|  |  |
| --- | --- |
| logo ngan.png | MINISTRY OF EDUCATION AND TRAINING |

|  |
| --- |
| **FPT UNIVERSITY** |
| Capstone Project Document |
| Build a Web-based application  that manages the activities of  delivery service system by coach |
|  |
| |  |  | | --- | --- | | **Group 18** | | | **Group Members** | Đào Bảo Long – Team Leader – SE60690  Lê Phúc Lữ – Team Member – 60563  Nguyễn Thanh Tùng – Team Member – SE60609  Nguyễn Tấn Đức – Team Member – SE60660 | | **Supervisor** | Nguyễn Trọng Tài | | **Ext Supervisor** | N/A | | **Capstone Project Code** | i-Deliver | |
| - Hồ Chí Minh City, May 2014 - |

*Acknowledgement*

To write these words for expressing our gratitude, we first say thanks to Mr. Nguyen Trong Tai about his help for all of us for more than three months duration. He always willing to answer our questions, guide to solve the problems not only at the two meeting days each week but also whenever we need him. We believe that each of time our team faced to difficulties gave caused of his concern. We also appreciate to his kindness and his trust to choose our team among two teams applied to the same his project at the beginning of the course.

After a long time of learning in FPT University, we have been taught by a lot of lectures. Each of them has a specific style but all give us the chances to gain the huge of precious knowledge for the future works and we will not forget their work. End of this duration, capstone project is really a milestone for our study journey. We all believe this is one of best subject we have learned and we also would like to say thanks to education departure of FU for set this subject with the real importance meanings and effects.

When receive the decision of forming our team, each member of our team do not know the others and we have not worked together in any project before that. It is quite hard at some first weeks for sharing the issues within team members. But after that, with the same objectives, we found out the solidarity spirit and everything became better. No doubts that our team not very good and we need to try day by day to finish our tasks for build the website for the release day.

In these days, we also receive the assistance from our friends and the encouragement from our family. Without these help, we may not get anything done and we have to say thanks to all of them as a duty.

And we also say thanks to Mr. Kieu Trong Khanh about his accept to the invitation to take part in our trial defense capstone project and gave us a lot of comments for improving our project. Our project may be contains some bugs but we do not afraid to find and fix them because this should be the main of software engineering daily work.

Group 18  
*Đào Bảo Long,*

*Lê Phúc Lữ,*

*Nguyễn Thanh Tùng,*

*Nguyễn Tấn Đức*

# Table of contents

[Table of contents 3](#_Toc388777408)

[Table of figures 11](#_Toc388777409)

[Terminology 13](#_Toc388777410)

[1. Introduction 14](#_Toc388777411)

[1.1. Introduction 14](#_Toc388777412)

[1.2. The initial idea of group 14](#_Toc388777413)

[1.3. Overview of existing methods 14](#_Toc388777414)

[1.1.1. Requests management 14](#_Toc388777415)

[1.1.2. Packages arrangement 14](#_Toc388777416)

[1.1.3. Packages management 14](#_Toc388777417)

[1.4. Limitations of existing system 15](#_Toc388777418)

[1.1.1. Requests management 15](#_Toc388777419)

[1.1.2. Packages management and arrangement 15](#_Toc388777420)

[1.1.3. Schedules and trips management 15](#_Toc388777421)

[1.5. Benefits of expected system 15](#_Toc388777422)

[1.6. Business outline 15](#_Toc388777423)

[1.7. Approaches 15](#_Toc388777424)

[1.8. Group of functions 16](#_Toc388777425)

[2. Software Project Management Plan (SPMP) 17](#_Toc388777426)

[2.1. Problem Definition 17](#_Toc388777428)

[2.1.1. Name of this Capstone Project 17](#_Toc388777431)

[2.1.2. Problem Abstract 17](#_Toc388777432)

[2.1.3. Project Overview 17](#_Toc388777433)

[2.2.3.1. The Current System 17](#_Toc388777440)

[2.2.3.2. The Proposed System 17](#_Toc388777441)

[2.2.3.3. Boundaries of the System 19](#_Toc388777442)

[2.2.3.4. Development Environment 19](#_Toc388777443)

[2.2. Project organization 20](#_Toc388777444)

[2.3.1. Software Process Model 20](#_Toc388777447)

[2.3.2. Roles and Responsibilities 20](#_Toc388777448)

[2.3.3. Tools and Techniques 21](#_Toc388777449)

[2.3. Project management plan 21](#_Toc388777450)

[2.4.1. Tasks 21](#_Toc388777452)

[2.2.1.1. Task 1: Initiating 22](#_Toc388777456)

[2.2.1.2. Task 2: Planning 22](#_Toc388777457)

[2.2.1.3. Task 3: Specifying requirements 22](#_Toc388777458)

[2.2.1.4. Task 4: Designing database 22](#_Toc388777459)

[2.2.1.5. Task 5: Creating Software Design Description 23](#_Toc388777460)

[2.2.1.6. Task 6: Implementing 23](#_Toc388777461)

[2.2.1.7. Task 7: Performing Testing 23](#_Toc388777462)

[2.2.1.8. Task 8: Writing User’s Manual 24](#_Toc388777463)

[2.2.1.9. Task 9: Deploying the Website 24](#_Toc388777464)

[2.2.1.10. Task 10: Finalizing and Closing 24](#_Toc388777465)

[2.4.2. Task sheet 24](#_Toc388777466)

[2.4. Coding Convention 30](#_Toc388777467)

[3. Software Requirement Specifications (SRS) 31](#_Toc388777470)

[3.1. User Requirement Specification 31](#_Toc388777472)

[3.1.1. Guest Requirements 31](#_Toc388777475)

[3.1.2. Logged User Requirements 31](#_Toc388777476)

[3.1.3. Customer Requirements 31](#_Toc388777477)

[3.1.4. System Admin Requirements 32](#_Toc388777478)

[3.1.5. Staff Requirements 32](#_Toc388777479)

[3.2. System Requirement Specification (Specific Requirements) 32](#_Toc388777480)

[3.2.1. External Interface Requirements 32](#_Toc388777482)

[3.2.1.1. User Interfaces 32](#_Toc388777483)

[3.2.1.2. Hardware Interfaces 32](#_Toc388777484)

[3.2.1.3. Software Interfaces 33](#_Toc388777485)

[3.2.2. Functional Requirements 33](#_Toc388777486)

[3.2.2.1. Overall use case 33](#_Toc388777487)

[3.2.2.2. « Guest » Register 34](#_Toc388777488)

[3.2.2.3. « Customer, Guest » Post request 35](#_Toc388777489)

[3.2.2.4. « Customer » Cancel request 37](#_Toc388777490)

[3.2.2.5. « Customer, Guest » View request detail 38](#_Toc388777491)

[3.2.2.6. « Customer, Guest » Tracking package 40](#_Toc388777492)

[3.2.2.7. « Customer » Payment 41](#_Toc388777493)

[3.2.2.8. « Customer » Rating 43](#_Toc388777494)

[3.2.2.9. « Customer » Comment 45](#_Toc388777495)

[3.2.2.10. « System administrator » Define delivery fee value 47](#_Toc388777496)

[3.2.2.11. « System administrator » Edit delivery fee value/ calculating formula 49](#_Toc388777497)

[3.2.2.12. « System administrator » Create new staff account 50](#_Toc388777498)

[3.2.2.13. « System administrator » Edit account information 51](#_Toc388777499)

[3.2.2.14. « System administrator » Delete account 53](#_Toc388777500)

[3.2.2.15. « System administrator » Statistic 54](#_Toc388777501)

[3.2.2.16. « System administrator » Delete comment 56](#_Toc388777502)

[3.2.2.17. « System administrator » Clear rating 57](#_Toc388777503)

[3.2.2.18. « Customer/Guest » Edit request information 59](#_Toc388777504)

[3.2.2.19. « Staff » Input invoice 61](#_Toc388777505)

[3.2.2.20. « Staff » Update status of new trip 63](#_Toc388777506)

[3.2.2.21. « Staff » Update status of arrival trip 65](#_Toc388777507)

[3.2.2.22. « Staff » Update status of departed trip 67](#_Toc388777508)

[3.2.2.23. « Staff » Manage expired request 69](#_Toc388777509)

[3.2.2.24. « Staff » Manage late request 71](#_Toc388777510)

[3.2.2.25. « Staff » View request on way and request delivered 73](#_Toc388777511)

[3.2.2.26. « Staff » Approve request 75](#_Toc388777512)

[3.2.2.27. « Staff » Reject request 76](#_Toc388777513)

[3.2.2.28. « Staff » Assign package 77](#_Toc388777514)

[3.2.2.29. « Staff » Update package(s) status 79](#_Toc388777515)

[3.2.2.30. « Staff » Create a schedule 81](#_Toc388777516)

[3.2.2.31. « Staff » Create a trip 82](#_Toc388777517)

[3.2.2.32. « Staff » Delete a trip 84](#_Toc388777518)

[3.2.2.33. « Staff » Add a station 86](#_Toc388777519)

[3.2.2.34. « Staff » Edit station information 88](#_Toc388777520)

[3.2.2.35. « Staff » Add a route 89](#_Toc388777521)

[3.2.2.36. « Staff » Delete a route 91](#_Toc388777522)

[3.2.2.37. « Staff » Add new coach 93](#_Toc388777523)

[3.2.2.38. « Staff » Delete a coach 94](#_Toc388777524)

[3.2.3. Non-Functional Requirements 96](#_Toc388777525)

[3.2.2.1. Reliability 96](#_Toc388777527)

[3.2.2.2. Availability 96](#_Toc388777528)

[3.2.2.3. Security 96](#_Toc388777529)

[3.2.2.4. Maintainability 96](#_Toc388777530)

[3.2.2.5. Portability 96](#_Toc388777531)

[3.2.2.6. Performance 96](#_Toc388777532)

[3.3. Entity Relationship Diagram 96](#_Toc388777533)

[4. Software Design Description (SDD) 98](#_Toc388777534)

[4.1. Design Overview 98](#_Toc388777536)

[4.2. System Architectural Design 98](#_Toc388777537)

[4.3. Component Diagram 100](#_Toc388777538)

[4.4. Detailed Description of Components 103](#_Toc388777539)

[4.4.1. Entities Classes 103](#_Toc388777545)

[4.4.2. Algorithm Processing Classes 104](#_Toc388777546)

[4.4.2.1. Class diagram 104](#_Toc388777547)

[4.4.2.2. Algorithms description 104](#_Toc388777548)

[4.4.2.3. IRouteChooser 108](#_Toc388777549)

[4.4.2.4. IAssigning 109](#_Toc388777550)

[4.4.3. Data Repository Classes 109](#_Toc388777551)

[4.4.3.1. Class diagram 109](#_Toc388777552)

[4.4.3.2. IRequestRepository Interface 110](#_Toc388777553)

[4.4.3.3. IUserRepository Interface 111](#_Toc388777554)

[4.4.3.4. IJourneyRepository Interface 112](#_Toc388777555)

[4.4.3.5. IAssigningRepository Interface 113](#_Toc388777556)

[4.4.3.6. IFeedbackRepository Interface 114](#_Toc388777557)

[4.4.3.7. IPriceRepository Interface 115](#_Toc388777558)

[4.4.4. Business Logic Classes 116](#_Toc388777559)

[4.4.4.1. Class diagram 116](#_Toc388777560)

[4.4.4.2. IAcount Interface 116](#_Toc388777561)

[4.4.4.3. IAssigning Interface 116](#_Toc388777562)

[4.4.4.4. IStage Interface 118](#_Toc388777563)

[4.4.4.5. ITrip Interface 119](#_Toc388777564)

[4.4.4.6. ICustomer Interface 119](#_Toc388777565)

