

**Dispose Pick and Load** Expression

Date of Preparation

30-03-2022

Table of Contents

[1 Requirement Positioning 4](#__RefHeading___Toc4695_201796809)

[​ 1.1 UI Flow 4](#__RefHeading___Toc4697_201796809)

[​ 1.1.1 Driver Dashboard | Dumper Dashboard 4](#__RefHeading___Toc3826_674525685)

[​ 1.2 Routing Def 6](#__RefHeading___Toc4699_201796809)

[​ 1.3 UI Def 7](#__RefHeading___Toc4701_201796809)

[​ 1.4 Test Cases 8](#__RefHeading___Toc4703_201796809)

[​ 1.5 Time Allotment 9](#__RefHeading___Toc4756_201796809)

[​ 1.5.1 Feature Time Allotment (Total Time: 12 hour) 9](#__RefHeading___Toc4705_201796809)

[​ 1.5.2 Routing Time Allotment 10](#__RefHeading___Toc1952_1253115458)

[​ 1.5.3 Work Unit Time Allotment 10](#__RefHeading___Toc545_1253115458)

[​ 2 Requirement Analysis 11](#__RefHeading___Toc4707_201796809)

[​ 2.1 Business Event Table 11](#__RefHeading___Toc4709_201796809)

[​ 2.2 Expression Table 14](#__RefHeading___Toc4711_201796809)

[​ 2.3 Post Analysis Event Table 15](#__RefHeading___Toc4713_201796809)

[​ 2.4 Swim Line 18](#__RefHeading___Toc4715_201796809)

[​ 2.5 URI Def Table 19](#__RefHeading___Toc4717_201796809)

[​ 2.6 Frontend Models 20](#__RefHeading___Toc4733_201796809)

[​ 2.7 ERD 21](#__RefHeading___Toc4735_201796809)

[​ 2.8 Artifact Position Table 22](#__RefHeading___Toc4752_201796809)

[​ 3 Time Estimation 23](#__RefHeading___Toc4725_201796809)

[​ 3.1 Feature Time Estimation 23](#__RefHeading___Toc4731_201796809)

[​ 3.2 Routing Time Estimation 24](#__RefHeading___Toc1952_12531154581)

[​ 3.3 Work Unit Time Estimation 24](#__RefHeading___Toc545_12531154581)

[​ 3.4 Test Script Time Estimation 24](#__RefHeading___Toc1950_1253115458)

[​ Appendix 26](#__RefHeading___Toc1976_1253115458)

[​ Appendix A: Weekly Work Schedule 26](#__RefHeading___Toc14189_3230761449)

[​ Appendix B: Weekly Deliverables 26](#__RefHeading___Toc14191_3230761449)

[​ Appendix C: Operation Flow 27](#__RefHeading___Toc14193_3230761449)

[​ Appendix D: Definitions 28](#__RefHeading___Toc14195_3230761449)

[​ Appendix E: UI Building Blocks 28](#__RefHeading___Toc14197_3230761449)

[​ Appendix F: Dependencies 30](#__RefHeading___Toc14199_3230761449)

[​ Appendix G: Understanding Approach 30](#__RefHeading___Toc14201_3230761449)

[​ Appendix H: Express Approach 30](#__RefHeading___Toc14055_3230761449)

[​ Appendix I: Document Preparation Flow 31](#__RefHeading___Toc6344_275431948)

[​ Appendix J: Coding Approach 31](#__RefHeading___Toc14203_3230761449)

[​ Appendix K: Document Structure Definition 32](#__RefHeading___Toc14205_3230761449)

[​ Appendix L: Coding Norms and Principles 36](#__RefHeading___Toc6891_275431948)

[​ Appendix M: Work Flow 47](#__RefHeading___Toc8895_294645069)

[​ Appendix N: Time Allotment Guideline 48](#__RefHeading___Toc1946_1253115458)

[​ Put and Analysis Time Allotment 48](#__RefHeading___Toc1948_1253115458)

[​ Component Time Allotment 48](#__RefHeading___Toc549_1253115458)

[​ Appendix O:DXR Project Policies 49](#__RefHeading___Toc8845_294645069)

[​ DXR Language Competency Policy 49](#__RefHeading___Toc8847_294645069)

[​ Language Competency in Front-End 49](#__RefHeading___Toc800_1710162942)

[​ Sequence Diagram 50](#__RefHeading___Toc802_1710162942)

[​ Return JSON: 50](#__RefHeading___Toc804_1710162942)

[​ Policy 51](#__RefHeading___Toc630_4264418900)

[​ References 51](#__RefHeading___Toc646_4264418900)

[​ Language Competency In Database 52](#__RefHeading___Toc806_1710162942)

[​ Policy 52](#__RefHeading___Toc650_4264418900)

[​ Backend REST URL Policy 53](#__RefHeading___Toc8849_294645069)

[​ Web Application Module Policy 54](#__RefHeading___Toc8851_294645069)

[​ Appendix P: Sample Tables 55](#__RefHeading___Toc4663_201796809)

[​ Sample Routing Def Table 55](#__RefHeading___Toc4754_201796809)

[​ Sample Event Table 56](#__RefHeading___Toc4665_201796809)

[​ Sample URI Table 57](#__RefHeading___Toc4667_201796809)

