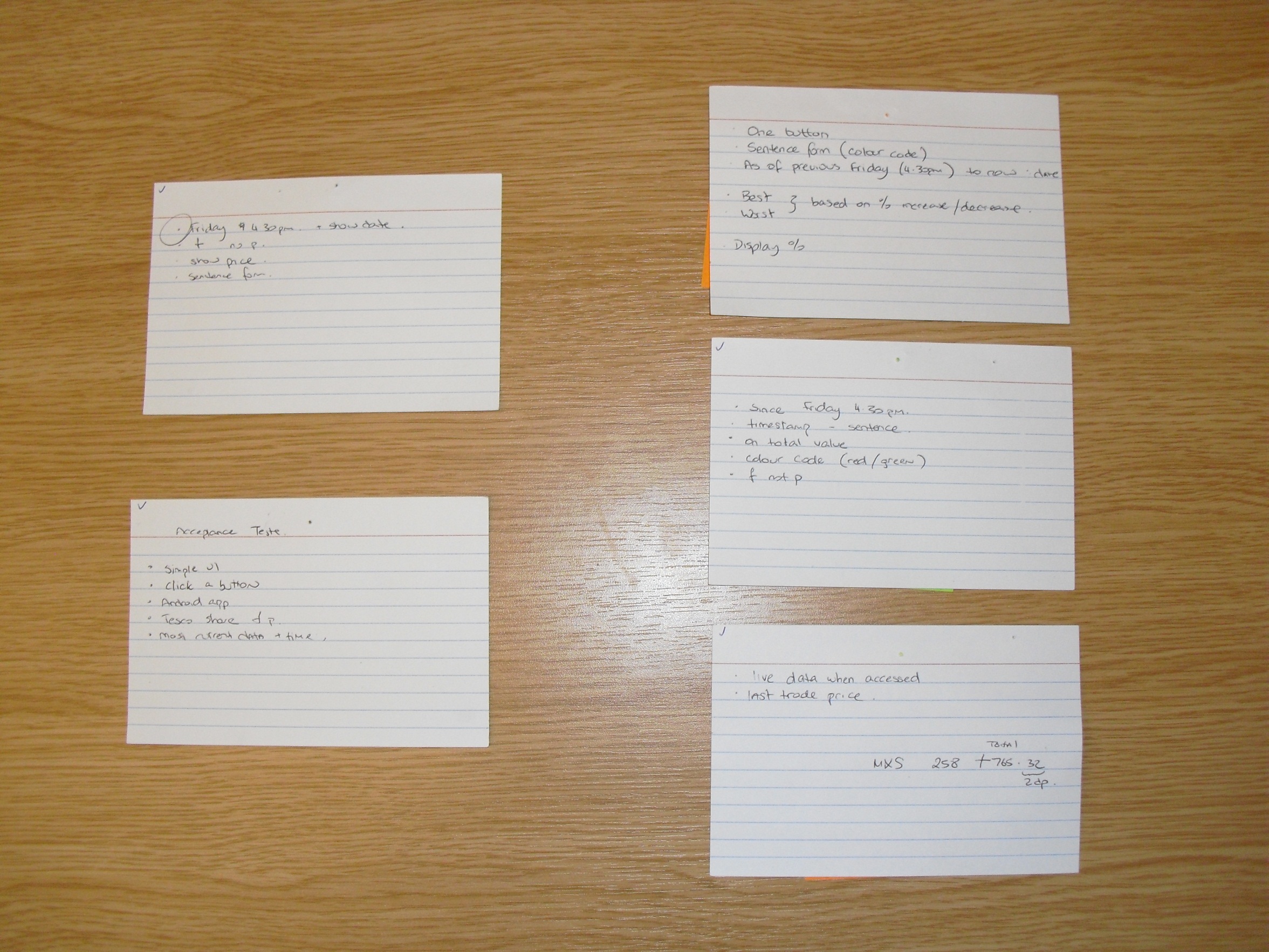
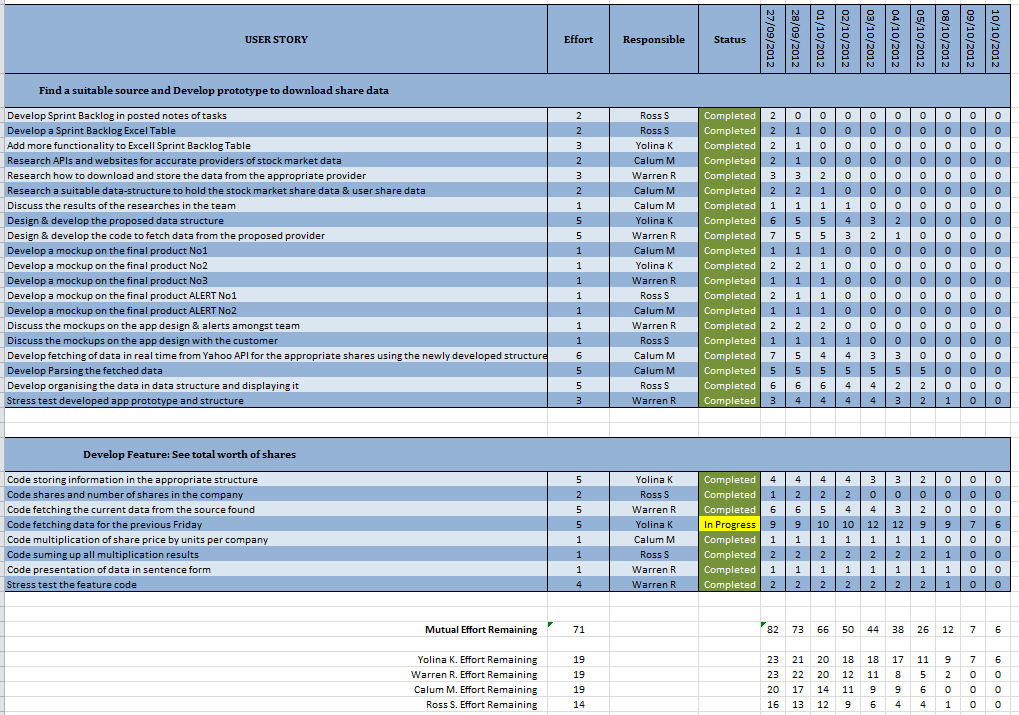
**TEAM 1: PRODUCT BACKLOG**

