Course code

Course Name

Phase 2: Software Design Document

Team Name

Full Name ID

Dec 2020

Phase 2
Project: <Project Name>

جامعة الإسكندرية
ALEXANDRIA UNIVERSITY
كلية العلوم

Software Design Document

Month & Year

Contents

Phase 2
Project: <Project Name>

جامعة الإسكندرية
ALEXANDRIA UNIVERSITY
كلية العلوم

Software Design Document

Instructions [To be removed]

- **IMPORTANT. Rename this document to TeamName-Topic-SDD.docx**
- **Remove the following notes and any red notes.**
- **This document is the template document for your SRS.**
- **For further guidelines and information, READ project details document (project decription).**
- **Figures included here are for GUIDANCE purpose. Do not copy them or imitate them. Use the notations taught in class or the best suitable notation for each design item**

Team

ID	Name	Email	Mobile
	1st name is team leader		

Document Purpose and Audience

- **Any document anywhere should tell us 2 things: (1) what this document is and (2) who is excepted to read it.**
- **Write in simple notes: What is this document?**
- **List the target audience to read this document (e.g. CEO? Project Manager? Customer...?)**

Phase 2
Project: <Project Name>

جامعة الإسكندرية
ALEXANDRIA UNIVERSITY
كلية العلوم

Software Design Document

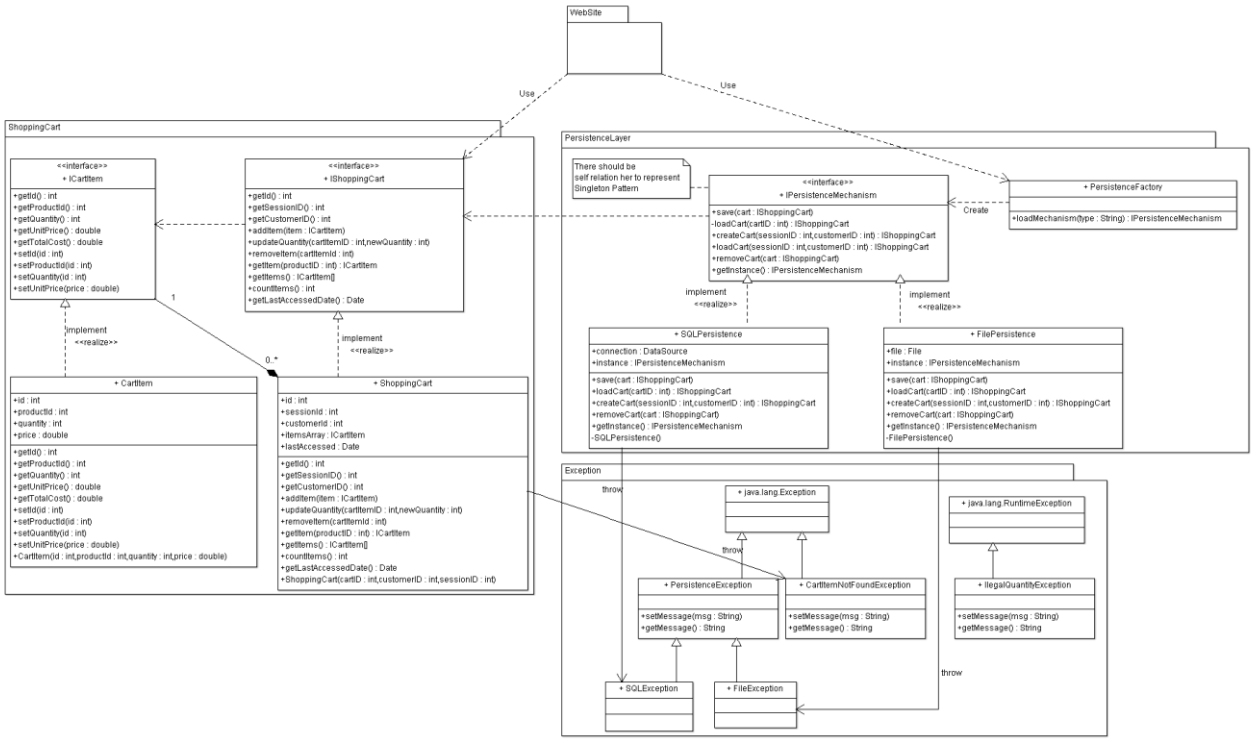
System Models

I. Class diagrams

- **You should provide your class diagram. In case on diagram is so complex, divide it to several ones of reasonable size or draw separate ones, each for one of the components on the system decomposition diagram.**
- **Class diagram is a static diagram and should not represent any dynamic flow of events.**
- **Put stereotypes of the classes to give more information. UML predefines some stereotypes like: <<interface>>, <<type>>, <<implementationClass>>, <<enumeration>>, etc. and you create your own also.**
- **Put Relationships between classes and the types of the relationships.**
- **Put multiplicity.**
- **Put relationship name (e.g. faculty "offer" course).**
- **Put attributes in the classes.**
- **Put functions & Put parameters.**
- **Put data types of each attributes and the parameters.**
- **Make sure to include all domain (entity), boundary and control classes needed to implement the system.**
- **Following is Shopping Cart Component class diagram.**

