IT314: Software Engineering

Lab-01

Kirtika Basant
202101515

a. Model: waterfall model
Reason: since it's a simple model, we'll have the set of requirements well set prior to working on it as its mere task is to process.
Example project: an attendance analyzer that can provide teachers with the required information using the data (present-absent counts)

b. Model: throw-away prototyping model
Reason: here the people working on it are new to the system so they must conduct the project through various prototypes. Also, as this model will yield a better user-interfaced end product.

c. Model: spiral model if tech support available, else incremental waterfall model
Reason: here we're having some basic features that can be implemented first and since other features are related to/based on it, we can perform further iterations to add on all the other desired features.
Example project: an on-farm project which processes the basic environmental conditions such as temperature, moisture, etc on a spreadsheet and produces information like when and how much to irrigate the farm.

d. Model: spiral model
Reason: first up we have an in-house development team, so technically no issue will arise. Also, as the requirements aren't fixed, the spiral model would work best.

e. Model: incremental waterfall model
Reason: since there are a lot of desired features, some fixed, and some to be added we can't use the waterfall model. Since the "time to market" is critical, we'll use an incremental waterfall model, so that we can add all the desired features along with great release time.

f. Model: spiral model
Reason: since it'll be a system with lots of possible cases to cover, implying we'll need to update the requirement set, we'll use the spiral model. Also, it requires great visibility and technical support.

g. Model: synchronized and stability model or spiral model
Reason: in this case, we'll begin with a certain frozen set of requirements for the build-up of the software because we know the additional requirements will arise along the way, so we'll incorporate them as per the demand of the feature and its effect on current software's stability.

h. Model: waterfall model/ incremental spiral model
   Reason: we'll use the waterfall model if all the requirements are limited/frozen. But we'll have to use the spiral model if more features are to be added.

i. Model: evolutionary prototyping model
   Reason: we'll have the set requirements to determine this train timing feature, which will be used to develop the prototype but as new terminals/train are added, we'll need to incorporate those too, hence evolutionary prototyping.

j. Model: waterfall model
   Reason: here the defence system will need the system to be resistant to iterations, hence all the requirements will be mentioned beforehand. So we'll use the waterfall model.

k. Model: spiral model
   Reason: here we need that the prior features/setting aren't hampered while incorporating the new ones.

l. Model: waterfall model (or synchronize and stabilize model if future iterations are to be kept in consideration)
   Reason: here all the requirements will be well set and frozen.

m. Model: waterfall model
   Reason: since it's mentioned that no changes will be made.