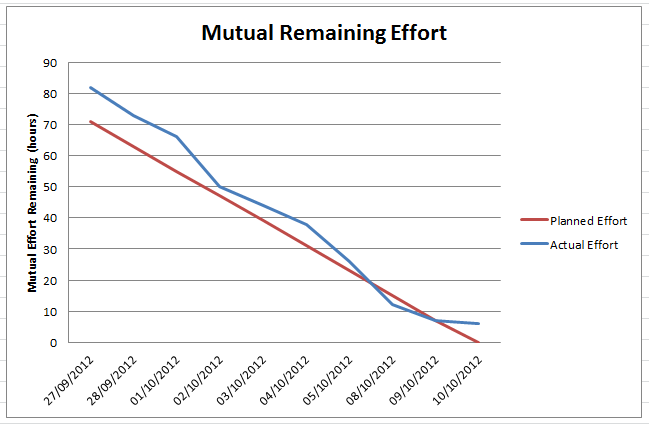
****

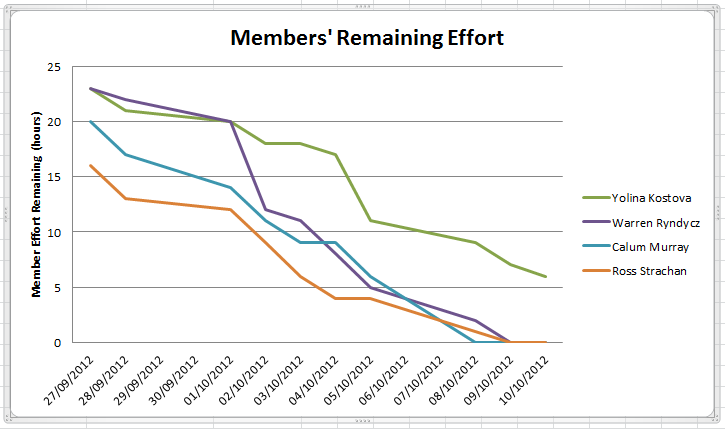
**SPRINT 1 ACCEPTANCE TESTS SPRINT 2 ACCEPTANCE TESTS**