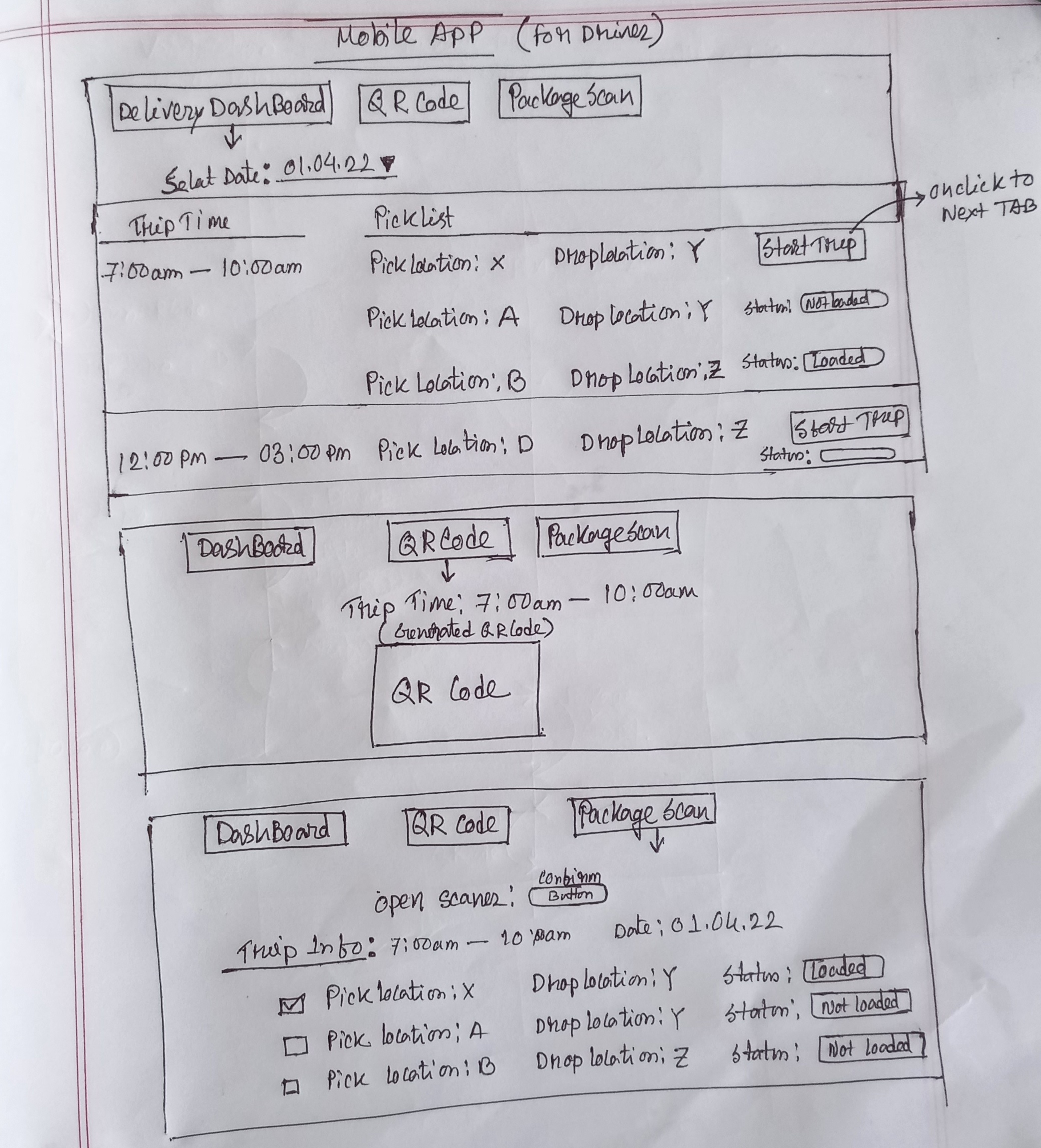
[​ Sample Frontend Model 57](#__RefHeading___Toc4669_201796809)

[​ Sample Artifact Position Table 58](#__RefHeading___Toc4671_201796809)

# 1 Requirement Positioning

## 1.1 UI Flow

### 1.1.1 Driver Dashboard



### 1.1.2 Driver Dashboard Sequence Diagram

## 1.2 Routing Def

## 1.3 UI Def

## 1.4 Test Cases

## 1.5 Time Allotment

### 1.5.1 Feature Time Allotment (Total Time: 12 hour)

|  |  |  |  |
| --- | --- | --- | --- |
| **Work Items** | **Tasks** | **Allocated Time** | **Responsible Person** |
| Put and Analysis | UI Flow, UI Def, Test Cases, Time Estimation | 30 min | Business Lead |
| Biz Table | 30 min | Frontend Developers |
| Expression Table, Post Analysis Event Table, Swim Line | 1 hr | Development Lead |
| URI, Frontend Model | Frontend Developers |
| Domain | ERD, Backend Models and Repo | 1 hr | Development Lead, Backend Developers |
| Data | Frontend Sample Data and Mock Service | 2 hr | Business Lead, Frontend Developers |
| Backend Sample Data and Mock Service | Business Lead, Backend Developers, Development Lead |
| Compilation | Compilation, Module and Artifact Positioning, Branching | 1 hr | Business Lead, Development Lead, Frontend Developers, Backend Developers |
| **Requirement Positioning Time Allotment** | | **6 hr** |  |
| Code Writeup | Routing | 1 hr | Frontend Developers |
| Frontend Work Unit | 2 hr | Frontend Developers |
| Backend Work Unit | Backend Developers |
| Calibration | 1 hr | Frontend Developers, Backend Developers |
| **Code Writeup Time Allotment** | | **4 hr** |  |
| Review | Review | 1 hr | Business Lead, Development Lead, Frontend Developers, Backend Developers |
| Check and Merge | 1 hr | Business Lead, Development Lead, Frontend Developers, Backend Developers |
| **Review Time Allotment** | | **2 hr** |  |

### 1.5.2 Routing Time Allotment

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1. **Item** | 1. **Menu** | 1. **Sub Menu** | 1. **Tab** | 1. **Time** |
|  |  |  |  |  |
|  |  |  |  |  |

### 1.5.3 Work Unit Time Allotment

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Work Unit** | **Info Menu** | **List** | **Popup** | **Form** | **Time** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

# 2 Requirement Analysis

## 2.1 Business Event Table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1. **SN.** | 1. **Component** | 1. **Event** | 1. **Input** | 1. **Output** | 1. **Time** | 1. **Remark** |
|  | Delivery Dashboard | onLoad() | CompanyId, userId, currentDate | tripList | 30m |  |
|  | dateWiseTripList | CompanyId, userId, currentDate | tripList |  |  |
|  | generateQRcode | Trip and picks | QR code | 10m |  |
|  | scanQRcode | QR Code | Pick details | 30m |  |
|  | verifyPackageLoad | Pick details | pickList | 30m |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  | Total = | 1.40 hr |  |

## 2.2 Expression Table

|  |  |  |
| --- | --- | --- |
| **SN.** | **Title** | **Content** |
|  | External  Dependencies | * Project Schedule |
|  | Business  Constraints |  |
|  | Technical  Constraints |  |
|  | Functions |  |
|  | Findings |  |
|  | Unexplored Issue |  |
|  | Deliverable |  |
|  | Notes |  |
|  | Questions |  |

## 2.3 Post Analysis Event Table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1. **SN.** | 1. **Component** | 1. **Event** | 1. **Input** | 1. **Output** | 1. **Time** | 1. **Remark** |
|  |  |  |  |  |  |  |
|  |  |  |  | Total = 15.5 hr | |  |