[4.4.4.7. IFee Interface 120](#_Toc388777566)

[4.4.4.8. IFeedback Interface 120](#_Toc388777567)

[4.4.4.9. IHome Interface 121](#_Toc388777568)

[4.4.4.10. IPaypal Interface 122](#_Toc388777569)

[4.4.4.11. IRequest Interface 123](#_Toc388777570)

[4.4.4.12. ISMS Interface 124](#_Toc388777571)

[4.4.4.13. IStaff Interface 124](#_Toc388777572)

[4.4.4.14. IStation Interface 125](#_Toc388777573)

[4.5. Behavioral Diagrams 125](#_Toc388777584)

[4.5.1. State Machine Diagram: Request Status Transition 125](#_Toc388777586)

[4.5.2. State Machine Diagram: Package Status Transition 126](#_Toc388777587)

[4.5.3. Sequence Diagram: Send Delivery Requests 127](#_Toc388777588)

[4.5.4. Sequence Diagram: Tracking 127](#_Toc388777589)

[4.5.5. Sequence Diagram: Payment 127](#_Toc388777590)

[4.5.6. Sequence Diagram: Feedback 128](#_Toc388777591)

[4.5.7. Sequence Diagram: Approve Requests 128](#_Toc388777592)

[4.5.8. Sequence Diagram: Assign Package 129](#_Toc388777593)

[4.5.9. Sequence Diagram: Package in-transit 129](#_Toc388777594)

[4.5.10. Sequence Diagram: Package arrived 130](#_Toc388777595)

[4.5.11. Sequence Diagram: Package delivered 130](#_Toc388777596)

[4.5.12. Sequence Diagram: Package return 131](#_Toc388777597)

[4.5.13. Sequence Diagram: Manage late payment requests 131](#_Toc388777598)

[4.5.14. Sequence Diagram: Create schedule 132](#_Toc388777599)

[4.5.15. Sequence Diagram: Create trip 133](#_Toc388777600)

[4.5.16. Sequence Diagram: Create staff account 134](#_Toc388777601)

[4.6. User Interface Design 135](#_Toc388777602)

[4.6.1. Home Page 135](#_Toc388777604)

[4.6.2. Send Delivery Request Page 135](#_Toc388777605)

[4.6.3. Online Payment Page 137](#_Toc388777606)

[4.6.4. Approve Request Page 137](#_Toc388777607)

[4.6.5. Assign Package Page 138](#_Toc388777608)

[4.6.6. Tracking Page 138](#_Toc388777609)

[4.6.7. Update Package Status Page 139](#_Toc388777610)

[4.6.8. Create Trip Page 141](#_Toc388777611)

[4.6.9. Feedback and Review Page 142](#_Toc388777612)

[4.6.10. Manage Fee Page 143](#_Toc388777613)

[4.7. Database Design 144](#_Toc388777615)

[4.7.1. Logical database design 144](#_Toc388777616)

[4.7.1.1. User 148](#_Toc388777621)

[4.7.1.2. UserInfo 148](#_Toc388777622)

[4.7.1.3. Coach 149](#_Toc388777623)

[4.7.1.4. CoachType 149](#_Toc388777624)

[4.7.1.5. Route 149](#_Toc388777625)

[4.7.1.6. Trip 149](#_Toc388777626)

[4.7.1.7. Station 150](#_Toc388777627)

[4.7.1.8. Schedule 151](#_Toc388777628)

[4.7.1.9. Request 151](#_Toc388777629)

[4.7.1.10. DeliveryStatus 152](#_Toc388777630)

[4.7.1.11. Invoice 152](#_Toc388777631)

[4.7.1.12. ManageFee 152](#_Toc388777632)

[4.7.1.13. Comment 153](#_Toc388777633)

[4.7.1.14. Rating 153](#_Toc388777634)

[4.7.1.15. Assigning 154](#_Toc388777635)

[4.7.1.16. Stage 154](#_Toc388777636)

[4.7.1.17. RouteStage 154](#_Toc388777637)

[4.7.1.18. Notification 155](#_Toc388777638)

[4.7.1.19. Province 155](#_Toc388777639)

[4.7.2. Physical database design 155](#_Toc388777640)

[5. Software Test Document 157](#_Toc388777641)

[5.1. Introduction 157](#_Toc388777643)

[5.1.1. System Overview 157](#_Toc388777650)

[5.1.2. Test Approach 160](#_Toc388777651)

[5.2. Test Plan 160](#_Toc388777652)

[5.2.1. Features to be tested 160](#_Toc388777654)

[5.2.2. Features not to be tested 161](#_Toc388777655)

[5.3. System Test Cases 162](#_Toc388777656)

[5.3.1. Send new delivery request 162](#_Toc388777658)

[5.3.2. Online payment 163](#_Toc388777659)

[5.3.3. Tracking package 164](#_Toc388777660)

[5.3.4. Feedback and review 165](#_Toc388777661)

[5.3.5. Approve delivery requests 166](#_Toc388777662)

[5.3.6. Assign packages 167](#_Toc388777663)

[5.3.7. Prepare for packages return 167](#_Toc388777664)

[5.3.8. Create the trip 168](#_Toc388777665)

[5.3.9. Create schedules 170](#_Toc388777666)

[5.3.10. Monitor price/fee 171](#_Toc388777667)

[5.3.11. Monitor price/fee 172](#_Toc388777668)

[5.3.12. Create staff account 172](#_Toc388777669)

[5.4. Checklists 174](#_Toc388777670)

[5.4.1. Checklist of Validation 174](#_Toc388777672)

[5.4.2. Submission Checklist 174](#_Toc388777673)

[6. Software User’s Manual 176](#_Toc388777674)

[6.1. Installation Guide 176](#_Toc388777683)

[6.1.1. Prerequisites 176](#_Toc388777684)

[6.1.2. Installation procedures 176](#_Toc388777685)

[6.2. User’s Guide 179](#_Toc388777691)

[6.2.1. Guides for Customer 179](#_Toc388777693)

[6.2.1.1. Log in to the system 179](#_Toc388777698)

[6.2.1.2. Log out of the system 180](#_Toc388777699)

[6.2.1.3. Register an account with Customer role 181](#_Toc388777700)

[6.2.1.4. Send a delivery request 182](#_Toc388777701)

[6.2.1.5. View station details and review 184](#_Toc388777702)

[6.2.1.1. Online payment 186](#_Toc388777703)

[6.2.1.2. Tracking package(s) 187](#_Toc388777704)

[6.2.2. Guides for Staffs and System Admins 188](#_Toc388777705)

[6.2.2.1. Approve pending requests 188](#_Toc388777707)

[6.2.2.2. Create a trip 190](#_Toc388777708)

[6.2.2.3. Assign package(s) 192](#_Toc388777709)

[6.2.2.4. Confirm package(s) departed from start-station 193](#_Toc388777710)

[6.2.2.5. Confirm package(s) arrived in end-station 196](#_Toc388777711)

[6.2.2.6. Confirm package(s) delivered 197](#_Toc388777712)

[6.2.2.7. Prepare package(s) for return 198](#_Toc388777713)

[6.2.2.8. Extend due-date for online payment 199](#_Toc388777714)

[6.2.2.9. Statistics 200](#_Toc388777715)

[6.2.2.10. Add a fee 202](#_Toc388777716)

[6.2.2.11. Monitor fees 202](#_Toc388777717)

[Appendix 203](#_Toc388777718)

[Reference 203](#_Toc388777719)

# Table of figures

[Figure 1. An overview of the i-Deliver system 18](#_Toc388777721)

[Figure 2. The waterfall software lifecycle model 19](#_Toc388777722)

[Figure 3. Overall use case 32](#_Toc388777723)

[Figure 4. Model-View-Controller Architecture 99](#_Toc388777724)

[Figure 5. Home page 134](#_Toc388777725)

[Figure 6. Send delivery request page 135](#_Toc388777726)

[Figure 7. Select new request to make a payment 136](#_Toc388777727)

[Figure 8. Make an online payment via Paypal 136](#_Toc388777728)

[Figure 9. Approve request page 137](#_Toc388777729)

[Figure 10. Assign packages page 137](#_Toc388777730)

[Figure 11. Tracking package page 138](#_Toc388777731)

[Figure 12. Confirm package departed 139](#_Toc388777732)

[Figure 13. Confirm package arrived 139](#_Toc388777733)

[Figure 14. Confirm package delivered 140](#_Toc388777734)

[Figure 15. Prepare package to return 140](#_Toc388777735)

[Figure 16. Extend due-date for late payment requests 140](#_Toc388777736)

[Figure 17. Create trip page 141](#_Toc388777737)

[Figure 18. Feedback and review page 142](#_Toc388777738)

[Figure 19. Manage fee page 143](#_Toc388777739)

[Figure 20. Logical database design 146](#_Toc388777740)

[Figure 21. Physical database design 155](#_Toc388777741)

[Figure 22. i-Deliver web application release folder 175](#_Toc388777742)

[Figure 23. Create i-Deliver database using SQL Query 176](#_Toc388777743)

[Figure 24. Add new web application 176](#_Toc388777744)

[Figure 25. Configure database connection string 177](#_Toc388777745)

[Figure 26. Run i-Deliver website 178](#_Toc388777746)

[Screen 1. Login screen 179](#_Toc388777747)

[Screen 2. Logout screen 180](#_Toc388777748)

[Screen 3. Register screen 181](#_Toc388777749)

[Screen 4. Customer screen 182](#_Toc388777750)

[Screen 5. Send delivery request screen 182](#_Toc388777751)

[Screen 6. Station information screen 183](#_Toc388777752)

[Screen 7. Customer main screen 184](#_Toc388777753)

[Screen 8. Station list for review screen 184](#_Toc388777754)

[Screen 9. Review screen 185](#_Toc388777755)

[Screen 10. Request view before make a payment screen 186](#_Toc388777756)

[Screen 11. Invoice review before checkout screen 186](#_Toc388777757)

[Screen 12. Tracking package screen 187](#_Toc388777758)

[Screen 13. Requests review screen 188](#_Toc388777759)

[Screen 14. Input invoice screen 188](#_Toc388777760)

[Screen 15. Trip list screen 190](#_Toc388777761)

[Screen 16. Create trip screen 190](#_Toc388777762)

[Screen 17. Assign package(s) screen 192](#_Toc388777763)

[Screen 18. Confirm departed package(s) screen 193](#_Toc388777764)

[Screen 19. List trip from the other station screen 195](#_Toc388777765)

[Screen 20. List request that delivered to customer screen 197](#_Toc388777766)

[Screen 21. List late deliver request screen 198](#_Toc388777767)

[Screen 22. List late payment requests screen 199](#_Toc388777768)

[Screen 23. Statistic screen 200](#_Toc388777769)

[Screen 24. Add a fee screen 201](#_Toc388777770)

[Screen 25. Monitor fees screen 201](#_Toc388777771)

[Diagram 1. Actor overview diagram 30](#_Toc388777772)

[Diagram 2. Entity relationship diagram 96](#_Toc388777773)

[Diagram 3. Component Diagram 100](#_Toc388777774)

[Diagram 4. Entities Class Diagram 102](#_Toc388777775)

[Diagram 5. Algorithm Processing Class Diagram 103](#_Toc388777776)

[Diagram 6. Data Repository Class Diagram 109](#_Toc388777777)

[Diagram 7. Business Logic Class Diagram 115](#_Toc388777778)

[Diagram 8. State machine diagram: Request status transition 125](#_Toc388777779)

[Diagram 9. State machine diagram: Package status transition 125](#_Toc388777780)

[Diagram 10. Sequence diagram: Send delivery request 126](#_Toc388777781)

[Diagram 11. Sequence diagram: Tracking package 126](#_Toc388777782)