Phase 2
Project: <Project Name>

Software Design Document



List down your classes and describe them

Class ID	Class Name	Subsystem ID	Description & Responsibility

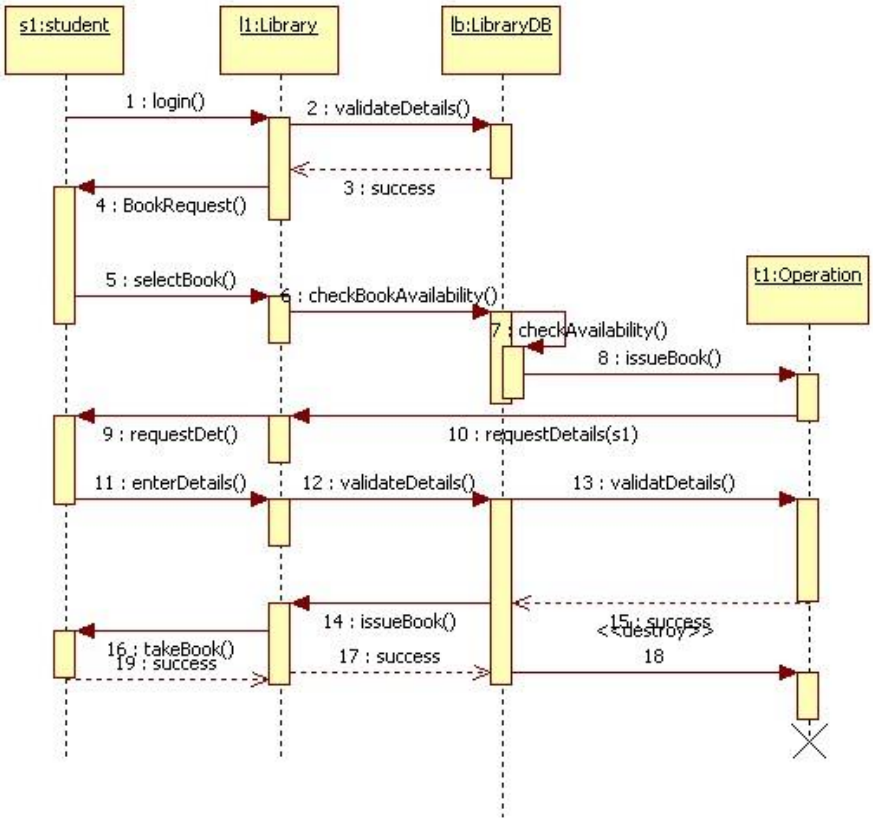
- **In the above table make sure that each class belongs to a subsystem.**
- **In the above table ALL classes should belong to subsystems. And each subsystem should at least contain one class.**