## 2.4 Swim Line

## 2.5 URI Def Table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SN.** | **Event** | **URI** | **Request Type** | **Input** | **Output** | **Time** |
|  | getDriverTripList | driver/trip-list | POST | CompanyId, driverId, date | tripList | 1hr |
|  | getPicks | driver/pick-list | POST | tripId | pickList | 30m |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  | Total = | 1.5 hr |

Converters = 1 hr

Total = 2.5 hr

## 2.6 Frontend Models

## 2.7 ERD

## 2.8 Artifact Position Table

|  |  |  |  |
| --- | --- | --- | --- |
| **Application** | **Module** | **Items** | **Path** |
| Web Application | Visitor |  |  |
| System Admin |  |  |
| Dumpers Admin |  |  |
| Processor Admin |  |  |
| Transporter Admin |  |  |
| Drivers |  |  |
| Dumper Account |  |  |
| Processor Account |  |  |
| Transporter Account |  |  |
| Company Admin |  |  |
| Common Directive |  |  |
| Common Util | Common Util |  |  |
|  |  |  |  |

# 3 Time Estimation

## 3.1 Feature Time Estimation

|  |  |  |  |
| --- | --- | --- | --- |
| **Work Items** | **Tasks** | **Allocated Time** | **Estimated Time** |
| Put and Analysis | UI Flow, UI Def, Test Cases, Time Estimation | 30 min | 1 hr |
| Biz Table | 30 min | 30 min |
| Expression Table, Post Analysis Event Table, Swim Line | 1 hr | 1 hr |
| URI, Frontend Model | 45 min |
| Domain | ERD, Backend Models and Repo | 1 hr | 1 hr |
| Data | Frontend Sample Data and Mock Service | 2 hr |  |
| Backend Sample Data and Mock Service | 1.5 hr |
| Compilation | Compilation, Module and Artifact Positioning, Branching | 1 hr | 1 hr |
| **Requirement Positioning Time Estimation** | | **6 hr** | **6 hr 45 min** |
| Code Writeup | Routing | 1 hr | 30 min |
| Frontend Work Unit | 2 hr | 3 hr |
| Frontend Event Handle |  | 1.5 hr |
| Frontend Testing |  |  |
| Backend Work Unit |  | 3 hr |
| Calibration | 1 hr | 2 hr |
| **Code Writeup Time Estimation** | | **4 hr** | **10 hr** |
| Review | Review | 1 hr | 1 hr |
| Check and Merge | 1 hr | 1 hr |
| **Review Time Estimation** | | **2 hr** | **2 hr** |

## 3.2 Routing Time Estimation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1. **Item** | 1. **Menu** | 1. **Sub Menu** | 1. **Tab** | 1. **Time** |
| Driver Dashboard |  |  | 1. 3 | 1. 30m |

## 3.3 Work Unit Time Estimation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Work Unit** | **Info Menu** | **List** | **Popup** | **Form** | **Time** |
| Driver Dashboard |  | 1 |  |  | 1hr |
| QR Code |  |  |  |  |  |
| Package Scan |  | 1 |  |  | 1hr |
|  |  |  |  |  | Total = 2 hr |

# 3.4 Test Script Time Estimation

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case** | **Test Script** | **Suit Name** | **Time** |
|  |  |  |  |
|  |  |  | Total = hr |

# Appendix

## Appendix A: Weekly Work Schedule

|  |  |  |  |
| --- | --- | --- | --- |
| 1. Day | 1. Task | 1. Start | 1. Slot |
| 1. Thursday | 1. 1. Fixing Work 2. 2. Put 3. 3. Express 4. 4. Work Schedule | 1. 10.30 AM | 1. 3 hour |
| 1. Sunday | 1. Trigger Review 2. (personally) | 1. 4 PM | 1. 45 min |
| 1. Tuesday | 1. Progress Review | 1. 3 PM | 1. 1.5 hour |
| 1. Wednesday | 1. Submission | 1. 10.30 AM | 1. 3 hour |
| 1. Everyday 2. (Except Wednesday) | 1. 1. Follow Up 2. 2. Plan 3. 3. Work 4. 4. Learning Capture | 1. 9 AM | 1. 9 hour |

## Appendix B: Weekly Deliverables

1. 1. Put :
2. UI
3. Rules
4. Sample
5. Process Flow
6. 2. Receive :
7. Process Flow
8. Constraints
9. External Dependencies
10. Deliverables
11. Findings
12. 3. Express :
13. Technical Workout
14. Operation Flow
15. Technical Workout :
16. View
17. Components
18. Data Fetch
19. View Data Preparation
20. Data Persist

## Appendix C: Operation Flow

1. Details given in Appendix K
2. 

## Appendix D: Definitions

1. **Different Types of Data**
2. 1. Business Data: All Data that are subjected to store/represent.
3. 2. Operational Data: From development point of view the data generated from assigned scope of work.
4. 3. Mock Data:
5. → Unavailable external data.
6. → Place the hard codded mock data in expected format in mock function using *callBack*.
7. → that hard codded data will be replaced by services owning the actual business data.
8. → mock service/function must confirm the signature defined by data manager, that is callback and params in specific order as its parameter.
9. 4. Array of Primitive Value: An array of integer/string/double type data.
10. 5. Inner Path Candidate: Object or array of objects
11. **Path Expression Guideline**
12. Path expression is the value of ‘path’ parameter in data get and save functions.
13. Path expression is a String
14. Path expression for root as bellow–
15. var path = “/[TECHDISER\_ID=” + rootObject.TECHDISER\_ID + “]”;
16. To save data into inner path of data structure path will extend up to inner attribute name plus object’s TECHDISER\_ID–
17. var path = “/[TECHDISER\_ID=” + rootObject.TECHDISER\_ID + “]/attributeName/[TECHDISER\_ID=” + innerObject.TECHDISER\_ID + “]”;
18. All sorts of required parameters are to be sent with param array

## Appendix E: UI Building Blocks

1. **1. View:** User Interface
2. **2. Components :** Sections of user interface that can be identified distinctly. So components are building blocks of view
3. **3. Controls :**
4. → Items like buttons, text-box, labels etc.
5. → These are the building blocks of components
6. → These provide the means of user interaction
7. **4. Events :** User interactions to user interface
8. **5. Services :**
9. 1. Service :
10. → It performs specific task
11. → These are placed in factories
12. → These are exposed with interface manager
13. → Each Json is handled by a specific factory or service container
14. → So services are the building blocks of factories
15. → Services can interact with other services through interface manager
16. 2. External Service : Any service beyond current working space

## Appendix F: Dependencies

1. **External :** Services, owned by other factories or work space, used for fetching data or performing external operations.
2. **Internal :** Services, providing data or doing operations with in same operational space but difference work space.
3. **Deliverables/Provider Service** : Services, used by services of different factories or operational space for providing data or performing operations on behalf.
4. Others:
5. **1. Flow** : The sequences of components and view that describe an operation.
6. **2. Constraints :** The rules to be conformed by processes.
7. Data space : Data space means Jsons.