[Diagram 12. Sequence diagram: Payment 127](#_Toc388777783)

[Diagram 13. Sequence diagram: Feedback 127](#_Toc388777784)

[Diagram 14. Sequence diagram: Approve Requests 128](#_Toc388777785)

[Diagram 15. Sequence diagram: Assign packages 128](#_Toc388777786)

[Diagram 16. Sequence diagram: Packages in-transit 129](#_Toc388777787)

[Diagram 17. Sequence diagram: Package arrived 129](#_Toc388777788)

[Diagram 18. Sequence diagram: Package delivered 130](#_Toc388777789)

[Diagram 19. Sequence diagram: Package return 130](#_Toc388777790)

[Diagram 20. Sequence diagram: Manage late payment requests 131](#_Toc388777791)

[Diagram 21. Sequence diagram: Create schedule 132](#_Toc388777792)

[Diagram 22. Sequence diagram: Create trip 133](#_Toc388777793)

[Diagram 23. Sequence diagram: Create staff account 133](#_Toc388777794)

# Terminology

|  |  |  |
| --- | --- | --- |
| No. | Terminology | Explanation |
| 1. | Customer | A person who needs to request for goods delivery from their province to another province. |
| 2. | Staff | A person in charge of managing delivery requests and related services of journeys. |
| 3. | System Administrator | A person take responsibilities of manage staff and price changes |
| 4. | Delivery Request | A request sent by a customer to the staff to ask for a delivery request with the i-Deliver system |

# Introduction

## Introduction

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Title:** | | *Building a web-based application that manages the activities of delivery service system by coach* | | | |
| **Start Date:** | | Jan 6, 2014 | | **Finished Date:** |  |
| **No** | **Full name** | | **Role** | **Position** | **Contact** |
| 1. | Nguyễn Trọng Tài | | Supervisor | Lecturer | [TaiNT@fpt.edu.vn](mailto:PhuongLHK@fpt.edu.vn) |
| 2. | Đào Bảo Long | | Developer, Tester | Team Leader | [LongDBSE60690@fpt.edu.vn](mailto:LongDBSE60690@fpt.edu.vn) |
| 3. | Lê Phúc Lữ | | Developer, Tester | Member | [LuLP60563@fpt.edu.vn](mailto:DucBM60324@fpt.edu.vn) |
| 4. | Nguyễn Thanh Tùng | | Developer, Tester | Member | [TungNTSE60609@fpt.edu.vn](mailto:TungNTSE60609@fpt.edu.vn) |
| 5. | Nguyễn Tấn Đức | | Developer, Tester | Member | [DucNTSE60660@fpt.edu.vn](mailto:DucNTSE60660@fpt.edu.vn) |

## The initial idea of group

Nowadays, freight traffic between cities in a day is huge. We have the passenger transportation service providers like Mai Linh or the goods delivery service providers like Tin Thanh, Hop Nhat. And we also have the combination like Phuong Trang. Before working on this project, our team conducted a survey in a passenger transportation service provider. Here are the major findings during the survey:

**Phuong Trang Travel & Transportation Company:**

This company provides passenger transport service using coach. Beside this service, this company also provides goods delivery service. They receive a lot delivery requests per day but all of them still managed by staff manually using papers/books. It makes planning and scheduling for goods delivery become complicated beside passenger transportation arrangement, also makes hard to manage all goods to ensure integrity.

From the problems above, our team decided to develop a delivery service system for passenger transportation service providers which using coach. It operates difference from professional delivery service which using cars specially made for goods delivery.

## Overview of existing methods

### Requests management

The goods delivery service’s staffs have to write down all of delivery requests, monitoring – planning – scheduling manually using papers/books and their memory.

### Packages arrangement

They just simply fill-up empty cargo compartments of each coach as much as possible.

### Packages management

Tracking, searching … is temporary unavailable

## Limitations of existing system

### Requests management

By using papers/books and memory, staffs cannot ensure information accuracy; controllable requests, delivery status of requests.

### Packages management and arrangement

They don’t have any specific fee calculating formula, it’s just estimation.

They don’t have any plan to arrange package on each coach so it’s hard to balance the freight on each route.

### Schedules and trips management

They don’t have detail plans to schedule for each trip, all still managed by demands on real-time.

## Benefits of expected system

The project aims to develop a web-based application that

* Create an easy way for customers to make goods delivery request by allow them to post a new requests on website with registered account then they can tracking their package to ensure delivery.
* Mainly assists service providers’ staff to manage all goods delivery request by using optimized planning and scheduling algorithms, manage the fee of the delivery requests
* Allow customers to comment and rating for the service.

## Business outline

Following the project objectives above, the scope of the project is constrained to the following statements:

* The users of the web-application should interact with the web-application itself through a friendly and attractive user interface.
* *For* ***customers***, this web-application should provide fundamental functions such as register, login then post, edit and cancel goods delivery requests. Through integrated e-payment services, they can pay for their delivery requests. They also can search for posted requested, tracking their package. After using service, customers can comment and rating for the service.
* *For* ***system administrators***, the web-application should allow them to monitor fundamental information about customers, staffs, coaches, routes, and fee.
* *For* ***staffs****,* i-Deliver is a management system allows them to import request, fee calculating, invoice making, planning and scheduling for goods delivery, assign packages for specific coach, specific route.

## Approaches

* Adopt the perspective of a customer during the whole development process, in order to develop a web-based application that promotes usability and interactivity as much as possible.
* Negotiate with web service providers to use their services, and then agree upon how the application communicates with the web services, what information should be retained and what should be eliminated.
* Try to provide staffs the customized tools for manage requests easily.
* Conduct research on how to determine the most appropriate fee based on the average fee of the same service providers in the market.
* Conduct research on how to optimize planning and scheduling for goods delivery.

## Group of functions

|  |  |
| --- | --- |
| Functions for customers | * Allow customers register, login, post/edit/cancel requests, search for, and tracking requests; * Allow customers to rate by different criteria and write comments/reviews on service; |
| Functions for staffs | * Allow staffs to manage fundamental information about requests, fee calculating, invoice making; * Allow staffs to planning and scheduling for goods delivery; * Allow staffs to monitoring and assigning for coaches and routes; |
| Functions for system admins | * Allow system admins to manage all relevant information about the system, customers, staffs, coaches and routes; * Allow system admins to grant access rights to other users of the system; * Allow system admins to decide fee calculating formulas; * Allow system admins to collect and export data to statistic. |

# Software Project Management Plan (SPMP)



## Problem Definition



### Name of this Capstone Project

|  |  |
| --- | --- |
| **Official name** | Building a web-based application that manages the activities of delivery service system by coach |
| **Vietnamese name** | Xây dựng ứng dụng web quản lý các hoạt động cho dịch vụ vận chuyển hàng hóa thông qua hệ thống xe khách đường dài |
| **Abbreviation** | i-Deliver |

### Problem Abstract

The idea of the project is to develop a web-based application that assists delivery service staffs/administrators in manages goods delivery requests, package arrangement, and planning and scheduling, fee management. It also helps customers to make goods delivery request, searching and tracking their package, make a payment online.

### Project Overview



#### The Current System

The idea of this project is developing a delivery service system for passenger transportation service providers which using coach. Before working on this project, we conducted a survey about traditional goods delivery process.

**Traditional goods delivery process:**

Customers will pick their package to delivery service station and provide the name of receiver, destination, and type of goods, weight and size. Then staffs of delivery service will calculate transport fee and give the customer a package invoice which contain provided information and destination station address.

#### The Proposed System

By working on this project, we will develop a service system that assists delivery services providers in Ho Chi Minh City to be closer to their customers. The system has some significant features:

**Create user-oriented interfaces for administrators to simplify management:**

The i-Deliver system support administrators to manage delivery fee like define or edit fee factor. It also helps them to manage related activities of a journey, included trips, stages, stations and routes. Of course the system will provide mechanisms to manage users/staffs.

**Support staffs in delivery requests management, package arrangement, planning and scheduling:**

This system provide easy-to-use interface for requests management like make a new request, edit request information, fee calculating, invoice making, packages arrangement and planning and scheduling, assign packages for specific coach/route.

**Easy-to-use tool for customer to make goods delivery requests, searching and review/edit requests, tracking their package:**

The i-Deliver system is integrated with some technique to help making goods delivery requests online, searching then review or edit requests information, tracking customer’s packages by using request code.

**Optimize arrangement, planning and scheduling for package delivery process**

In traditional way, delivery service staffs have to planning, scheduling and arrange packages for each coaches manually. They just simply fill-up empty cargo compartments as much as possible. This system helps them do their works easily, efficiently by using tools, which optimized by arrangement, planning and scheduling algorithms.

**The i-Deliver system’s users:**

1. **Guests**: non-authorized members can register new account of the i-Deliver website to be granted full access permission or they just can search for routes which delivery service providers operating.
2. **Members:** guests had an authorized account can login to the i-Deliver website to

* Make goods delivery requests;
* Search for posted requests;
* Edit posted requests;
* Cancel posted requests (also have constraints);
* Tracking packages;
* Comment and rating for service;
* Statistics

1. **Administrators**: owners of the i-Deliver website who have highest permission can

* Create new staff account;
* Edit staff account information;
* Delete staff account;
* Define fee calculating formula and fixed value;
* Edit fee calculating formula and fixed value;
* Statistics

1. **Staffs**: users who has account which created by administrators have right to

* Manage requests: approve, reject, update status, assign and scheduling;
* Search for packages, routes or members;
* Manage journeys: include add new, edit or delete routes, stations, trips and coaches;
* Manage comments and rates: delete violated/spam comments

