**THEORY-PRACTICE CORRESPONDENCE DOCUMENT.**

**FOR**

**AGILEPDS**

**Prepared By:**

Saurav Singh

Ronak Jain

Raaghav Mehta

Siddhant Misra

Shiv Bajpai

**NIIT UNIVERSITY**

* **Introduction**

**Key Concepts**

To define Software Engineering.

**Significance and Relevance**

Software Engineering defines how the internal modulation of how a software works and how it interacts with hardware.

**Real-World Contents**

Software Engineering helps the students to understand the making and designing of software.

**Inter-disciplinary Connections**

It links the software with hardware modules and allows the module to run across cross platforms.

**Critical Thinking**

Software Engineering is different from other respective studies as it helps in differentiating the design of software.

**Technology Tools and Techniques**

Involves learning about basic designing and structures of software.

**Plan Project Management**

Helps the project from the beginning to the end. Basic design of the project.

**Project Sketch**

We have integrated all the earlier mentioned requirements into the project to deliver better performance on both ends.

* **Introduction to Software Development Life-cycle**

**Key Concepts**

Requirements analysis, software design, coding, testing, maintenance, etc

We chose SDLC Model.

**Significance and Relevance**

A software development life cycle model is a descriptive and diagrammatic representation of the software life cycle.

**Real-World Contents**

Helpful in analyzing the order of tasks in project stages.

**Inter-disciplinary Connections**

It also captures the order in which software activities are to be performed.

**Critical Thinking**

Several models interface different types of procedures.

**Technology Tools and Techniques**

Life cycle model represents all the activities required to make a software product transit through its life cycle stages.

**Plan Project Management**

The models are suitable for development of technically challenging and difficult software products that are prone to various kinds of risks.

**Project Sketch**

We incorporated the Spiral Model into the project after researching and correspond to our methodology to the module.

* **Various Software Development Methodologies**

**Key Concepts**

Prototyping, interactive, waterfall model, enhancement, spiral model. Role of Management in software development. Role of metrics and measurement.

**Significance and Relevance**

A descriptive and diagrammatic representation of the software life cycle.

**Real-World Contents**

The entry and exit criteria for every phase. A phase can begin only if its phase-entry criteria has been satisfied.

**Inter-disciplinary Connections**

Without software life cycle model the entry and exit criteria for a phase cannot be recognized.

**Critical Thinking**

Develop and validate the next level of the product after resolving the identified risks.

**Technology Tools and Techniques**

Progressively more complete version of the software gets built with each iteration around the spiral.

**Plan Project Management**

The spiral model is perfect for development of technically challenging software products that are prone to several kinds of risks.

**Project Sketch**

We incorporated the Spiral Model into the project after researching and thinking out our methodology through to the module.

* **Software Requirement Specification**

**Key Concepts**

Problem analysis, requirement specification, validation, metrics, monitoring and control.

**Significance and Relevance**

The main objective of the SRS document is to describe the main requirements and engineering activities and to introduce techniques for requirements elicitation and analysis.

**Real-World Contents**

Without the SRS, the definition of the document is not complete.

**Inter-disciplinary Connections**

To describe requirements validation and to discuss the role of requirements management in support of other requirements engineering processes.

**Critical Thinking**

Helps the project developers to draw parallels between the requirements so that they can be connected.

**Technology Tools and Techniques**

Requirement Traceability Matrix and Development Matrices help defining relationships.

**Plan Project Management**

Helps to understand the project even closely and in a better form and the SRS helped the developers to design the app more efficiently.

**Project Sketch**

System Features that presented the complete module of the app/project and have been explained.

* **System Design**

**Key Concepts**

Problem partitioning, abstraction, top-down and bottom-up design, structured approach. Functional versus object-oriented approach, design specification and verification metrics, monitoring and control.

**Significance and Relevance**

It is a formal way of representing how a business system interacts with its environment and illustrates the activities that are performed by the users of the system.

**Real-World Contents**

The design phase documents define the way the software is designed.

**Inter-disciplinary Connections**

The design document works a cross between the requirement phase and the actual codes.

**Critical Thinking**

Design phase lays the foundation of how software actually gets designed.

**Technology Tools and Techniques**

The design phase requires the class diagrams, sequence diagrams and state diagrams.

**Plan Project Management**

After the design phase gets created, work starts upon the actual coding.

**Project Sketch**

The design document works as an empirical part of the software development.

* **Coding**

**Key Concepts**

Top-down and bottom-up, structured programming, information hiding, programming style, and internal documentation. Verification, Metrics, monitoring and control.

**Significance and Relevance**

The coding is the most intricate part of the software as it basically makes the software work.

**Real-World Contents**

Coding makes the software or app actually workable.

**Inter-disciplinary Connections**

The coding makes the software actually go cross platform.

Like using Bootstrap for a Dynamic responsive Design.

**Critical Thinking**

Working with the coding gives the complete idea of how each and every function works out.

**Technology Tools and Techniques**

The IDE that works on the code, scripts and frame works.

**Plan Project Management**

The codes get planned after the documentation on design and requirements and get completed before the testing.

**Project Sketch**

The codes sketch out the whole project all in all.

* **Testing**

**Key Concepts**

Levels of testing functional testing, structural testing, test plan, test cases specification, reliability assessment.

**Significance and Relevance**

Goal of testing is finding faults in the software and demonstrating that there are no faults in the software (for the test cases that has been used during testing)

**Real-World Contents**

It is not possible to *prove* that there are no faults in the software using testing

Testing should help locate errors, not just detect their presence.

**Inter-disciplinary Connections**

Testing the driving test cases automatically from a formal specification of the functional requirements

**Critical Thinking**

The number of test cases increase exponentially with the number of input/output variables

**Technology Tools and Techniques**

J-Unit, Appdynamics, WAP testing tools. We know that if we find an error during unit testing it is in the module we are testing

**Plan Project Management**

Testing clearly removes out all the errors from the software plan.

**Project Sketch**

Took out bugs from the project codes.

* **Software Project Management**

**Key Concepts**

Cost estimation; Project scheduling, Staffing, Software configuration management, Quality assurance, Project Monitoring, Risk management, etc.

**Significance and Relevance**

It is an agile process that allows us to focus on delivering the highest business value in the shortest time.

**Real-World Contents**

The business sets the priorities.

**Inter-disciplinary Connections**

It allows us to rapidly and repeatedly inspect actual working software (every two weeks to one month).

**Critical Thinking**

The project management is the most crucial part of the project that defines it.

**Technology Tools and Techniques**

Agile Scrum method, Continuous meetings and analysis.

**Plan Project Management**

Every two weeks to a month anyone can see real working software and decide to release it as is or continue to enhance for another iteration.

**Project Sketch**

We followed this module and did rigorous research every week to develop and enhance the project development process.