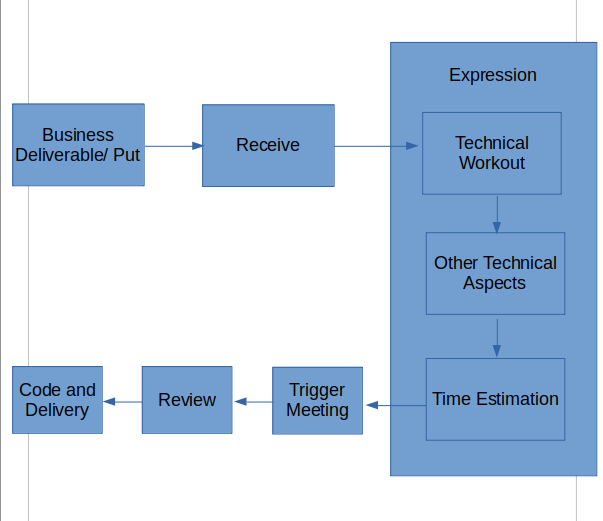
## Appendix G: Understanding Approach

1. step 1. Understand business,
2. step 2. Understand views,
3. step 3. Analyze business and views/UI,
4. step 4. Find odds and confusion,
5. step 5. After clarifying odds or confusions, go for requirement approach.

## Appendix H: Express Approach

1. step 1. Draw out data flow according to view.
2. step 2. Draw out external dependencies.
3. step 3. Draw out events and operations.
4. step 4. Express business problems in Data space (problem-to-data).
5. step 5. Enlist required functions to solve the problem from data space to problem space (data-to-problem).
6. Step 6. Express business rules and constraints in terms of Functions
7. step 7. Express operations with swim line.

## Appendix I: Document Preparation Flow

1. 

## Appendix J: Coding Approach

1. step 1. Confirm business requirements, understandings and expressions.
2. step 2. Align with the coding approach.
3. step 3. Make algorithms for difficult operations.
4. step 4. Start coding.
5. step 5. If problem, discuss with colleagues and find solution.
6. Step 6. note down the problem and solutions(capture learnings).

## Appendix K: Document Structure Definition

1. **Introduction**
2. In this section we describe Feature, Sub-goals, Chapters, Technology and Tools.
3. Feature: A feature is a complete operation which serve a business goal.
4. Sub-goal: If we divide a feature operation into some levels then those are sub-goals of the feature.
5. Chapter: A chapter consist one or more sub-goal.
6. Technology and Tools: Technology and tools to be used for developing the feature.
7. **Steps and Goals**
8. A table of content which describes Chapters, their sub-goals and steps of those sub-goals.
9. Then we analyze business and technical aspects of the sub-goals and document as bellow sections-
10. **1.1 Steps and Goals**
11. ‘Steps and Goals’ table of the chapter.
12. **1.2 UI**
13. All UI drawings.
14. **1.3 UI Work Flow**
15. How UI components will incorporate in code
16. **1.4 Workspace Routing**
17. How user will navigate to a UI
18. **1.5 Expression Table**
19. In Expression Table we mention External Dependencies, Business Constraints, Technical Constraints, Functions , Findings, Unexplored Issue, Deliverable and Note. Lets understand these items one by one–

|  |  |  |  |
| --- | --- | --- | --- |
| 1. **Sl.#** | 1. **Section** | 1. **Purpose** | 1. **Responsible Person** |
|  | 1. External Dependencies | 1. Necessary external data for the sub-goal. Data provided by other sub-goal or feature are external data. | 1. Business 2. Technical |
|  | 1. Business Constraints | 1. Important business condition/demand/statement given by business person. | 1. Business |
|  | 1. Technical Constraints | 1. Important technical condition/demand which are different than usual and vital to achieve standard quality. | 1. Technical |
|  | 1. Functions | 1. Complex functions which needs to be discussed with both technical and business lead. We describe function steps in terms of sentence. | 1. Technical |
|  | 1. Findings | 1. If you found any gaps or broken links in business mention then is this section. | 1. Technical |
|  | 1. Unexplored Issue | 1. If there is any technical aspect/tool needed to develop this sub-goal mention those in this section. | 1. Technical |
|  | 1. Deliverable | 1. If there is any technical aspect/tool needed to develop this sub-goal mention those in this section. | 1. Technical |
|  | 1. Note | 1. Something close to usual but need to mention. |  |

1. **1.6 Operation Flow**
2. Operation flow is a flow diagram of Data to Operation then Operation to Data. There is a defined structure to draw Operation Flow. Lets understand components of Operation Flow–

|  |  |  |  |
| --- | --- | --- | --- |
| 1. **Component** | 1. **Purpose** | 1. **Content Type** | 1. **Note** |
| 1. Data Container | 1. Data JSON for the operation | 1. Data |  |
| 1. Param Data | 1. The Data we get from another workspace/controller/service by param/state/url is param data | 1. Data |  |
| 1. Fetch External Service Data/ Mock Data | 1. The data we get from service of another feature/sub-goal | 1. Operation/ Data |  |
| 1. Fetch Own Service Data | 1. The data we get from own service | 1. Operation |  |
| 1. Prepare Pre-render Data | 1. Prepared data for rendering view | 1. Operation |  |
| 1. Render View | 1. View we get from data set. | 1. View |  |
| 1. Prepare Pre-persist Data | 1. Prepared data for save. | 1. Operation |  |
| 1. Persist | 1. Data JSON for data save | 1. Data |  |