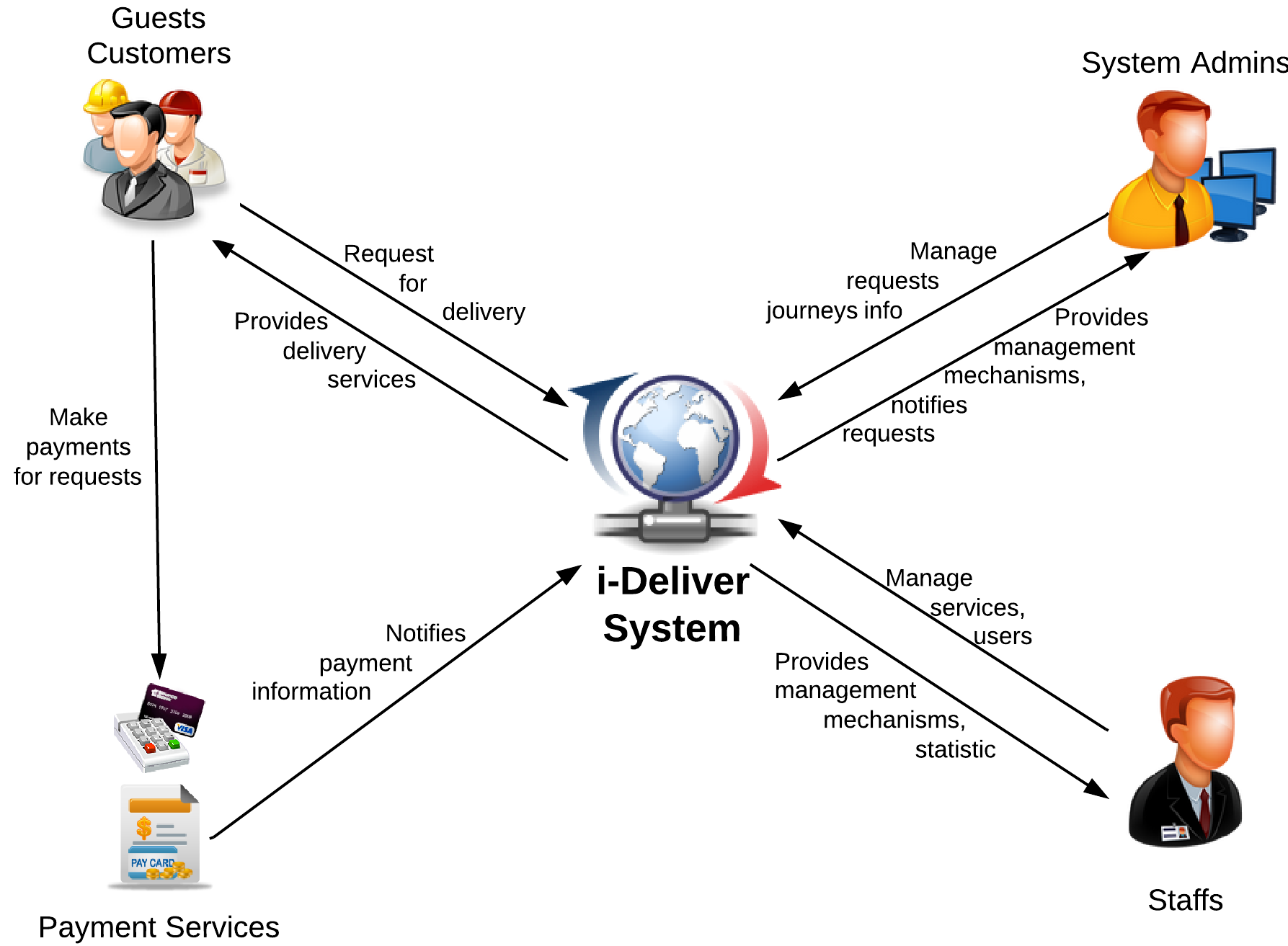


Figure . An overview of the i-Deliver system

#### Boundaries of the System

There is no previous version of this system. The product will be developed from scratch, independent of any current system.

As said previously, the system under development is not a delivery service management system. It does not provide mechanisms to manage all activities related to goods delivery. In our team’s scope, the i-Deliver is a system that manage related activities of transportation service providers which using coaches. It means that the service provider only working on 2-stations routes (likes Saigon-Nha Trang, Saigon-Vung Tau, Saigon-Hanoi); they don’t provide mechanism to deliver packages to stations between starting point and destination. That is our team’s future plan for this system.

It focuses mainly on providing easy-to-use interfaces and tools, which support both customers and delivery service staffs.

The final product of this Capstone Project includes

* A service portal which helps customers make goods delivery requests;
* A management module for staffs/administrators of delivery service;
* All the documents involved in the development process.

#### Development Environment

Below is the list of hardware and software requirements needed for the development environment of the project.

**Hardware requirements**

* Personal computers for developing with the minimum configuration: 2 Gb of RAM, 100GB of hard disk, Core 2 Duo 2.0 Ghz;
* A server computer for testing with the minimum configuration: 4 Gb of RAM, 100GB of hard disk, Core 2 Duo 2.0 Ghz;
* All computers must be connected to the Internet.

**Software requirements**

* Operating system: Windows 7 or above;
* Web Server: IIS Express 8;
* Framework: .NET Framework 4.5;
* IDE: Visual Studio 2012;
* DBMS: SQL Server 2008 R2;
* Source Control: Tortoise SVN 1.8.4.

## Project organization



### Software Process Model

The waterfall software lifecycle model will be used to guide the development of the system. The waterfall model includes five major phases as in the figure below, enforcing moving to the next phase only after completion of the previous phase.

Requirement Specifications

System and Software Designs

Implementation and Unit Testing

Integration and System Testing

Operation and Maintenance

Figure . The waterfall software lifecycle model

### Roles and Responsibilities

|  |  |  |  |
| --- | --- | --- | --- |
| No | Full name | Role in group | Responsibilities |
| 1 | Nguyễn Trọng Tài | Supervisor | * Give advice on business and technical problems; * Review and approve of project documents and product deliverables; * Assess the performance of team members. |
| 2 | Đào Bảo Long | Team Leader,  Developer,  Tester | * Create project management plan and distribute tasks to the other members; * Monitor the development process and review the deliverables; * Work on system architecture and detailed designs; * Implement; * Prepare documents; * Perform unit testing; * Deploy the final product. |
| 3 | Lê Phúc Lữ | Developer, Tester | * Research on; * Design user interfaces; * Work on detailed designs; * Implement; * Prepare documents; * Perform unit testing, system testing, and integration test. |
| 4 | Nguyễn Thanh Tùng | Developer, Tester | * Research on; * Design user interfaces; * Work on detailed designs; * Implement; * Perform unit testing, system testing, and integration test. |
| 5 | Nguyễn Tấn Đức | Developer, Tester | * Research on; * Design user interfaces; * Work on detailed designs; * Implement; * Perform unit testing, system testing, and integration test. |

### Tools and Techniques

The tools that will be used to develop the system include:

* *Developing tools:* Microsoft Visual Studio 2012; Tortoise SVN 1.8.4; Microsoft SQL Server 2008 RC;
* *Modeling tools:* StarUML 5.0.2.1570;
* *Document tools:* Microsoft Office 2010.

## Project management plan



### Tasks

Below are all the major tasks that need to be performed sequentially during the development of the system.



#### Task 1: Initiating

|  |  |
| --- | --- |
| ***Task name*** | Initiating |
| ***Descriptions*** | Perform research/survey on some delivery service providers and pricing model; decide upon the technology that will be used to develop the system. |
| ***Deliverables*** | Report 1 – Project Introduction |
| ***Resources needed*** | All team members; 6 days |
| ***Dependencies and constraints*** | N/A |
| ***Risks*** | Performing survey on the delivery service providers can be difficult because of their business; the chosen technology is new to some members. |

#### Task 2: Planning

|  |  |
| --- | --- |
| ***Task name*** | Planning |
| ***Descriptions*** | Create the project management plan; break the system into modules and assign tasks to each member. |
| ***Deliverables*** | Report 2 – Software Project Management Plan |
| ***Resources needed*** | All team members; 6 days |
| ***Dependencies and constraints*** | Task 1 has finished |
| ***Risks*** | Team leader has no experience in managing software projects; all members are still not acquainted with the new technology. |

#### Task 3: Specifying requirements

|  |  |
| --- | --- |
| ***Task name*** | Specifying requirements |
| ***Descriptions*** | Discuss and agree upon the software requirements, what is to be developed and what is not; generate detailed descriptions of all the functions to be developed. |
| ***Deliverables*** | Report 3 – Software Requirement Specification |
| ***Resources needed*** | All team members; 15 days |
| ***Dependencies and constraints*** | Task 2 has finished |
| ***Risks*** | Many aspects of the problem are still unclear to team members; has no experience of working in a delivery service management project. |

#### Task 4: Designing database

|  |  |
| --- | --- |
| ***Task name*** | Designing database |
| ***Descriptions*** | Design the database based on the requirements collected, through three major steps: Conceptual, Logical, and Physical Design |
| ***Deliverables*** | ERD and the physical database with sample data |
| ***Resources needed*** | All team members; 3 days |
| ***Dependencies and constraints*** | Task 3 has finished |
| ***Risks*** | Some of the requirements specified are not clear and cannot be translated into corresponding entities; little experience in organizing data. |

#### Task 5: Creating Software Design Description

|  |  |
| --- | --- |
| ***Task name*** | Creating Software Design Description |
| ***Descriptions*** | Agree upon the system architecture; work on the detailed design of each module; decide which techniques are appropriate to which modules; design the user interfaces for users to interact with. |
| ***Deliverables*** | Report 4 – Software Design Description |
| ***Resources needed*** | All team members; 12 days |
| ***Dependencies and constraints*** | Task 4 has finished |
| ***Risks*** | Some functions are difficult to find appropriate methods to implement; initial development environment setup is also difficult. |

#### Task 6: Implementing

|  |  |
| --- | --- |
| ***Task name*** | Implementing |
| ***Descriptions*** | Each team member implement all the functions that he or she was assigned and regularly check in the code to the Tortoise SVN; regularly validate that the implementation is consistent with the system and detailed designs. |
| ***Deliverables*** | The implemented website |
| ***Resources needed*** | All team members; 24 days |
| ***Dependencies and constraints*** | Task 5 has finished |
| ***Risks*** | Some design documents contain errors; implementation is not always consistent with the system and detailed designs. |

#### Task 7: Performing Testing

|  |  |
| --- | --- |
| ***Task name*** | Performing Unit Testing |
| ***Descriptions*** | Create and perform appropriate test cases for all main functions; record the test results for later reference; fix all the bugs found during the testing sessions. |
| ***Deliverables*** | Report 5 – Software Test Documentation |
| ***Resources needed*** | All team members; 12 days |
| ***Dependencies and constraints*** | Task 6 has finished |
| ***Risks*** | Lack of test cases for some non-critical functions; not enough time to intensively test all the functions. |

#### Task 8: Writing User’s Manual

|  |  |
| --- | --- |
| ***Task name*** | Writing User’s Manual |
| ***Descriptions*** | Writing a user’s manual to instruct the users, including guest, users, staffs, and system administrators, how to use the system. |
| ***Deliverables*** | Report 6 – User’s Manual |
| ***Resources needed*** | All team members; 5 days |
| ***Dependencies and constraints*** | Task 7 has finished |
| ***Risks*** | Some of the functions are not consistent with the user requirements, causing the user’s manual to be inconsistent with the user requirements. |

#### Task 9: Deploying the Website

|  |  |
| --- | --- |
| ***Task name*** | Deploying the Website |
| ***Descriptions*** | Perform acceptance testing and deploy the website to a host on the Internet |
| ***Deliverables*** | The complete website |
| ***Resources needed*** | All team members; all days left |
| ***Dependencies and constraints*** | Task 8 has finished |
| ***Risks*** | Little experience of deploying a website to a real host |

#### Task 10: Finalizing and Closing

|  |  |
| --- | --- |
| ***Task name*** | Finalizing and Closing |
| ***Descriptions*** | Finalize all related documents and prepare for presentation |
| ***Deliverables*** | The complete website and related documents, presentation |
| ***Resources needed*** | All team members; 4 days |
| ***Dependencies and constraints*** | Task 9 has finished |

