ToDo: pretty good “to do” application

*This specification was generated from a specification-only editor. Item identifiers, data sources and action statements were edited and error checked interactively while working within the UI designer and editor.*

*All italicised text is commentary. Black text is part of the original specification. Red text has been added here for explanation of this specification sample. This sample demonstrates all of the specification sections including the optional Data Storage and Lexicon sections.*

*This is a small but quite powerful ToDo application. It enables users access from any device from any location while always viewing their information in their home culture (dates, times etc.). They can sign in for a whole day without having to sign in again from the same device. It color codes tasks by age and can place tasks within categories and enable sub-tasks under a parent task.*

*It demonstrates specification language that maintains a balance between readability and specificity. The form (or page) layouts have been created using Balsamiq Mockups.*

## Host: Cloud

### Start.Process

*Starts the application. Looks for a cookie; if cookie is alive the user doesn’t need to sign in again. Project leader/facilitator would walk client though this.*

Culture = Australia *Always show dates, times etc in Australia format*

TodayDATE = Today + 1 day

TomorrowDATE = Today + 2 days

Next10DATE = Today + 11 days

Next30DATE = Today + 31 days

Selection = 'Tasks for next 10 days'

Load ToDoControl for this computer *Enables a user to access from different devices*

If ToDoControl.Code = LoginRef *LoginRef comes from cookie on this device*

 MyUserId = ToDoControl.Usr

Load ToDoUser *Setup user if cookie is still active*

 Load tasks for next 10 days

>Main.Form

>SignIn.Form *Sign in if there is no active cookie*

**End of Start.Process**

### SignIn.Form



UsrId: myUserId *the input UsrId in form above is stored as myUserId*

[Sign In] Error if UsrId = Nothing or Password = Nothing

-Heading = UsrId

-Message = 'You must enter a User Id and a Password'

Load ToDoUser

Error if Password <> ToDoUser.Pass

-Heading = myUserId

-Message = 'Check password and try again

Login for 1 day

ToDoControl.Code = LoginRef *LoginRef is provided by Login*

ToDoControl.Auto = Keep\_me\_signed\_in

Load tasks for next 10 days

>Main.Form

[Register] Error if UsrId = Nothing or Password = Nothing

 -Heading = UsrId

 -Message = 'You must enter a User Id and a Password'

Error if myUserId is in ToDoUser

-Heading = 'ToDo registration'

-Message = 'User Id already registered. Try again.'

Create ToDoUser

-Id = autoID

-UsrId = myUserId

-Pass = Password

 Login for 1 day

Create ToDoControl *this enables a user to use multiple devices*

-Code = LoginRef

-Usr = ToDoUser.Id

-Auto = Keep\_me\_signed\_in

>Main.Form

**End of SignIn.Form**

### Main.Form



*Items ending in : below (eg. TaskName: ) refer to identifiers on form above. All dynamic data items require a source (eg. ToDo.Name). Stakeholders are introduced to this by the project leader and quickly understand it.*

TaskName: ToDo.Name *TaskName is bound to Todo.Name in database*

TaskDate: ToDo.TaskTime(dd/MM/yyyy) *eg. 31/07/2021, pick from dropdown calendar*

TaskDetails: ToDo.Details

Category: ToDo.Category *shows category for this ToDo task*

 DropList ToDoCategory.Name *list all categories to enable change of category*

Parent: ToDo.Parent

 DropList ToDoParent.Name

This is a parent task: ToDo.IsParent

ChooseCategory: DropList ToDoCategory.Name *list category names*

 Load tasks for this category *on selection of category*

ChooseParent: DropList ToDoParent.Name

 Load tasks for this parent

TaskDetailsx: ToDo.Show

Tasksx: ToDo.Name

ForeColor = 'black'

