

**CS251 – Software Engineering I**

|  |  |
| --- | --- |
| **Staff Memebers** | **Email** |
| **Dr Mohamed El Ramly** | **m.elramly@fci-cu.edu.eg** |
| Eng Catherine Bedrossian | c.bedrossian@fci-cu.edu.eg |
| Eng Sara Tarek Ali Othman | s.tarek@fci-cu.edu.eg |
| Eng Mohamed Samir | m.samir@fci-cu.edu.eg |
| Eng Marwa Nabil | m.nabil@fci-cu.edu.eg |
| Eng Yomna Magdy Mohamed | yomna@fci-cu.edu.eg |
| Eng Desoky Abd El-qawy | d.abdelqawy@fci-cu.edu.eg |
| Eng Omar Khaled Ali Ragab | o.khaled@fci-cu.edu.eg |
| Eng Mohamed Elarnaoty | m.elarnaoty@fci-cu.edu.eg |
| Eng Mostafa Saad | m.saad@fci-cu.edu.eg |

Contents

[Instructions [To be removed] 3](#_Toc402855162)

[Team 3](#_Toc402855163)

[Document Purpose and Audience 3](#_Toc402855164)

[System Models 3](#_Toc402855165)

[System Decomposition 3](#_Toc402855166)

[Class diagrams 6](#_Toc402855167)

[Important Algorithm 7](#_Toc402855168)

[Sequence diagrams 7](#_Toc402855169)

[Class - Sequence Usage 9](#_Toc402855170)

[Physical Entity-Relationship Diagram 10](#_Toc402855171)

[User Interface Design 10](#_Toc402855172)

[Screen 1 – XXX 11](#_Toc402855173)

[Screen 2 – YYY 11](#_Toc402855174)

[Screen 3 – Login Screen (example) 11](#_Toc402855175)

[Screen 4 – Main Interface (example) 11](#_Toc402855176)

[Ownership Report 12](#_Toc402855177)

[Policy Regarding Plagiarism: 13](#_Toc402855178)

[References 13](#_Toc402855179)

[Authors 13](#_Toc402855180)

# Instructions [To be removed]

* **IMPORTANT. Rename this document to CS251-TAName-LeaderID-SDDDocument.docx**

**(e.g. CS251-MostafaSaad-20040752-SDDDocument.docx)**

* **Remove the following notes and any red notes**
* **This document is the template document for your Software Design.**

# Team

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Email** | **Mobile** |
|  | 1st name is team leader |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

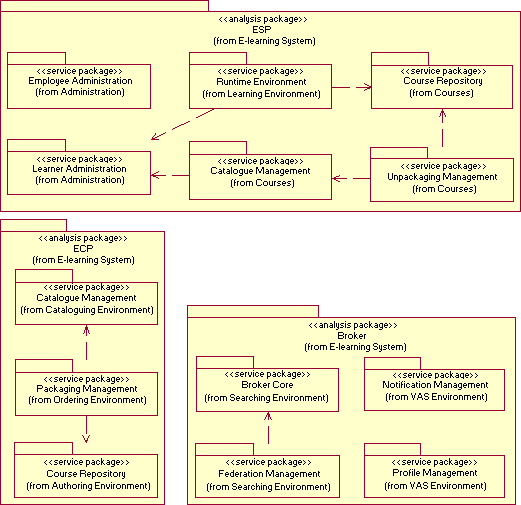
# Document Purpose and Audience

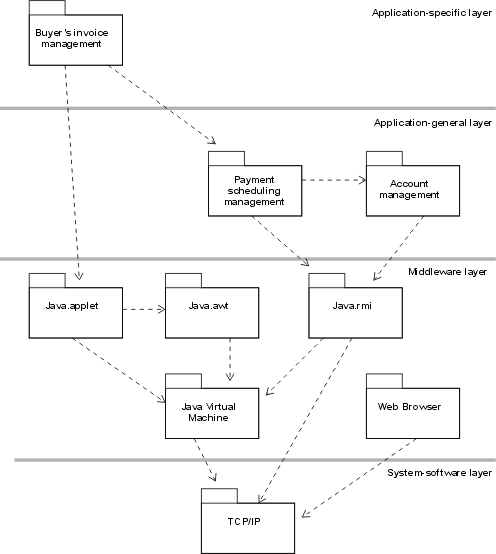
* **Any document anywyere should tell us 2 things: What is this document? Who is execpetd to read it?**
* **Write in simple notes, What is this documet?**
* **List the taget audience to read this document (e.g. CEO? Project Manager? Customer...?)**