### Task sheet

Refer to the next page for the detailed task sheet of the project plan.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task Name | Length | Start | Finish | Predecessor | Resources |
| 1. Initiating | **6 days** | **Mon 06/01/14** | **Sat 11/01/14** |  |  |
| * 1. Identify key stakeholders | 1 day | Mon 06/01/14 | Mon 06/01/14 |  | DucNT,LongDB,LuLP,TungNT |
| * 1. Research & discuss about business and technology | 2 days | Tue 07/01/14 | Wed 08/01/14 | 1.1 | DucNT,LongDB,LuLP,TungNT |
| * 1. Research on similar existing system | 2 days | Thu 09/01/14 | Fri 10/01/14 |  | DucNT,LongDB,LuLP,TungNT |
| * 1. Hold project kick-off meeting | 0.5 days | Sat 11/01/14 | Sat 11/01/14 | 1.2, 1.3 | DucNT,LongDB,LuLP,TungNT |
| * 1. Report 1 – Project Introduction | 0.5 days | Sat 11/01/14 | Sat 11/01/14 | 1.4 | DucNT,LongDB,LuLP,TungNT |
| 1. Planning | **6 days** | **Mon 13/01/14** | **Mon 20/01/14** |  |  |
| * 1. Hold team planning meeting | 0.5 days | Mon 13/01/14 | Mon 13/01/14 | 1.5 | DucNT,LongDB,LuLP,TungNT |
| * 1. Prepare problem abstract, proposed solution, coding convention | 2 days | Mon 13/01/14 | Wed 15/01/14 | 2.1 | TungNT |
| * 1. Prepare major tasks for the whole team | 1.5 days | Wed 15/01/14 | Thu 16/01/14 | 2.1 | LuLP |
| * 1. Prepare management plan and determine task resources, durations, and dependencies | 2 days | Wed 15/01/14 | Thu 16/01/14 | 2.1 | LongDB |
| * 1. Configure Tortoise SVN | 0.5 days | Mon 20/01/14 | Mon 20/01/14 | 2.4 | LongDB |
| * 1. Report 2 – Project Management Plan | 1.5 days | Fri 17/01/14 | Mon 20/01/14 | 2.1-2.5 | DucNT,LongDB,LuLP,TungNT |
| 1. Specifying requirements | **15 days** | **Mon 20/01/14** | **Fri 07/02/14** |  |  |
| * 1. Identify users and users’ requirements | 3 days | Mon 20/01/14 | Thu 23/01/14 | 2.6 | LuLP,TungNT,DucNT |
| * 1. Define system requirements | 1 day | Thu 23/01/14 | Fri 24/01/14 | 3.1 | LongDB |
| * 1. Define non-functional requirements | 1 day | Thu 23/01/14 | Fri 24/01/14 | 3.1 | DucNT |
| * 1. Determine main flows | 3 days | Thu 23/01/14 | Tue 28/01/14 | 3.1 | LuLP,TungNT |
| * 1. Specify functional requirements for users/customers (requests making, searching and reviews posted requests) | 3 days | Tue 28/01/14 | Fri 31/01/14 |  | TungNT |
| * 1. Specify functional requirements for staffs (requests management, package arrangement and scheduling) | 3 days | Tue 28/01/14 | Fri 31/01/14 |  | LuLP |
| * 1. Specify functional requirements for administrators | 3 days | Tue 28/01/14 | Fri 31/01/14 |  | DucNT |
| * 1. Specify functional requirements for coaches, trips, stages, stations, routes management, pricing model, fee calculating | 3 days | Tue 28/01/14 | Fri 31/01/14 |  | LongDB |
| * 1. Specify functional requirements for statistics | 2 days | Fri 31/01/14 | Tue 04/02/14 |  | TungNT |
| * 1. Report 3 – Software Requirement Specification | 2 days | Wed 05/02/14 | Fri 07/02/14 | 3.1-3.9 | LongDB |
| 1. Designing database | **3 days** | **Fri 07/02/14** | **Tue 11/02/14** |  |  |
| * 1. Discuss on conceptual data model | 0.5 days | Fri 07/02/14 | Fri 07/02/14 | 3.11 | DucNT,LuLP |
| * 1. Creating ERD | 0.5 days | Fri 07/02/14 | Fri 07/02/14 | 4.1 | LongDB,TungNT |
| * 1. Validate ERD against software requirements | 1 day | Mon 10/02/14 | Mon 10/02/14 | 4.2 | DucNT,LongDB,LuLP,TungNT |
| * 1. Create logical model and physical database | 1 day | Tue 11/02/14 | Tue 11/02/14 | 4.3 | DucNT,LongDB,LuLP,TungNT |
| 1. Creating Software Design Description | **12 days** | **Wed 12/02/14** | **Thu 27/02/14** |  |  |
| * 1. Discuss on system architecture | 0.5 days | Wed 12/02/14 | Wed 12/02/14 | 3.11 | DucNT,LuLP,TungNT |
| * 1. Configure development environment | 0.5 days | Wed 12/02/14 | Wed 12/02/14 |  | LongDB |
| * 1. Design the master page | 1.5 days | Wed 12/02/14 | Thu 13/02/14 | 3.11 | LongDB |
| * 1. Design the master customer layout | 2.5 days | Wed 12/02/14 | Fri 14/02/14 | 3.11 | TungNT |
| * 1. Design the master staff layout | 2.5 days | Wed 12/02/14 | Fri 14/02/14 |  | LuLP |
| * 1. Design the master admin layout | 1.5 days | Wed 12/02/14 | Thu 13/02/14 |  | DucNT |
| * 1. Design pages for log in, log out, register of customer | 1 day | Fri 14/02/14 | Fri 14/02/14 |  | DucNT |
| * 1. Design pages for posting new request, edit request, view request | 2 days | Mon 17/02/14 | Tue 18/02/14 |  | LuLP |
| * 1. Design pages for customer to view and edit profile | 1.5 days | Mon 17/02/14 | Tue 18/02/14 |  | DucNT |
| * 1. Design page for admin manage fee, stage, route, staff. | 2.5 days | Mon 17/02/14 | Wed 19/02/14 |  | TungNT,LongDB |
| * 1. Design page for staff manage trip, request and invoice. | 2.5 days | Wed 19/02/14 | Fri 21/02/14 |  | LongDB,TungNT |
| * 1. Design page for staff statistic request and trip information. | 1.5 days | Tue 18/02/14 | Wed 19/02/14 |  | DucNT |
| * 1. Design page for staff schedule the package delivery time. | 1 day | Wed 19/02/14 | Wed 19/02/14 |  | LuLP |
| * 1. Design layout for user rate and post comment of routes | 1 day | Mon 24/02/14 | Mon 24/02/14 |  | TungNT |
| * 1. Design page for tracking the package | 1 day | Mon 24/02/14 | Mon 24/02/14 |  | LongDB |
| * 1. Design page for admin manage comment, rating | 1 day | Thu 20/02/14 | Thu 20/02/14 |  | LuLP |
| * 1. Design page for user statistic their request information | 0.5 days | Thu 20/02/14 | Thu 20/02/14 |  | DucNT |
| * 1. Create main sequence diagrams of view, search, edit and delete request | 1.5 days | Thu 20/02/14 | Fri 21/02/14 |  | DucNT |
| * 1. Create main sequence diagrams of manage comment, rating, fee, stage, route, and staff | 1.5 days | Mon 24/02/14 | Tue 25/02/14 |  | DucNT,LuLP |
| * 1. Create main sequence diagrams of posting and rating a route | 0.5 days | Tue 25/02/14 | Tue 25/02/14 |  | DucNT |
| * 1. Create main sequence diagrams of assigning package, edit status of request and manage time (departure/arrival) of coach. | 1 day | Tue 25/02/14 | Tue 25/02/14 |  | TungNT |
| * 1. Draw entity class diagram | 0.5 days | Tue 25/02/14 | Tue 25/02/14 |  | LongDB |
| * 1. Draw model class diagram | 0.5 days | Tue 25/02/14 | Tue 25/02/14 |  | LongDB |
| * 1. Compose physical database description | 1 day | Wed 26/02/14 | Wed 26/02/14 |  | DucNT,LongDB,LuLP,TungNT |
| * 1. Report 4 – Software Design Description | 1 day | Thu 27/02/14 | Thu 27/02/14 |  | DucNT,LongDB,LuLP,TungNT |
| 1. Implementing | **24 days** | **Fri 28/02/14** | **Wed 02/04/14** |  |  |
| * 1. Log in, log out, register | 1 day | Fri 28/02/14 | Fri 28/02/14 | 5.5 | DucNT |
| * 1. Search and view routes | 1 day | Fri 28/02/14 | Fri 28/02/14 | 5.6 | LongDB |
| * 1. Post and view request | 1.5 days | Fri 28/02/14 | Mon 03/03/14 | 5.6 | TungNT |
| * 1. Search, edit, delete request | 1.5 days | Mon 03/03/14 | Tue 04/03/14 | 5.7 | TungNT |
| * 1. Rating for route | 1 day | Fri 28/02/14 | Fri 28/02/14 | 5.7 | LuLP |
| * 1. Comment for route | 1 day | Mon 03/03/14 | Mon 03/03/14 | 5.3 | LuLP |
| * 1. Tracking package | 3 days | Mon 03/03/14 | Wed 05/03/14 | 5.3 | LongDB |
| * 1. Manage all coach and relative information | 4 days | Mon 03/03/14 | Thu 06/03/14 | 5.8 | DucNT |
| * 1. Manage all user and relative information | 4 days | Wed 05/03/14 | Mon 10/03/14 | 5.9 | TungNT |
| * 1. Manage all rating of user | 2 days | Tue 04/03/14 | Wed 05/03/14 | 5.11 | LuLP |
| * 1. Manage all comment user | 2 days | Thu 06/03/14 | Fri 07/03/14 | 5.10 | LuLP |
| * 1. Manage fee of service | 2 days | Mon 10/03/14 | Tue 11/03/14 | 5.10 | DucNT |
| * 1. Statistics of users | 2 days | Thu 06/03/14 | Fri 07/03/14 |  | LongDB |
| * 1. Statistics of staffs | 2 days | Mon 10/03/14 | Tue 11/03/14 |  | LuLP |
| * 1. Statistics of administrators | 2 days | Mon 10/03/14 | Tue 11/03/14 |  | LongDB |
| * 1. Manage all trips | 3 days | Tue 11/03/14 | Thu 13/03/14 |  | TungNT |
| * 1. Manage package | 2 days | Wed 12/03/14 | Thu 13/03/14 |  | DucNT |
| * 1. Scheduling for package | 3 days | Wed 12/03/14 | Fri 14/03/14 |  | LuLP |
| * 1. Integrate all implemented functions | 13 days | Mon 17/03/14 | Wed 02/04/14 |  | DucNT,LongDB,LuLP,TungNT |
| 1. Performing Testing | **12 days** | **Thu 03/04/14** | **Fri 18/04/14** |  |  |
| * 1. Write appropriate unit test cases for the functions each member implemented | 3 days | Thu 03/04/14 | Mon 07/04/14 | 6.14 | DucNT,LongDB,LuLP,TungNT |
| * 1. Perform unit testing and record the results | 2 days | Tue 08/04/14 | Wed 09/04/14 | 7.1 | DucNT,LongDB,LuLP,TungNT |
| * 1. Fix the bugs discovered during unit testing | 3 days | Thu 10/04/14 | Mon 14/04/14 | 7.2 | DucNT,LongDB,LuLP,TungNT |
| * 1. Complete code review checklists | 1 day | Tue 15/04/14 | Tue 15/04/14 | 7.3 | DucNT,LongDB,LuLP,TungNT |
| * 1. Report 5 – Software Test Documentation | 1 day | Wed 16/04/14 | Wed 16/04/14 | 7.4 | DucNT,LongDB,LuLP,TungNT |
| * 1. Perform integration test and fix the bugs discovered | 2 days | Thu 17/04/14 | Fri 18/04/14 | 7.5 | DucNT,LongDB,LuLP,TungNT |
| 1. Writing Users’ Manual | **5 days** | **Mon 21/04/14** | **Fri 25/04/14** |  |  |
| * 1. Write users’ manual for all the functions that each member implemented | 3.5 days | Mon 21/04/14 | Thu 24/04/14 | 7.5 | DucNT,LongDB,LuLP,TungNT |
| * 1. Check if the users’ manual and software requirements are consistent | 1.5 days | Thu 24/04/14 | Fri 25/04/14 | 8.1 | DucNT,LongDB,LuLP,TungNT |
| 1. Deploying the Website | **6 days** | **Sat 26/04/14** | **Fri 02/05/14** |  |  |
| * 1. Add more data to the database | 2 days | Sat 26/04/14 | Mon 28/04/14 |  | DucNT,LuLP,TungNT |
| * 1. Deploy the website the a real host | 0.5 days | Tue 29/04/14 | Tue 29/04/14 | 7.6 | LongDB |
| * 1. Test the website on the real host | 3.5 days | Tue 29/04/14 | Fri 02/05/14 |  | DucNT,LongDB,LuLP,TungNT |
| 1. Finalizing and closing | **4 days** | **Mon 05/05/14** | **Thu 08/05/14** |  |  |
| * 1. Prepare final project report | 3 days | Mon 05/05/14 | Wed 07/05/14 |  | TungNT,LongDB |
| * 1. Prepare final project presentation | 2 days | Mon 05/05/14 | Tue 06/05/14 |  | DucNT,LuLP |
| * 1. Final team meeting | 1 day | Thu 08/05/14 | Thu 08/05/14 |  | DucNT,LongDB,LuLP,TungNT |
| * 1. Close the project | 0 days | Thu 08/05/14 | Thu 08/05/14 |  | DucNT,LongDB,LuLP,TungNT |

# Software Requirement Specifications (SRS)



## User Requirement Specification

The system should allow 4 types of actors, namely Guest, Customer, Staff, and System Administrator, and an abstract actor named Logged User, to interact with. Each of these types of user is granted a set of functions as specified below.