BackColor = 'white' *codes can be selected interactively from color selector*

BackColor = '#ffe599' if ToDo.TaskTime < Next30DATE

BackColor = '#F1c232' if ToDo.TaskTime < Next10DATE

BackColor = '#e69138' if ToDo.TaskTime < TomorrowDATE

BackColor = '#cc0000' if ToDo.TaskTime < TodayDATE

ForeColor = 'white' if ToDo.TaskTime < TodayDATE

Image1: ToDo\_Header.jpg'

SelectedTasks: Selection{ReadOnly}

[new task] Create ToDo

-Id = autoID

-Create d = Now

-TaskTime = Now

-Icon = 'ToDo\_Pencil.jpg'

-Category = 100 *code for Select a Category*

-Parent = 100

[settings] Display -Message = 'Settings not implemented yet!'

[Today] Backcolor = '#cc0000' *colour code selected from colour selector*

 ForeColor = 'white'

 Load tasks for today

 Selection = 'Tasks to be completed by today'

[Tomorrow] BackColor = '#e69138'

Load tasks for tomorrow

Selection = 'Tasks to be completed by tomorrow'

[Next 10 days] BackColor = '#f1c232'

Load tasks for next 10 days

Selection = 'Tasks for next 10 days'

[Next 30 days] BackColor = '#ffe599'

Load tasks for next 30 days

Selection = 'Tasks for next 30 days'

[Beyond 30 days] Load tasks beyond 30 days

 Selection = 'Tasks beyond 30 days'

[All Tasks] Load all tasks

 Selection = 'All Tasks' *Selection shows as SelectedTasks on form*

[DONE] ToDo.Done = True *mark ToDo item as done*

[DELETE] Delete Todo

[SAVE CHANGES] Create ToDoParent if ToDo.IsParent = True and\_

ToDo is not in ToDoParent

 -Name = ToDo.Name

 -ToDo\_Id = ToDo.Id

Delete ToDoParent for this ToDo if ToDo.IsParent = False

ToDo.Show = ToDo.TaskTime & LF & ToDo.Details

[Selectsx3x6\_ToDo] ForeColor = 'white' *ToDo grid is 3 wide and 6 deep*

Backcolor = '#666666'

[Next Page] >NextPage

[Previous Page] >PreviousPage

**End of Main.Form**

### Main\_Phone.Form



ChooseCategory: DropList ToDoCategory.Name

 Load tasks for this category *on selection of category*

ChooseParent: DropList ToDoParent.Name

Load tasks for this parent

TaskDetailsx: ToDo.Show

Tasksx: ToDo.Name

ForeColor = 'black'

BackColor = 'white'

BackColor = '#ffe599' if ToDo.TaskTime < Next30DATE

BackColor = '#F1c232' if ToDo.TaskTime < Next10DATE

BackColor = '#e69138' if ToDo.TaskTime < TomorrowDATE

BackColor = '#cc0000' if ToDo.TaskTime < TodayDATE

ForeColor = 'white' if ToDo.TaskTime < TodayDATE

Selected: DropList ToDoSelect.Name

 Load tasks for today if ToDoSelect.Name = 'Today'

 Load tasks for tomorrow if ToDoSelect.Name = 'Tomorrow'

 Load tasks for next 10 days if ToDoSelect.Name = '10 Days'

 Load tasks for next 30 days if ToDoSelect.Name = '30 Days'

 Load all tasks if ToDoSelect.Name = 'All Tasks'

[Selectsx1x20\_ToDo] ForeColor = 'white' *one ToDo task across and 20 deep*

 Backcolor = '#666666'

 >Edit.Form *when Select button is clicked*

[settings] Display -Message = 'Settings not implemented yet!'