1. **1.7 Data Structure**
2. Add all ‘Save Data’ and ‘Mock Data’ structure for the operation.
3. [Note: mark ‘Mock Data’ title with asterisk symbol ( \* ) ]
4. **1.8 Data Flow Diagram**
5. In this section we describe all save operations with bellow information.
6. Task: UI route of the task from which save operation generate
7. JSON: Data JSON name
8. Path: Data save path
9. Sample Data: Sample save data
10. **1.9 Derivation**
11. Derivation or source of data to prepare a view component. We show derivations by a table which describes View Component, UI Routing, JSON/Mock and Path. Lets understand these items one by one-
12. View Component: View component generated from data
13. UI Routing: UI Route of the view component
14. JSON/Mock: Data JSON or Mock data name
15. Path: Path of the data in data structure.
16. **1.10 Technical Workout**
17. In this table we gather all the coding aspect. Lets understand table columns of the table one by one–
18. Workspace: Workspace name
19. Components/Controls: Component or Controls used in this workspace
20. Data Fetch: Fetched data sets for this workspace
21. View Data Preparation: Prepared view-data for this workspace
22. Event/Operations: Event or Operations involved in this workspace
23. In-memory/Save Data: List of In memory and Save data of this workspace
24. Workout Guideline:
25. Technical workout is the analysis of all technical and business aspect of the given business.
26. Items of 1.1 to 1.6 will be generated from technical workout
27. **1.11 Time Estimation**
28. We describe estimated time to complete the sub-goal in Time Estimation table. Lets understand table columns of the table one by one–
29. Menu/Sub-menu: Menu/Sub-menu name
30. Component: UI Components of the Menu/Sub-menu
31. Task Hints: Hints for the specific UI component
32. Support Tools and Snippets: List of Support tool and snippets which will be used
33. Time: Estimated time for the task/UI component
34. **Conclusion → Mock Data List**
35. A table to describe mock data used in the expression. Lets understand table columns of the table one by one–
36. Feature: Feature name
37. Menu/Sub-menu: Menu/Sub-menu name where this mock used
38. Mock: Mock data name
39. Services: Service function name
40. Location/Factory: Service name

## Appendix L: Coding Norms and Principles

1. **1. NORMS**
2. **1.1 Null/ Undefined check**
3. Q: Why should we know the norms?
4. Ans: It will help us to give us perfection and will consume our time and effort.
5. Let’s know the norms of coding…



Is my Data ok?

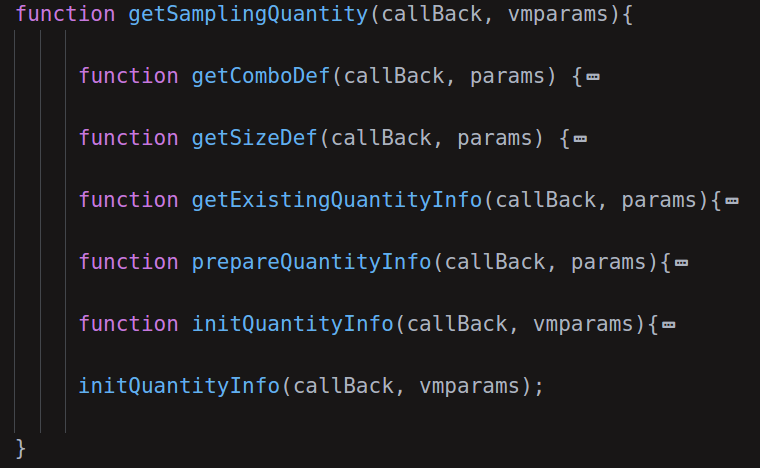
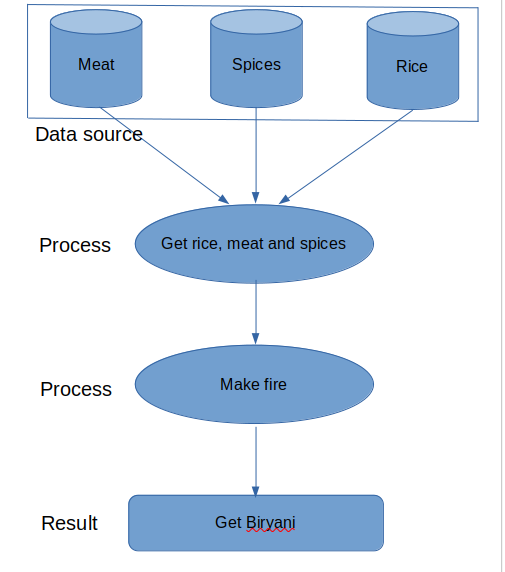
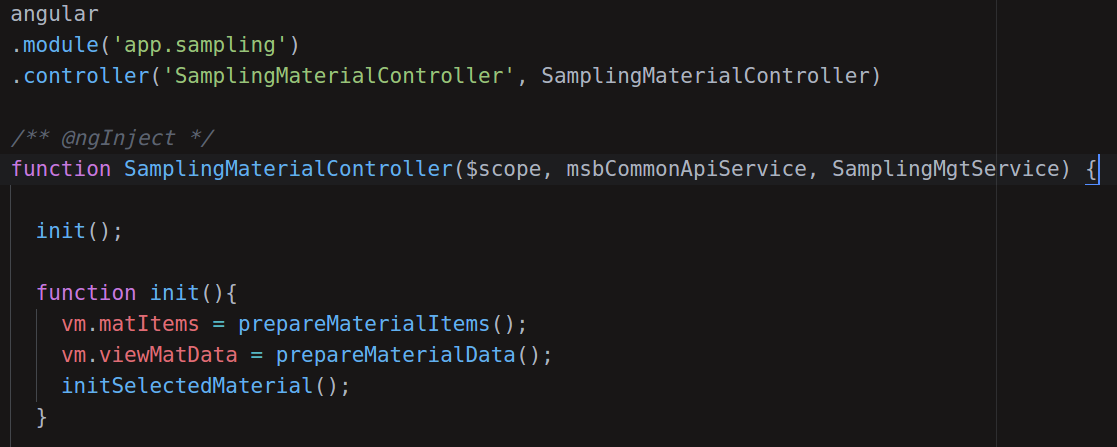
1. 
2. **1.2 Single function to perform single task**
3. 

## 

1. **1.3 Naming / structure convention**



A tree is known by its fruits.

1. 
2. **1.4 Data-driven Programming**
3. Data driven programming is a programming paradigm where the data itself controls the flow of the program and not the program logic or steps.
4. 
5. **2. Coding Approach**
6. Q: What approach we are looking for?
7. Ans: We are looking for approach to develop controller and services here.
8. Q: Is it necessary, I mean I can develop those anyhow I want to do!!!
9. Ans: Sorry son, you are now part of Never-land. You should make it clean.
10. Let’s learn it…
11. **2.1 Controller**
12. **2.1.1 init() function**
13. 
14. **2.1.2 Data Collection**
15. 