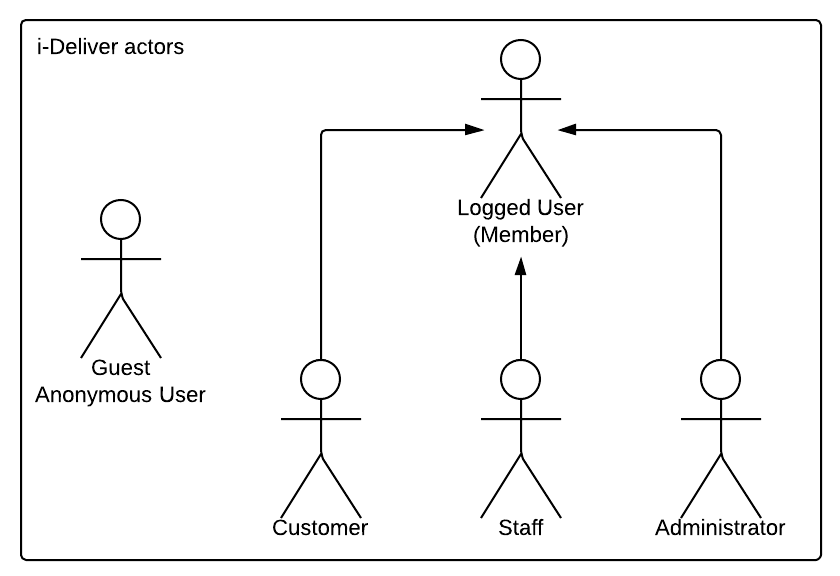


Diagram . Actor overview diagram



### Guest Requirements

A guest is an unauthenticated user of the website. He or she can:

* Log in with his authorized account;
* Register a new account;
* View, search for, and filter routes by a number of criteria that suit his or her needs;
* View details of an arbitrary routes.

### Logged User Requirements

A logged user is an authenticated user of the website (e.g., a user who logged on to the system with a valid username and password). They can:

* Log out;
* Change their own password;
* View and update their account details.

Note that that this actor is an abstract one and does not represent a real actor in practice.

### Customer Requirements

A Customer is a logged user (see 3.1.2. Logged User Requirement) and is given all the functions of a logged user. In addition, he or she is granted all the functions of a guest (see 3.1.1. Guest Requirement) except for the Log in and Register functions. A Customer also has his or her own set of functions:

* Create and submit the goods delivery requests;
* Cancel unwanted requests if they weren’t paid or approved;
* Edit submitted delivery information in a limit duration;
* Leave reviews and rate for the service of a station;
* Tracking for package (when the package left station, time left to be delivered, …);
* Make an online payment via third-party service.

### System Admin Requirements

A System Administrator is a logged user (see 3.1.2. Logged User Requirement) and is given all the functions of a logged user. In addition, a system Administrator also has his or her own set of functions:

* Manage Staff accounts;
* Add more System Administrator accounts;
* Manage fee value and calculating formula;
* Statistics: get data about revenue, performance.

### Staff Requirements

A Staff is a logged user (see 3.1.2. Logged User Requirement) and is given all the functions of a logged user. In addition, a system Administrator also has his or her own set of functions:

* Manage all requests: approve, reject online requests, create new, edit and cancel offline requests;
* Manage all related information about routes, stations, trips and coaches;
* Arrange and schedule for packages delivery;
* Searching for information about requests, routes, stations, trips and coaches;
* Manage comments and rating: delete violated comments and clear spam rating;

## System Requirement Specification (Specific Requirements)



### Overall use case

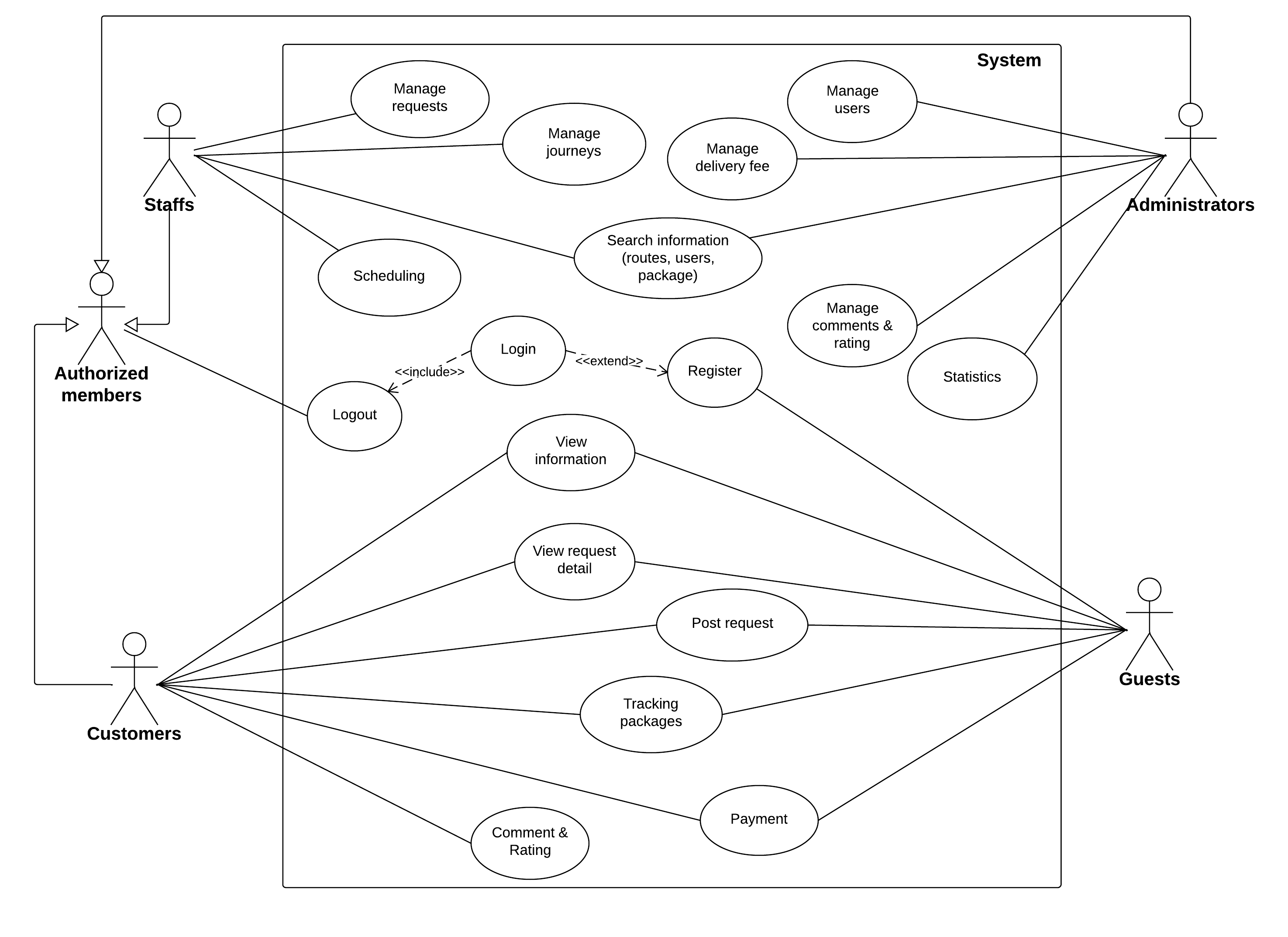


Figure . Overall use case

## Entity Relationship Diagram

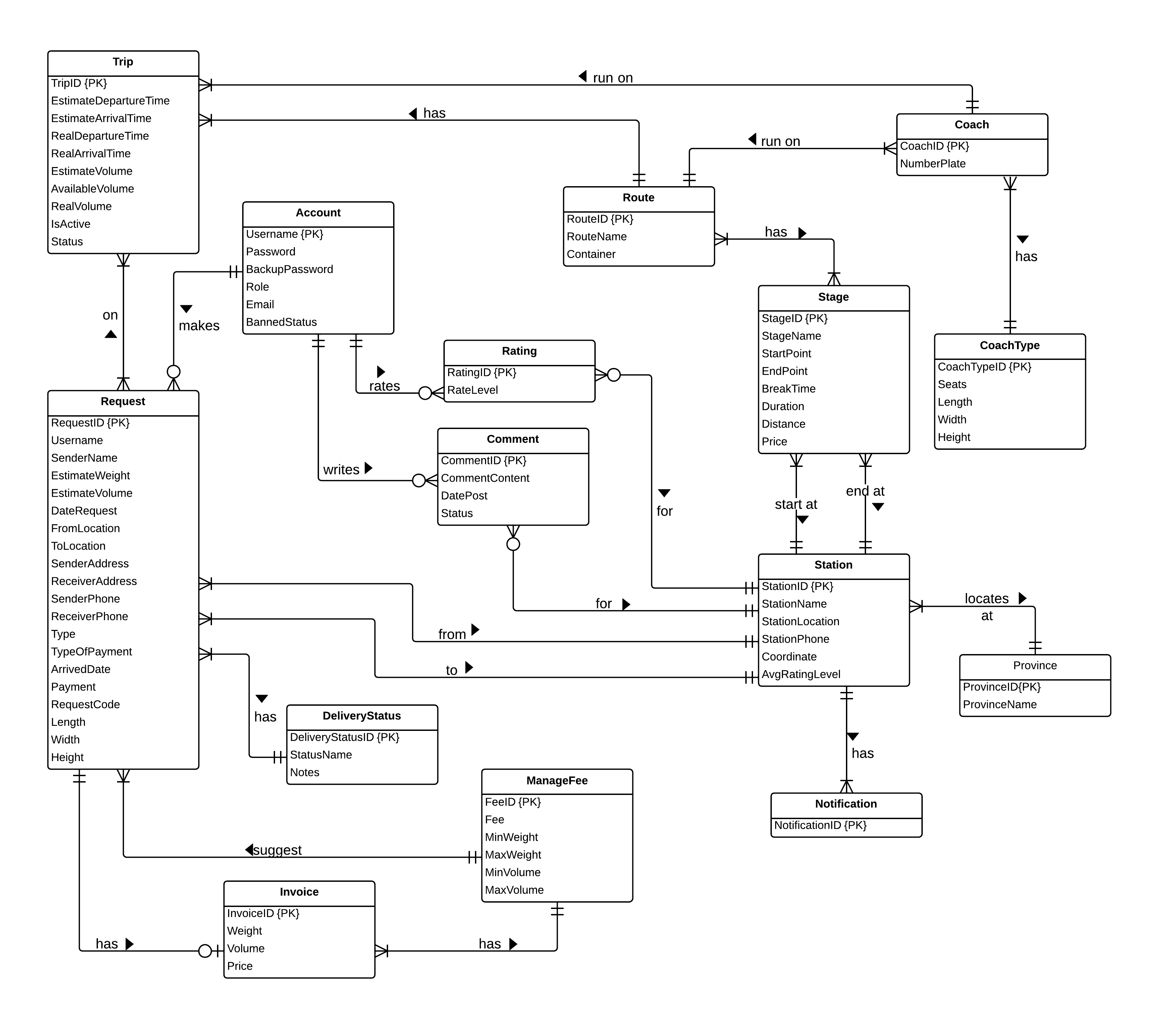


Diagram . Entity relationship diagram

# Software Design Description (SDD)



## Design Overview

The SDD describes the system architectural design and the detailed designs, including the user interfaces, of the system.