[new task] Create ToDo

-Id = autoID

-Created = Now

-TaskTime = Now

-Icon = 'ToDo\_Pencil.jpg'

-Category = 100

-Parent = 100

 >Edit.Form

[Next Page] >NextPage

[Previous Page] >PreviousPage

**End of Main\_Phone.Form**

### Edit.Form



TaskName: ToDo.Name

TaskDate: ToDo.TaskTime(dd/MM/yyyy)

TaskDetails: ToDo.Details

Category: ToDo.Category

 DropList ToDoCategory.Name

Parent: ToDo.Parent

DropList ToDoParent.Name

This is a parent task:ToDo.IsParent

[DONE] ToDo.Done = True

[DELETE] Delete Todo

[SAVE CHANGES] Create ToDoParent if ToDo.IsParent = True and ToDo is not in ToDoParent

 -Id = autoID

 -Name = ToDo.Name

 -ToDo\_Id = ToDo.Id

 Delete ToDoParent for this ToDo if ToDo.IsParent = False

 ToDo.Show = ToDo.TaskTime & LF & ToDo.Details

 >Main\_Phone.Form

[BACK] >Main\_Phone.Form

**End of Edit.Form**

### Lexicon:

*The lexicon is optional and managed by the project leader/facilitator. It enables easy-to-read phrases (left element) to be used in the specification. The right element contains the information that is used in the running application. The client would not generally be expected to review or approve this section.*

ToDo is not in ToDoParent : ToDo.Id = ToDoParent.ToDo\_Id

ToDoParent for this ToDo : ToDoParent.ToDo\_Id = ToDo.Id

UsrId is in ToDoUser : UserId = ToDoUser.UsrId

Load tasks for today : SELECT \* FROM ToDo WHERE TaskTime < [TodayDATE] ORDER BY TaskTime

Load tasks for tomorrow : SELECT \* FROM ToDo WHERE TaskTime < [TomorrowDATE]\_

ORDER BY TaskTime

Load tasks for next 10 days : SELECT \* FROM ToDo WHERE TaskTime < [Next10DATE]\_

ORDER BY TaskTime

Load tasks for next 30 days : SELECT \* FROM ToDo WHERE TaskTime < [Next30DATE\_

ORDER BY TaskTime

Load tasks beyond 30 days : SELECT \* FROM ToDo WHERE TaskTime >= [Next30DATE]\_

ORDER BY TaskTime

Load all tasks : SELECT \* FROM ToDo ORDER BY TaskTime

Load tasks for this category : SELECT \* FROM ToDo WHERE CategoryId = [ToDoCategory.Id]

Load tasks for this parent : SELECT \* FROM ToDo WHERE ParentId = [ToDoParent.Id]

Load ToDoControl for this computer : SELECT \* FROM ToDoControl WHERE Code = [LoginRef]

Load ToDoControl for UsrId : SELECT \* FROM ToDoControl WHERE Usr = [ToDoUser.Id]

**End of Lexicon**

### Data Storage and Access:

*This section is optional but commonly used. If you are using existing database tables you don’t need this section although it’s common to nominate the tables being used.. If you are working from existing data sources, you can specify a friendlier name and associate it with a more obscure database name (eg. Customer Table: database name=LondonCust004).*

*In this example, these are new database tables. The required tables and associated columns are a byproduct of the specification-only design process and can be automatically created directly or via generated scripts. This section would be monitored by the project leader/facilitator; the client would not generally be expected to review or approve this section.*

**DBMS: Cloud**

ToDo Table

-IsParent Boolean

-Done Boolean

-Category Id

-Parent Id

ToDoCategory Table

Select Command: SELECT \* FROM ToDoCategory

-Id Id

-Name 100

ToDoParent Table

Select Command: SELECT \* FROM ToDoParent

-Id Id

-Name 100

-ToDo\_Id Id

ToDoUser Table

Select Command: SELECT \* FROM ToDoUser WHERE UsrId = [MyUserId]

-Id Id

-UsrId 30

-Pass 20

ToDoControl Table

-Id Id

-Code 100

-Usr Id

-Auto Boolean

ToDoSelect Table

Select Command: SELECT \* FROM ToDoSelect

-Id Id

-Name 20

**End of Host**

**End of Specification**