I. Sequence diagrams

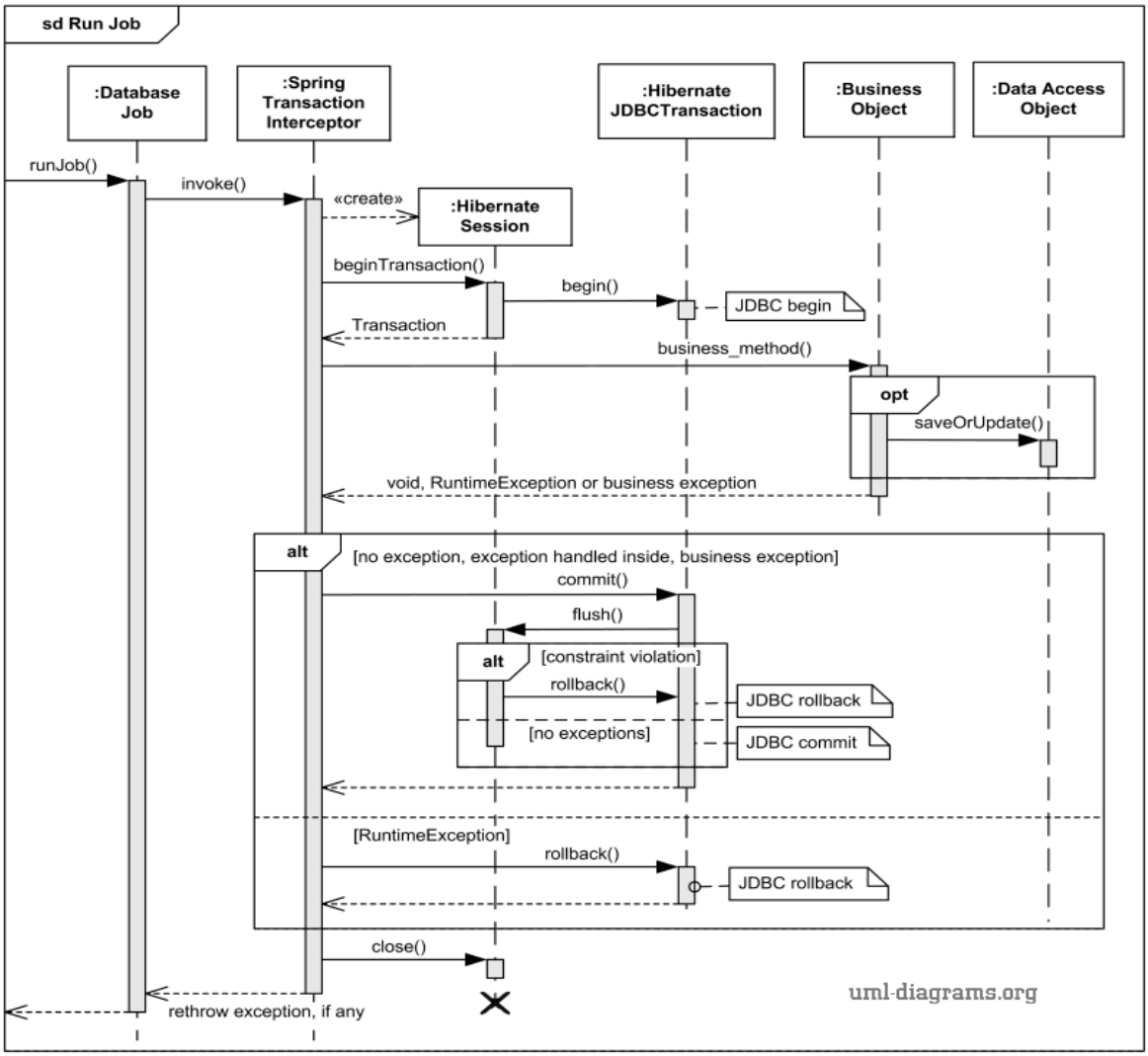
- **List Sequence diagrams for all requirements. Provide for each Sequence an ID.**
- **Usually each use case is represented by a sequence diagram or more.**
- **Overall, all the diagrams should represent all requirements and possible flows.**
- **Make sure that each object in the sequence diagram has a corresponding class in the class description table above. If not, it will be REJECTED.**
- **Put actual function calls with proper parameters and return types corresponding to class diagrams.**

Software Design Document

- **Following are couple of examples for small / meduim examples. We expect such diagrams, however there is a missing thing in them. Most of calls don't have parameters. Please always specify the parameters in the call, matching the class diagram.**