The components should communicate through interfaces. The detailed implementation of each component should be transparent to other components. The passive MVC III pattern is the preferable architecture for the website.

The detailed designs of the system should adopt the basic principles of software design, including “high cohesion, low coupling” and “open to extension, closed to modification” principles.

Each of the following sections is summarized below:

* Section 4.2: Gives a specification of the system architecture design, describing the overall architecture of the system and subsystems.
* Section 4.3: Specifies all the components that should be included in the system, and the communications between them.
* Section 4.4: Describes the class and package diagrams, as well as their descriptions in details, to provide a static view of the system.
* Section 4.4: Describes the sequence diagrams for the use cases specified in the SRS, to provide a dynamic view of the system.
* Section 4.5: Describes the user interfaces, to specify what will be seen by the users of the system.
* Section 4.5: Describes the Database Design, including the relations and the relationships between them.

## System Architectural Design

The MVC III (Model – View – Controller) pattern is used as the overall system architecture, because it specifies a clear distinction between the responsibilities of the components and is appropriate for developing web applications.

3. Exchange data

4. Results

**CONTROLLER**

* Intercepts user input;
* Coordinates the view and model;
* Handles communication between the model and data layer.

**VIEW**

* Binds to the model;
* Renders the UIs (HTML, CSS, JavaScript);
* Allows navigating between controllers.

**MODEL**

* Communicates with data source;
* Exposes functionalities, business logics, and data validation.

Database   
Server

Application Server

1. Request

8. Response

5. Select views

7. Changes made

2. Invoke methods

6. Query states

Method invocations

Change notifications

**CONTROLLER**

* Intercepts user input;
* Coordinates the view and model;
* Handles communication between the model and data layer.

**VIEW**

* Binds to the model;
* Renders the UIs (HTML, CSS, JavaScript);
* Allows navigating between controllers.

**MODEL**

* Communicates with data source;
* Exposes functionalities, business logics, and data validation.

Database   
Server

Application Server

1. Request

8. Response

5. Select views

7. Changes made

2. Invoke methods

6. Query states

Method invocations

Change notifications

**CONTROLLER**

* Intercepts user input;
* Coordinates the view and model;
* Handles communication between the model and data layer.

**VIEW**

* Binds to the model;
* Renders the UIs (HTML, CSS, JavaScript);
* Allows navigating between controllers.

**MODEL**

* Communicates with data source;
* Exposes functionalities, business logics, and data validation.

Database   
Server

Application Server

1. Request

8. Response

5. Select views

7. Changes made

2. Invoke methods

6. Query states

Method invocations

Change notifications

Figure . Model-View-Controller Architecture

## Component Diagram

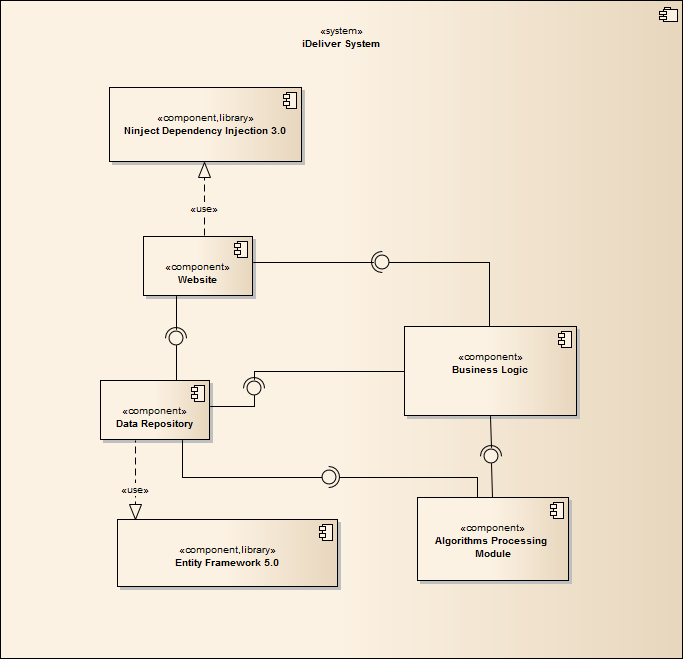


Diagram . Component Diagram

The i-Deliver system includes the following major components:

* **Entity Framework 5.0:** An object-relational mapper that enables working with relational data persistence using domain-specific objects;
* **Data Repository:** A data store that provides data access functionality, using the Entity Framework to communicates with the database;
* **Business Logic:** A module that uses interfaces exposed by the Data Repository to encapsulate important businesses of in the i-Deliver system;
* **Ninject Dependency Injection 3.0:** A lightweight dependency injection framework for .NET applications which helps split the application into a collection of loosely-coupled, highly-cohesive pieces, and then glue them back together in a flexible manner.
* **Website:** A web-application that helps exposes all the functionalities to end-users.
* **Algorithms Processing Module:** A module that uses Interface and DataRepository to execute the business of system by processing through some relative entities and find out the needed connections between them that satisfying the given constraint.

From the point of view of the MVC-III pattern, the components Entity Framework 5.0, Data Repository, Business Logic, and Ninject Dependency Injection 3.0 all belong to the Model part. The Controller and the View parts are wrapped in the component Website.

## Database Design

### Logical database design

From the ERD specified in the SRS section, logical database design includes the following relations:

|  |  |  |
| --- | --- | --- |
| **Index** | **Table Name** | **Description** |
| 1 | Account | List of user that registered to the system with valid username and password. |
| 2 | UserInfo | Detail information of each user. |
| 3 | Coach | List of all coach the center has. |
| 4 | CoachType | List of type of coach base on the number of seat that the coach has. |
| 5 | Route | List of route that center can deliver to, including the route name. |
| 6 | Trip | Information of trip of each coach the center has every day. |
| 7 | Station | The list of all stations the center has. |
| 8 | Schedule | The schedule of coach for run on the specific route during a day. |
| 9 | Request | Information of each request that user posted to the system. |
| 10 | DeliveryStatus | Status of the request of base on the action of user and staff. |
| 11 | Invoice | Detail of invoice corresponding to the approve request. |
| 12 | ManageFee | The fee of service base on the range of volume and range of weight. |
| 13 | Comment | Content of comment of user posted for each station. |
| 14 | Rating | Rating level that user rate for each station. |
| 15 | Assigning | Table of mapping between coach and request. |
| 16 | Stage | The partial of the way of route, each stage contains the start and end station and addition information of this stage. |
| 17 | RouteStage | Mapping between route and stage |
| 18 | Notification | Check whether user has seen the notification or not yet. |
| 19 | Province | List of provinces that the system has some station in there. |



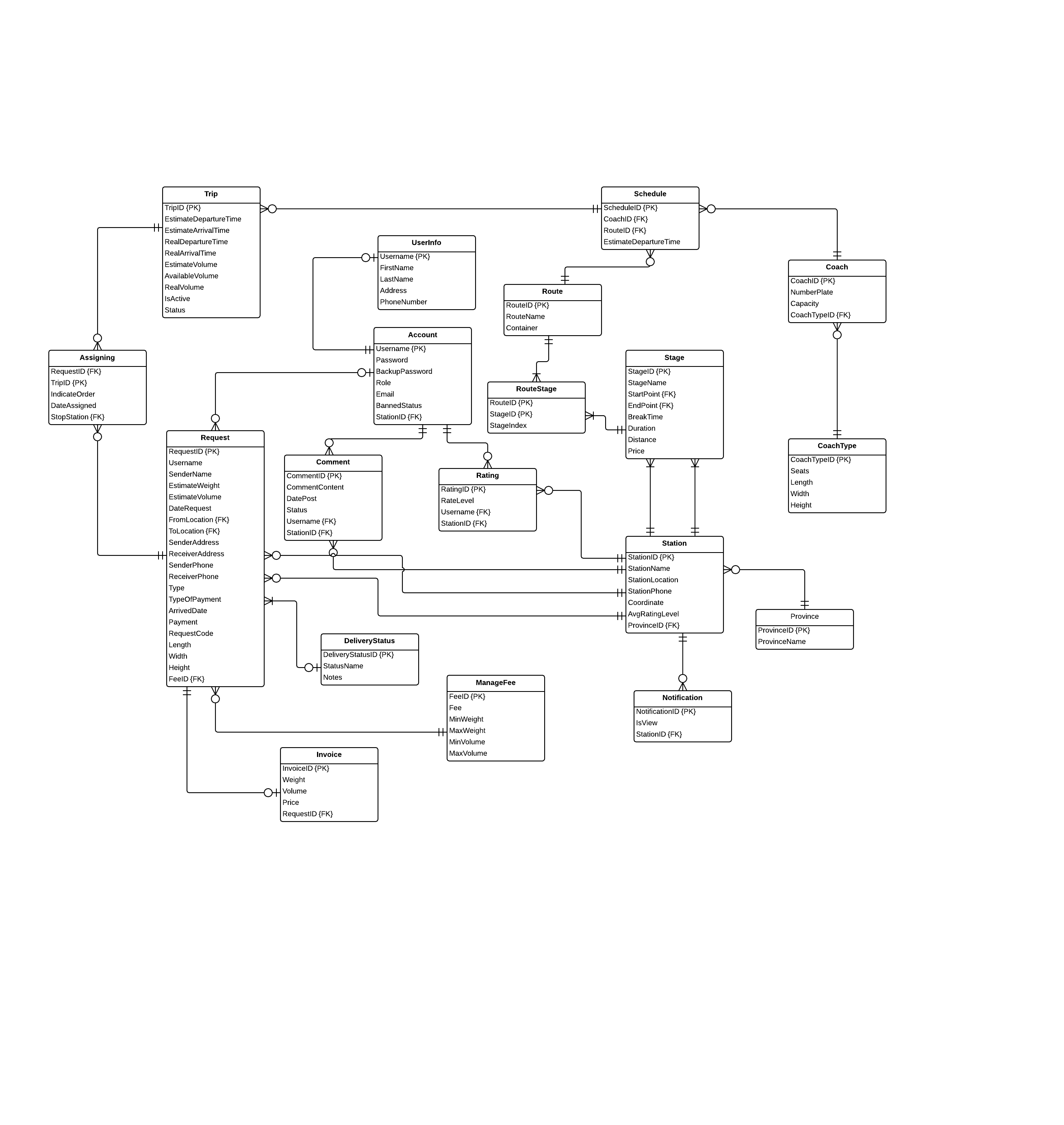


Figure . Logical database design

The details of each relation are specified below

#### User

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| Username | int |  | - | PK | Username of user |
| Password | nvarchar(50) |  | - |  | Password of user |
| BackupPassword | nvarchar(50) |  | - |  | Backup password for user |
| Role | nvarchar(50) |  | - |  | Role of user in the system:  customer, admin or staff |
| Email | nvarchar(50) |  | - |  | Email of user |
| Phone | nvarchar(50) |  | - |  | Phone of user |
| BannedStatus | bit |  | - |  | User is banned or not |
| StationID | int |  | 11 | FK | The station of the staff belong to (just for staff account) |

Unique: Username.

Foreign key: StationID.