I have collected Data

1. **2.2 Service**
3. **2.2.1 Ready Data for View**

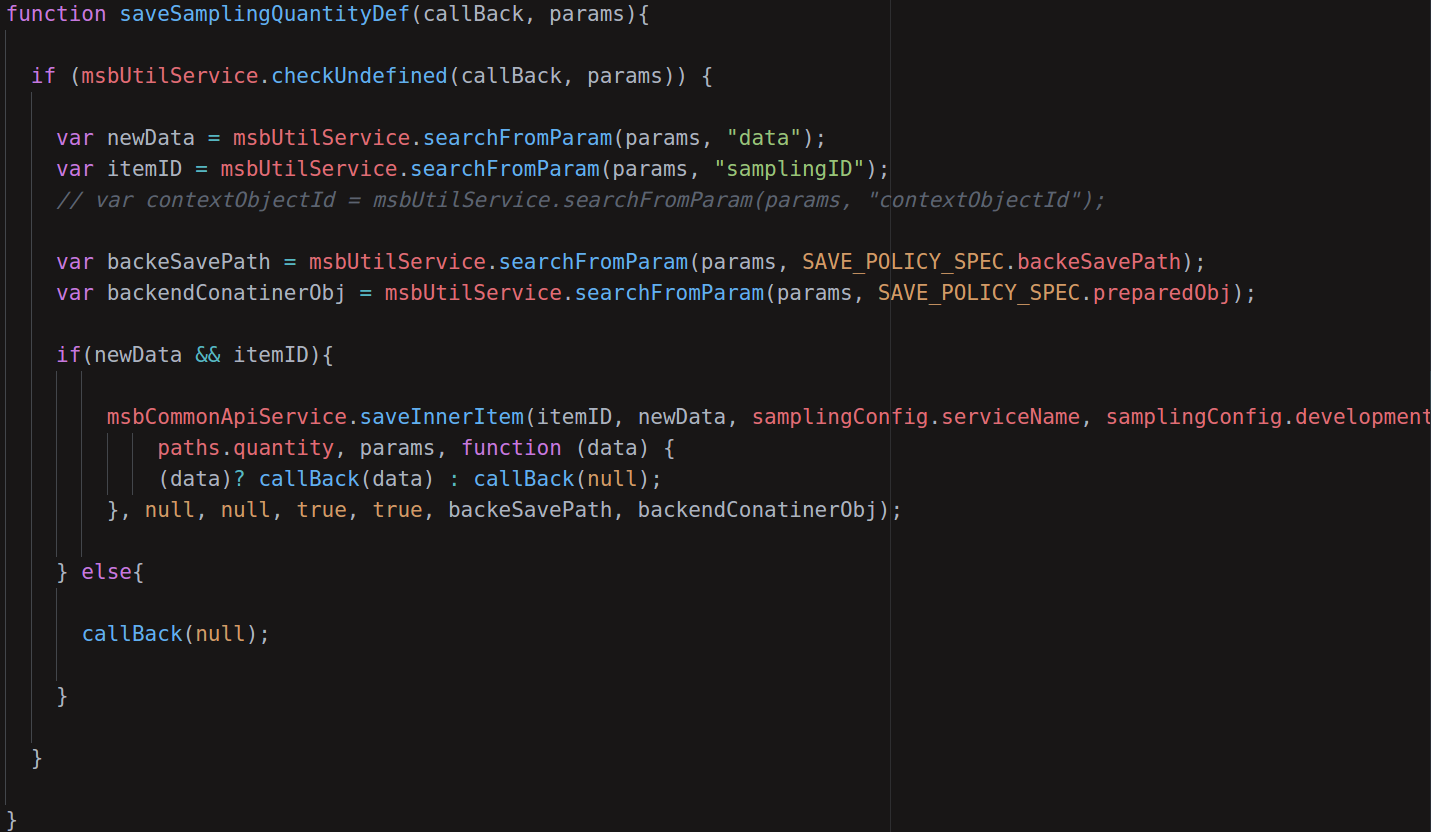


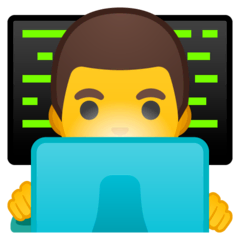
Here is your data for view

1. 
2. **2.2.2 Save Data**



Save Complete

1. 
2. **3. Coding Principles**
3. Q: What about principles?
4. Ans: Our main principle is user friendly clean coding.
5. let’s talk about our principles..
6. **3.1 No Cheap Coding**



Coding is an art,let’s make it beautiful

1. 
2. **3.2 Code for Others**
3. 