Software Design Document



Class - Sequence Usage Table

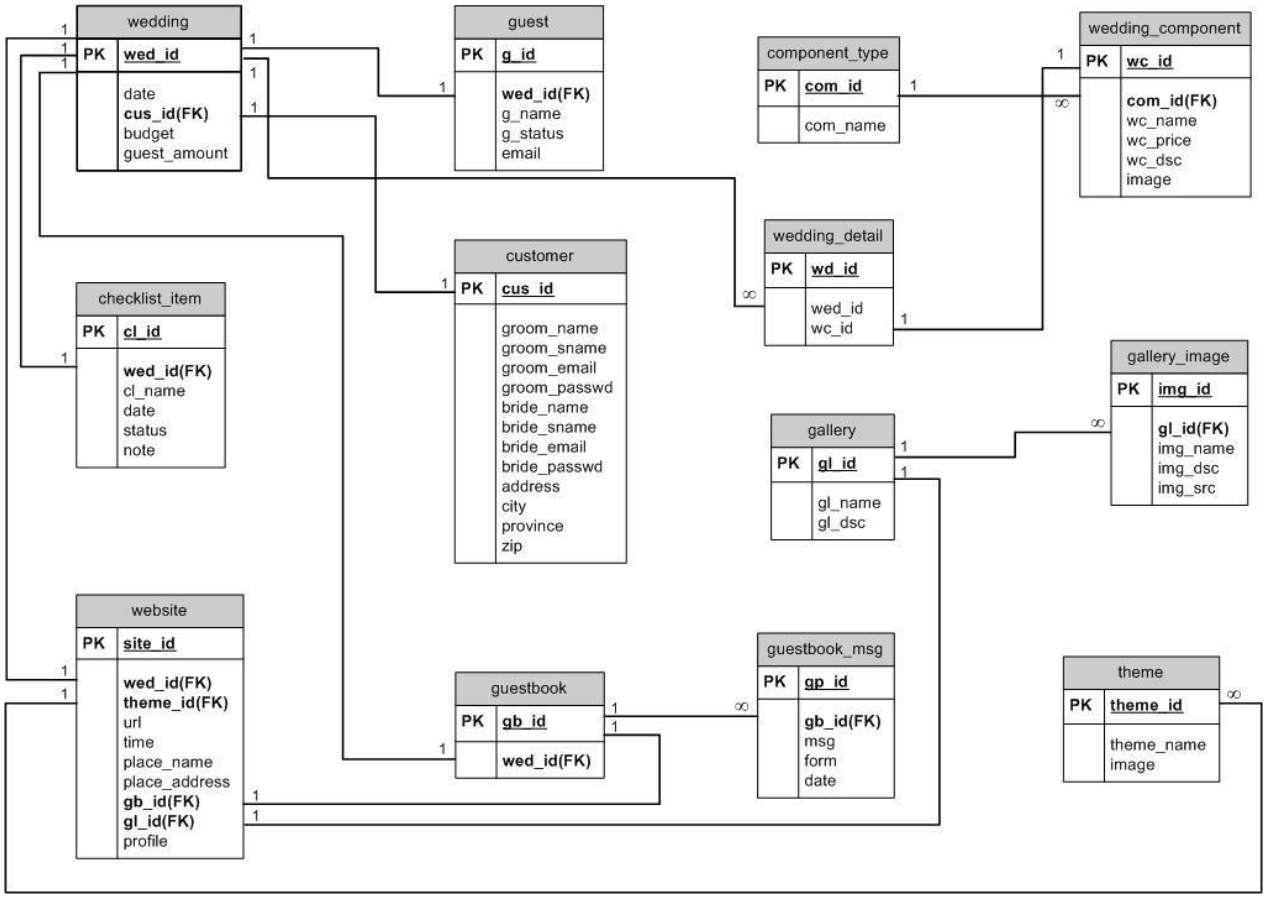
- **In this table, we will list EVERY class in class diagram and which sequences used this class diagram. This helps in avoiding either unused classes or extra classes appears in sequence diagrams. In "Overall used methods" section, put all functions appeared in all sequences.**

Class Name	Sequence Diagrams	Overall used methods
E.g. Employee	1, 3, 5 (means Seq Ids 1, 3, 5 used Employee class)	Save, GetData

Software Design Document

III. Physical Entity-Relationship Diagram

- **Provide the ERD Diagram**
- **All entity classes on the class diagram that need to be stored permanently, convert them and their relationships into ERD diagram.**
- **Don't list any boundary or control entities!**
- **Use a suitable notation for representing ER diagram, e.g., the one proposed on the book or an alternative version. Build it using a tool.**
- **Following is an example of ER diagram, using different notation.**