#### UserInfo

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| Username | nvarchar(50) |  | - | FK | Username of user registered to the system or the staff |
| Firstname | nvarchar(50) |  | - |  | Firstname of user registered to the system or the staff |
| Lastname | nvarchar(50) |  | - |  | Lastname of user registered to the system or the staff |
| Address | nvarchar(500) |  | - |  | Address of user registered to the system or the staff |

Unique: Username.

Foreign key: N/A.

#### Coach

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| CoachID | int |  | Auto generated | PK | Uniquely identifies of the coach |
| NumberPlate | nvarchar(50) |  | - |  | Number plate of coach |
| CoachTypeID | int |  | - | FK | The type of coach |

Unique: CoachID, NumberPlate.

Foreign key: CoachTypeID (table CoachType).

#### CoachType

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| CoachTypeID | int |  | Auto generated | PK | Uniquely identifies of the coach |
| Seats | int |  | - |  | The number of seats of the coach, that define the type of coach |
| Capacity | float |  |  |  | The percentage of the cabin that coach can be serving. |

Unique: Seats.

Foreign key: N/A.

#### Route

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| RouteID | int |  | Auto generated | PK | Uniquely identifies a route |
| RouteName | nvarchar(50) |  | - |  | The name of route |

Unique: RouteID.

Foreign key: StartPoint (table Station), EndPoint (table Station).

#### Trip

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| TripID | int |  | Auto generated | PK | Uniquely identifies a route |
| EstimateDepartureTime | time |  | - |  | The estimate time for the trip departures in each day |
| RealDepartureTime | time |  | - |  | The real time that the trip departures in a specific day |
| EstimateArrivalTime | time |  | - |  | The estimate time for the trip departures in each day |
| RealArrivalTime | float |  |  |  | The real time for the trip arrival to the station in a specific day |
| AvailableVolume | float |  |  |  | The volume that the coach on this trip can deliver |
| Date | date |  |  |  | The date that trip start. |
| RouteID |  |  |  | FK | The route that the trip traveling on |
| CoachID |  |  |  | FK | The coach of this trip. |

Unique: RouteID.

Foreign key: RouteID (table Route), CoachID (table Coach).

#### Station

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| StationID | int |  | Auto generated | PK | Uniquely identifies a route |
| StationName | nvarchar(50) |  | - |  | The name of the station |
| StationLocation | nvarchar(50 |  | - |  | The address of location |
| BreakTime | float |  | - |  | The default break time that for each trip come to this station |
| StationPhone | nvarchar(50) |  | - |  | The telephone number of this station |
| Description | nvarchar(500) |  | - |  | The description for this station |
| Longitude | float |  | - |  | The longitude of the station location |
| Latitude | float |  | - |  | The latitude of the station location |
| Province | nvarchar |  | - |  | The province of the station location |

Unique: StationID.

Foreign key: N/A.

#### Schedule

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| ScheduleID | int |  | Auto generated | PK | Uniquely identifies the volume |
| CoachID | Int |  | - |  | Uniquely identifies the coach used to scheduling |
| RouteID | Int |  | - |  | Uniquely identifies the route to be scheduled |
| EstimateDepartureTime | time |  | - |  | The minimum volume the coach can be deliver on the route |
| EstimateArrivalTime | time |  | - | FK | The route in this volume define |

Unique: N/A.

Foreign key: CoachID (table Coach), RouteID (table Route).

#### Request

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| ScheduleID | int |  | Auto generated | PK | Uniquely identifies the volume |
| ArrangeID | float |  | - |  | The maximum volume the coach can be deliver on the route |
| EstimateDepartureTime | time |  | - |  | The minimum volume the coach can be deliver on the route |
| EstimateArrivalTime | time |  | - | FK | The route in this volume define |

Unique: N/A.

Foreign key: ArrangeID (table CoachArrangement).

#### DeliveryStatus

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| DeliveryStatusID | Int |  | Auto generated | PK | Uniquely identifies the delivery status |
| StatusName | nvarchar(50) |  | - |  | The name of delivery status of the package |
| Note | nvarchar(500) | X | - |  | Some more description for the status |

Unique: DeliveryStatusID.

Foreign key: N/A.

#### Invoice

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| InvoiceID | Int |  | Auto generated | PK | Uniquely identifies an invoice |
| Weight | float | X | - |  | Weight of package |
| Volume | int | X | - |  | Volume of package |
| Price | float |  | - |  | Price of delivery service |
| RequestID | int |  | - | FK | The request that the invoice point to |

Unique: InvoiceID.

Foreign key: RequestID (table Request).

#### ManageFee

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| FeeID | int |  | Auto generated | PK | Uniquely identifies a fee rule |
| Fee | float |  | - |  | The fee of this weight and volume range |
| MinWeight | float |  | - |  | The minimum weight of the weight range |
| MaxWeight | float |  | - |  | The maximum weight of the weight range |
| MinVolume | float |  | - |  | The minimum volume of the volume range |
| MaxVolume | float |  | - |  | The maximum volume of the volume range |

Unique: FeeID.

Foreign key: N/A.

#### Comment

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| CommentID | int |  | Auto generated | PK | Uniquely identifies the comment |
| Username | nvarchar(50) |  | - | FK | The user post comment |
| CommentContent | nvarchar(4000) |  | - |  | Content of comment |
| DatePost | date |  | - |  | Date that user post the comment |
| Status | bit |  | - |  | Approve or reject the comment |
| StationID | int |  | - | FK | The station that customer comment for |

Unique: CommentID.

Foreign key: Username (table User), StationID (table Station).

#### Rating

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| RatingID | int |  | Auto generated | PK | Uniquely identifies the rating |
| Username | nvarchar(50) |  | - | FK | The user post comment |
| RateLevel | int |  | - |  | The level of rate: from 1 to 5 |
| StationID | int |  | - | FK | The station that user rating for |

Unique: RatingID.

Foreign key: Username (table User), StationID (table Station).

#### Assigning

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| CoachID | int |  | - | PK | The ID of coach for the specific request |
| RequestID | int |  | - | The ID of request |
| IndicateOrder | int |  | - |  | The indicated order of the assigning the package of this request |

Unique: (CoachID, RequestID).

Foreign key: N/A.

#### Stage

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| CoachID | int |  | - | PK | The ID of coach |
| StartPoint | int |  | - | FK | The start position (station) of the stage |
| EndPoint | int |  | - | FK | The end position (station) of the stage |
| Duration | float |  | - |  | The estimate duration (in hours) for moving on this route |
| Distance | float |  | - |  | The estimate distance (in km) between start and end location |
| Container | float |  | - |  | The percentage of coach that allow to travel on the stage |
| Price | float |  | - |  | The fee for travel on the stage |

Unique: N/A.

Foreign key: StartPoint, EndPoint.

#### RouteStage

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| RouteID | int |  | - | PK | The ID of route |
| StageID | int |  | - | The ID of stage |
| StageIndex | int |  | - |  | The index of stage in the list stage of the route |

Unique: N/A.

Foreign key: N/A.

#### Notification

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| ID | int |  | **-** | PK | The index of notification |
| Username | Int |  | - |  | Username of user |
| StationID | int |  | - |  | The ID of station the user mange’s |
| isView | bit |  | - |  | Status of notification, it is seen by user or not. |

Unique: N/A.

Foreign key: StationID, Username.

#### Province

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| ProvinceID | int |  | **-** | PK | The index of province |
| ProvinceName | nvarchar(50) |  | - |  | Name of province |

Unique: N/A.

Foreign key: N/A.

### Physical database design

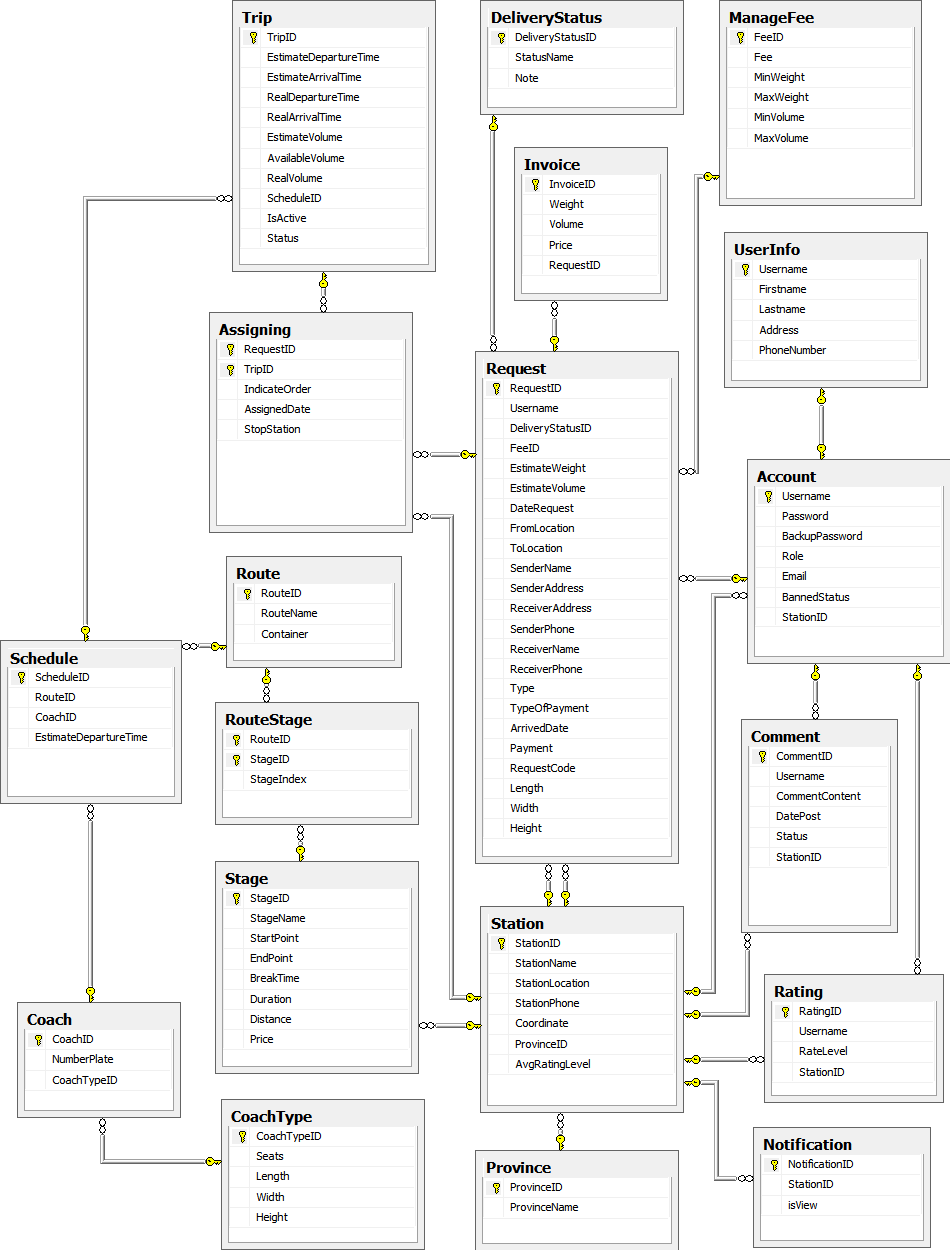


Figure . Physical database design

# Appendix

## Reference

<http://en.wikipedia.org/wiki/Model-view-controller>

<http://www.visual-paradigm.com/product/vpuml/tutorials/>

<http://www.packtpub.com/mastering-ninject-for-dependency-injection/book>

<http://en.wikipedia.org/wiki/Waterfall_model>

<http://www.asp.net/mvc/tutorials/mvc-4>

<http://msdn.microsoft.com/en-us/library/vstudio/ff926074.aspx>

<http://www.csharpfriends.com/articles/getarticle.aspx?articleid=336>

<http://www.uml-diagrams.org/>

<http://en.wikipedia.org/wiki/Database_design>