company, everyone is expected to play multiple roles when needed.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Activities** | **Manager** | **Engineer-I** | **Engineer-II** | **Customer Represent.** |
| Requirement analysis | **M** |  |  | **M** |
| Architecture design | **M** | **M** |  |  |
| Program design |  | **M** |  |  |
| Coding |  | **M** | **M** |  |
| Unit Testing |  | **M** | **M** |  |
| System Testing |  |  | **M** |  |
| Documentation | **M** |  | **M** | **M** |
| Project presentation | **M** |  |  | **M** |
| Submit report/codes | **M** |  |  |  |