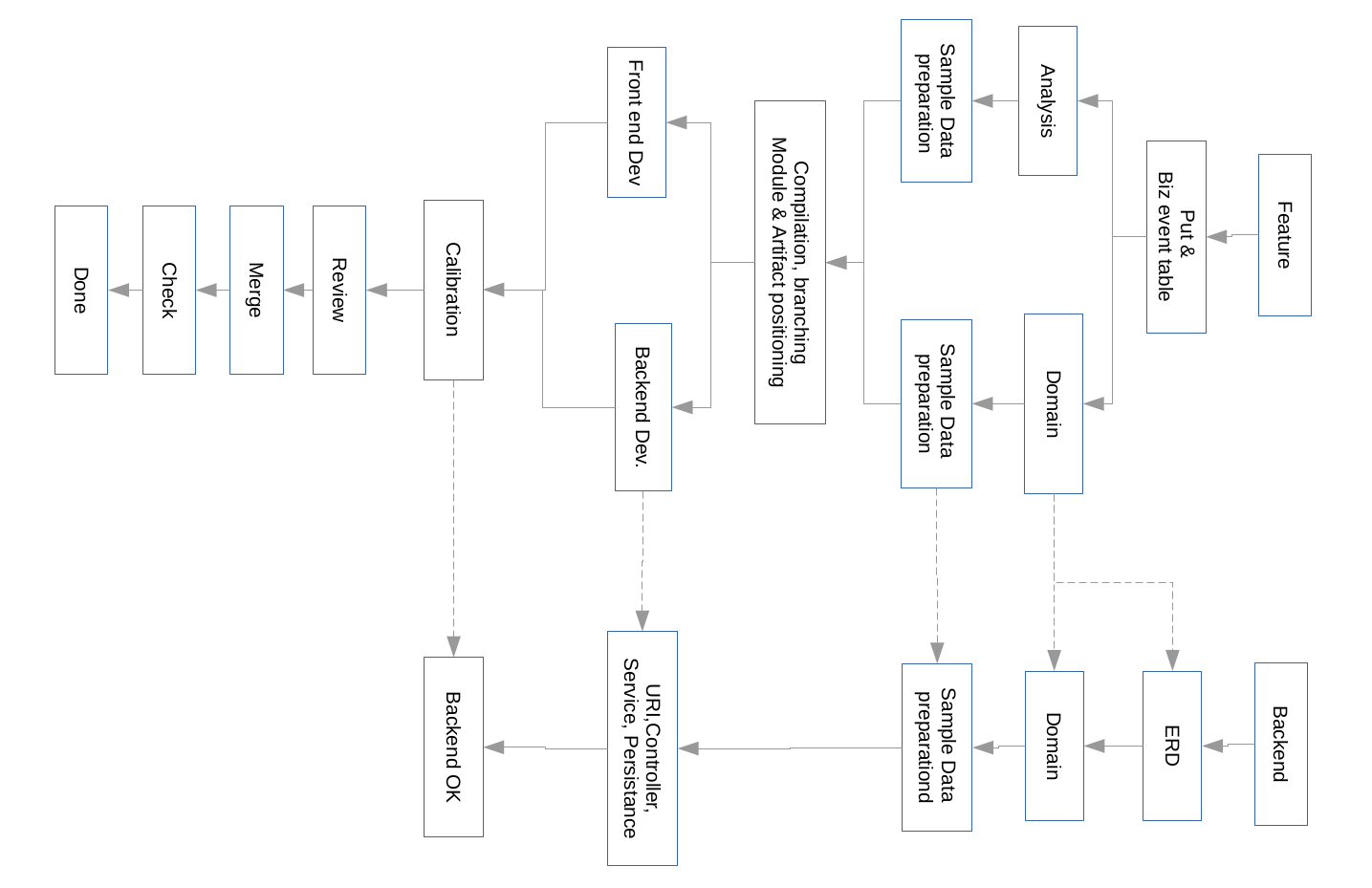


If my art is not understandable,

Then what kinda artist am I !

1. **3.3 Building Blocks**
2. Q: Building blocks!!! What’s that?
3. Ans: Let’s find out together..
4. **3.4 Project Structure**
5. 
6. **3.5 Naming Convention**
7. “There is no principle worth the name if it is not wholly good.”
8. - Mahatma Gandhi.
9. JSON attributes and structure naming convention should be followed.
10. **3.6 View Components**
11. “The best view comes after the hardest climb” - Unknown Author.
12. **3.7 Logic**
13. Find out how many difficulties a user can face and solve them as logically as you can.
14. “Logic: The art of thinking and reasoning in strict accordance with the limitations and incapacities of the human misunderstanding.” - Ambrose Bierce.
15. **3.8 Data Structure**
16. “Bad programmers worry about the code. Good programmers worry about data structures and their relationships.” - Linus Torvalds.
17. **3.9 Business**
18. “Your purpose in life is to find your purpose and give your whole heart and soul to it.” - Buddha
19. **3.10 Common Service**
20. “The best way to find yourself is to lose yourself in the service of others.” - Mahatma Gandhi.
21. **3.11 JSON Manager**
22. "Management is doing things right; leadership is doing the right things" - Peter F. Drucker

## Appendix M: Work Flow



## Appendix N: Time Allotment Guideline

### Put and Analysis Time Allotment

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1. **Level** | 1. **Put & Biz Table** | 1. **Analysis** | 1. **Domain** | 1. **Data** | 1. **Compilation** | 1. **Review** | 1. **Total** |
| 1. Beginner | 1. 2 hr | 1. 1.30 hr | 1. 2.30 hr | 1. 2.0 hr | 1. 1.3 hr | 1. 1 hr | 1. 9.5 hr |
| 1. Mid level | 1. 1.15hr | 1. 1 hr | 1. 2.00 hr | 1. 1.30 hr | 1. 1.30 hr | 1. 1 hr | 1. 7.15 hr |
| 1. Expert | 1. 1hr | 1. 45 min | 1. 2.00 hr | 1. 1.15 hr | 1. 1.30 hr | 1. 1 hr | 1. 6.30 hr |

### Component Time Allotment

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Level** | **Menu Group** | **Sub Menu** | **Info Menu** | **List** | **Form** | **Tab** |
| Beginner | 15 | 15 | 30 | 45 | 30 | 30 |
| Mid level | 10 | 10 | 25 | 30 | 20 | 20 |
| Expert | 10 | 10 | 20 | 30 | 20 | 15 |