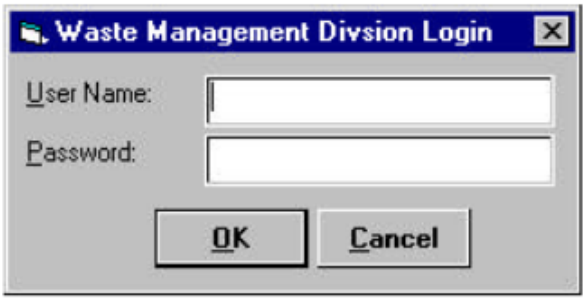


IV. User Interface Design

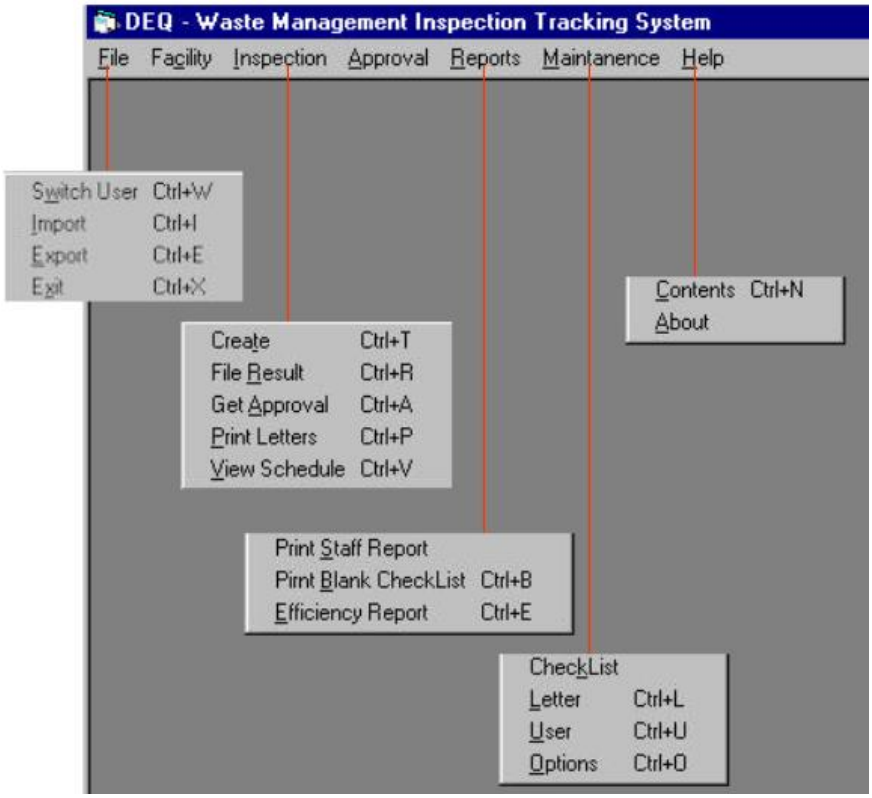
- **Use a prototyping tool like https://app.moqups.com, http://Infragistics.com or NinjaMock or using a GUI builder (like the one in NetBeans) to build your interface.**
- **Develop a prototype for each screen / page that your application will have and relate them to each other showing which one leads to which one.**

Software Design Document

- **For each screen specify the buttons, menus, etc. that will be on it and their functions.**
- **An example is shown below.**
- **Screen 1 – Login Screen (example)**



- **Screen 2 – Main Interface (example)**



- **Navigation tree:** **Login Screen** → **Main Screen**

Phase 2
Project: <Project Name>

جامعة الإسكندرية
ALEXANDRIA UNIVERSITY
كلية العلوم

Software Design Document

V. Dataflow diagram (DFD)

- **Provide the DFD**

VI. Algorithms and Data Structures

- **Specify what algorithms you need in order to build the application. If it is an existing one, just refer to it. If it is one you will develop, then write in detail in mathematical notation, pseudo code, or as a flowchart. Example of such algorithms:**
  - **The steps for calculating if there is winner in a two-player game.**
  - **The steps for calculating the salary in a payroll program.**
  - **The algorithm for deciding which posts to show first in a social network application.**
- **Specify which data structures (DS) you will use to store which data in memory, other than regular arrays and array lists. Justify your choice and explain the reasons behind it.**
- **In the rare occasion that no existing data structures supports your requirements and you need to create a new one or implement a non-implemented one, include the design of this new DS.**

Ownership Report

- **Remove the following notes and any red notes**
- **For every item in this document, write the owners. If someone is owner of something, s/he understands it 100.%**
- **Team leader must verify the table with the team members.**

Item	Owners

Policy Regarding Plagiarism:

Students have collective ownership and responsibility of their project. Any violation of academic honesty will have severe consequences and punishment for ALL team members.

1. تشجع الكلية على مناقشة الأفكار و تبادل المعلومات و مناقشات الطلاب حيث يعتبر هذا جوهريا لعملية تعليمية سليمة
2. ساعد زملاءك على قدر ما تستطيع و حل لهم مشاكلهم فى الكود و لكن تبادل الحلول غير مقبول و يعتبر غشا.
3. أى حل يتشابه مع أى حل آخر بدرجة تقطع بأنهما منقولان من نفس المصدر سيعتبر أن صاحبيهما قد قاما بالغش.
4. قد توجد على النت برامج مشابهة لما نكتبه هنا أى نسخ من على النت يعتبر غشا يحاسب عليه صاحبه.
5. إذا لم تكن متأكدا أن فعلا ما يعد غشا فلتسأل المعيد أو أستاذ المادة.
6. فى حالة ثبوت الغش سيأخذ الطالب سالب درجة المسألة ، و فى حالة تكرار الغش سيرسب الطالب فى المقرر.