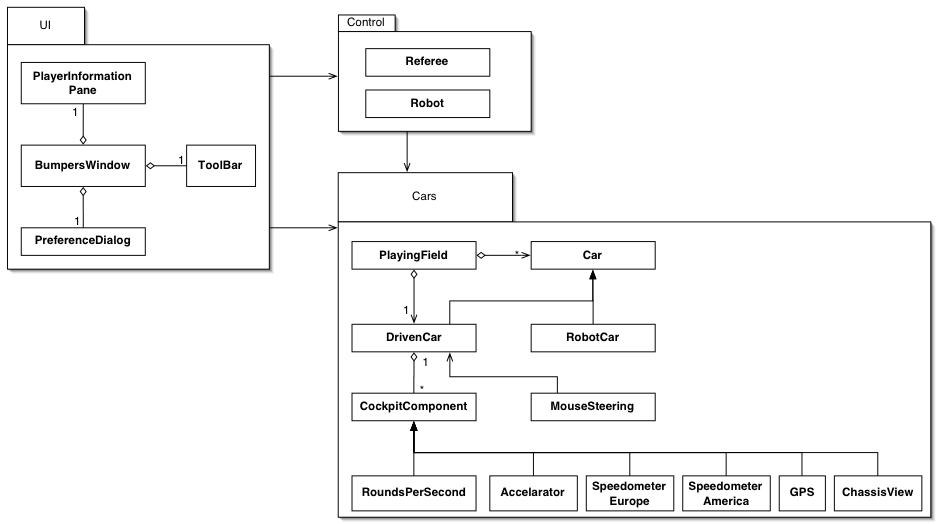
# System Models

## System Decomposition

* **Provide and describe a figure that depicts the overall system decomposition.**
* **Read about “Software Coupling and Cohesion” Conceps. Reflect that in model**
* **3 following images from web is for purpose of explaining. First one is how model is designed, but there are little connections (so it is not so good). Other ones has better connection. Consider the 3 to know how a good model should be.**







## Class diagrams

* **You should provide your class diagram. In case on diagram is so complex, divide it to several ones of reasonable size.**
* **Class diagram is a static diagram and should not represent any dynamic flow of events.**
* **Put stereotypes of the classes.**
* **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.**
* **Concentrate in the class diagram on control and entity classes. But, in the table below put all boundary, control and entity classes.**
* **Highly perfered: Each class has a corresponding interface**
  + **Let all objects parameters and returns be of interface type.**
  + **See Shopping Cart Case Study**
* **Following is Shopping Cart Component class diagram.**



**List down your classes and describe them**

| **Class ID** | **Class Name** | **Subsystem ID** | **Description** |
| --- | --- | --- | --- |
|  |  |  |  |

* **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.**

### Important Algorithm

* **If any method in a class is implementing an algorithm (complex enough), then you should describe it here.**

## Sequence diagrams

* **List Sequence diagrms for all requirments. Provide each Sequence an ID.**
* **Not neccessairly diagram per requiremnt. May be better to merge more than a flow. Typically, 1 sequnce diagram per your use case table.**
* **Overall of diagrams represents 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.**
* **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.**