## Appendix O:DXR Project Policies

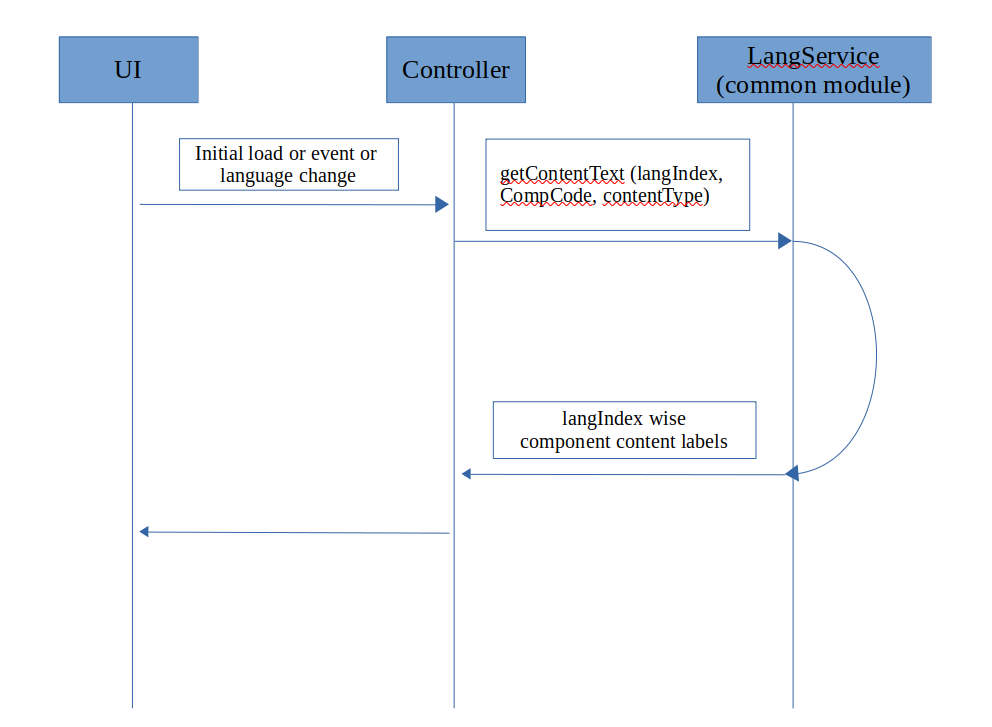
### DXR Language Competency Policy

### Language Competency in Front-End

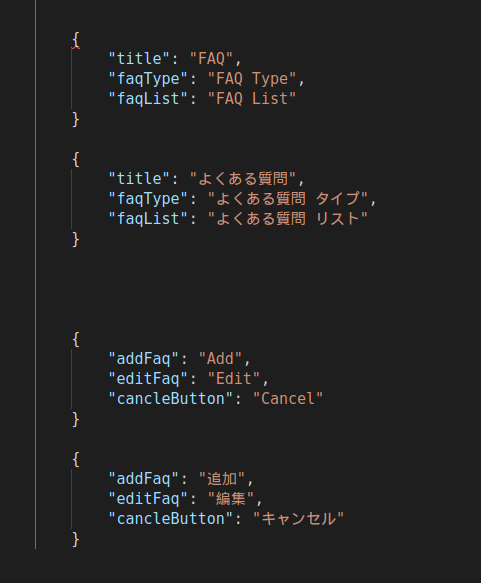
Application UI labels and texts will be stored component wise in Configuration module. Such as –



### Sequence Diagram



### Return JSON:



### Policy

1. In a variable named lang\_setup in environment files (environment.prod.ts and environment.ts)
2. Inject this variable in language service.
3. We will maintain a JSON file for taking Japanese text
4. **Local Conversion**: we will convert our JSON into Japanese then merge with our original JSON.
5. **Client/Japanese Conversion**: We will provide merged JSON to client for corrections.
6. **Development**: We will put this JSON data in our environment.prod.ts
7. **Uses**: In html pages use expression instead of hardcoded text.

### References

PostgreSQL Character Support

https://www.postgresql.org/docs/9.1/multibyte.html

Derby Character Support

https://db.apache.org/derby/docs/10.0/manuals/tools/tools38.html

### Language Competency In Database

### Policy

1. We will use UTF-8 charset in database so that both English and Japanese text can be stored.
2. ~~We will use two database. One for English language settings in application and another for Japanese language settings in application.~~
3. ~~All REST url of frontend controller have to be prefixed by ‘langPrefix’ using httpHandlerService.~~

### Backend REST URL Policy

1. In Backend Mobile app url will be prefixed by /mob
2. All url of mobile app have to be prefixed by ‘/mob’
3. In Backend Web app url will be prefixed by /web
4. All url of web app have to be prefixed by ‘/web’
5. In backend there will separate REST Controller for Mobile App and Web App

### Web Application Module Policy

|  |  |  |  |
| --- | --- | --- | --- |
| **SN.** | **Module** | **Note** |  |
|  | Visitor |  |  |
|  | System Admin |  |  |
|  | Dumpers Admin |  |  |
|  | Processor Admin |  |  |
|  | Transporter Admin |  |  |
|  | Drivers |  |  |
|  | Dumper Account |  |  |
|  | Processor Account |  |  |
|  | Transporter Account |  |  |
|  | Common Util |  |  |
|  | Common UI Items |  |  |

## Appendix P: Sample Tables

### Sample Routing Def Table

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Primary Menu | | Secondary Menu | | Workspace | | Popup | | Remarks |
| Level I | Level II | Level I | Level II | Workspace | Workspace Tab | Popup | Popup Tab |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |

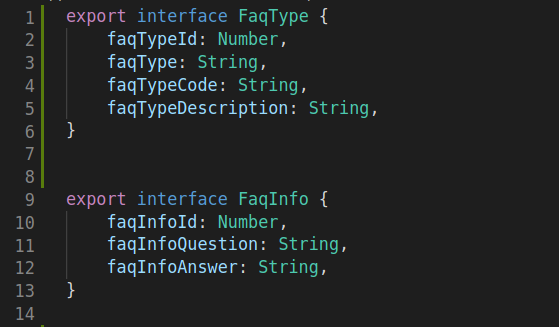
### Sample Event Table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Operation** | **Event** | **Input** | **Output** | **Time** | **Status** |
| Admin popup | FAQ Type tab |  | faqTypeList | 1 hr |  |
| FAQ List tab | First FAQ’s faqTypeId | faqList of specific faqType | 1.5 hr |  |
| FAQs tab | First FAQ’s faqTypeId | faqList of specific faqType | 1.5 hr |  |
| Add/Edit FAQ Type | Add | faq\_type | faqTypeList | 30 min |  |
| Edit | faq\_type | faqTypeList | 30 min |  |
| FAQ List | faqTypeId | faqList of specific faqType | – |  |
| Add/Edit FAQ | Add | fac\_info | faqList | 30 min |  |
| Edit | fac\_info | faqList | 30 min |  |
|  |  |  |  |  |  |

### Sample URI Table

|  |  |  |  |
| --- | --- | --- | --- |
| **Event** | **URI** | **Method** | **Parameter** |
| Onload() | 192.168.0.123:8001/web/students | GET |  |
| add() | 192.168.0.123:8001/web/students/add | POST |  |
| update() | 192.168.0.123:8001/web/students/update | POST |  |
| remove() | 192.168.0.123:8001/web/students/delete | DELETE |  |
| onLoad() | 192.168.0.123:8001/web/courses | GET |  |
| add() | 192.168.0.123:8001/web/courses/add | POST |  |
| update() | 192.168.0.123:8001/web/courses/update | POST |  |
| remove() | 192.168.0.123:8001/web/courses/delete | DELETE |  |
|  |  |  |  |

### Sample Frontend Model



### Sample Artifact Position Table

|  |  |  |  |
| --- | --- | --- | --- |
| **Application** | **Module** | **Items** | **Path** |
| Web Application | Visitor | FaqTypeList | visitor/FaqType |
| FaqQuestionAnswer | visitor/FaqQuestionAnswer |
| System Admin | FaqTypeForm | system-admin/FaqTypeForm |
| FaqQuestionAnswerForm | system-admin/FaqQuestionAnswerForm |
| Dumpers Admin |  |  |
| Processor Admin |  |  |
| Transporter Admin |  |  |
| Drivers |  |  |
| Dumper Account |  |  |
| Processor Account |  |  |
| Transporter Account |  |  |
| Common UI Items |  |  |
| Common Util | Common Util |  |  |
|  |  |  |  |