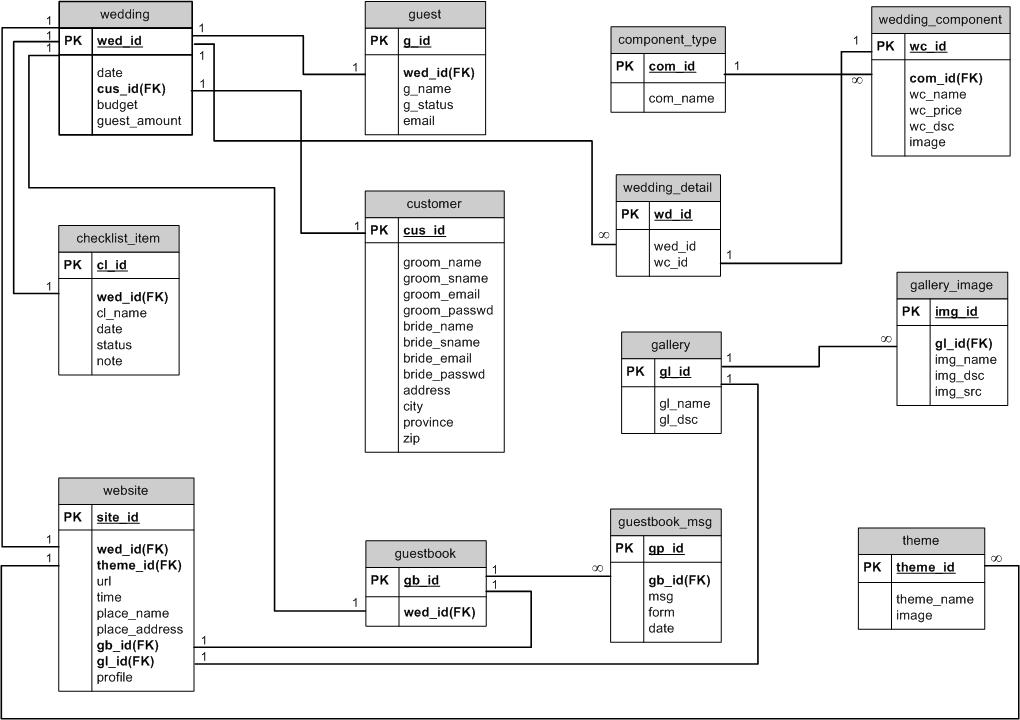
### Class - Sequence Usage

* **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. If this table was built in ignorance of actual class / sequence diagrams = REJECTED for whole document.**

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

## Physical Entity-Relationship Diagram

* **Provide the ERD Diagram**
* **convert all entity objects of the class diagram and their relationships into ERD**
* **Don’t list any boundry or control entities!**
* **Following is an example of ERD**



## User Interface Design

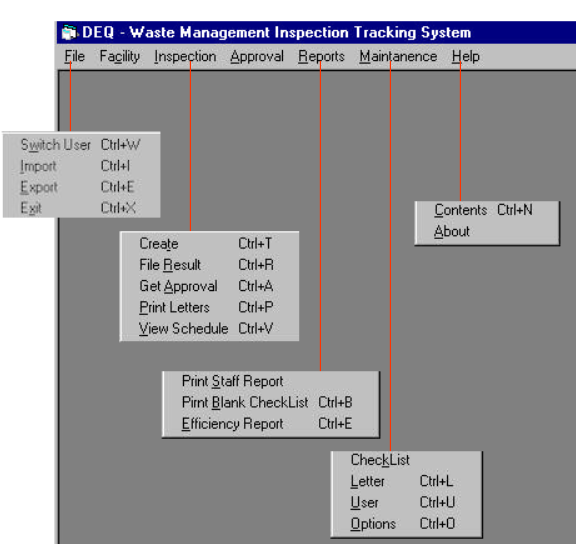
### Screen 1 – XXX

### Screen 2 – YYY

### Screen 3 – Login Screen (example)



### Screen 4 – Main Interface (example)



# 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.

# References

* http://www.mhhe.com/engcs/compsci/pressman/graphics/Pressman5sepa/common/cs1/design.pdf

# Authors

* Eng Mostafa Saad