****

**TEAM 1: SPRINT BACKLOG**







**TEAM 1: TASK TACKLING & DEFINITION**

|  |
| --- |
| **DEVELOPER USER STORY** |
| 1. **Develop Sprint Backlog in posted notes of tasks**   Having the tasks written out on posted notes with relative estimates of effort |
| 1. **Develop a Sprint Backlog Excel Table**   Sprint backlog Excel Table would include:  Breakdown of tasks  Effort estimate per task  Hours remaining for the sprint  Team member allocation per task  Hours remaining per team member  Status indicator for the task  Be updated daily by each team member |
| 1. **Add more functionality to Excell Sprint Backlog Table**   Script done for the remaining times of each member to be calculated  Burn-down graph generated |
| 1. **Research APIs and websites for accurate providers of stock market data**   Results found on providers with information for the reliability of the API |
| 1. **Research how to download and store the data from the appropriate provider**   Results found on ways of implementing a reliable connection to the API – making sure that those work properly, thus giving accurate information that can be stored |
| 1. **Research a suitable data-structure to hold the stock market share data & user share data**   The result of research on structures which would suit the purposes of the program best, thus store any piece of needed information per share |
| 1. **Discuss the results of the researches in the team**   Team meeting held and decisions made on API, way of fetching information reliably from the API & data structure to hold the information |
| 1. **Design & develop the proposed data structure**   Data structure produced which is reliable and flexible enough to hold the information to be stored per each share set |
| 1. **Design & develop the code to fetch data from the proposed provider**   Boundary class with appropriate methods produced to accommodate reliable fetching from the API – giving the most accurate information |
| 1. **Develop a mockup on the final product No1**   Picture on appropriate design to be suggested for the finished product |
| 1. **Develop a mockup on the final product No2**   Picture on appropriate design to be suggested for the finished product |
| 1. **Develop a mockup on the final product No3**   Picture on appropriate design to be suggested for the finished product |
| 1. **Develop a mockup on the final product ALERT No1**   Picture on appropriate design to be suggested for the alert of the product |
| 1. **Develop a mockup on the final product ALERT No2**   Picture on appropriate design to be suggested for the alert of the product |
| 1. **Discuss the mockups on the app design & alerts amongst team**   Team meeting held and decisions made on which designs are to be suggested to the customer |
| 1. **Discuss the mockups on the app design with the customer**   Meeting with the customer to accomplish the mockups demo |
| 1. **Develop fetching of data in real time from Yahoo API for the appropriate shares using the newly developed structure**   Boundary class methods organized to accommodate reliable fetching in real time from the API – giving the most accurate information for each of the customer’s shares |
| 1. **Develop Parsing the fetched data**   Method developed for accurate parsing of the fetched information so that it could be stored in the appropriate function |
| 1. **Develop organising the data in data structure and displaying it**   Methods developed:  to store the parsed data in the developed structure  to allow display of the data from the structure  to allow fetching and using the information in the structure |
| 1. **Stress test developed app prototype and structure**   Testing the response for the case of having no internet connection available or the application having slow loading times |
|  |

|  |
| --- |
| **TOTAL PORTFOLIO USER STORY** |
|  |
| 1. **Code storing information in the appropriate structure**   Method extended to store even more than the information required for the prototype |
| 1. **Code shares and number of shares in the company**   Class with methods to set up the whole portfolio of the customer |
| 1. **Code fetching the current data from the source found**   Method extended to fetch even more than the information required for the prototype |
| 1. **Code fetching data for the previous Friday**   Method developed to store the information from the previous stock week’s close value |
| 1. **Code multiplication of share price by units per company**   Method developed to multiply the number of shares by the price fetched and stored in the structure provided |
| 1. **Code suming up all multiplication results**   Method developed to sum up all results stored in the structure provided |
| 1. **Code presentation of data in sentence form**   Method extended to accommodate sentence form for the sum of total share value |
| 1. **Stress test the feature code**   Testing for appropriate responsiveness if no internet connection is available or slow loading times, accurate information gathered and connection to the API |

**TEAM 1: VELOCITY**

**We as a team decided that for the first sprint we would aim to finish the most valuable user stories which implied that the first ones to be suggested would be the ones with the highest result of the division of business value by the complexity (team effort required). The results suggested that there were two stories with the highest value which added up to team effort of 9. We as a team discussed the matter and found that the challenge would be motivating for us, thus we chose the two user stories.**

**As discovered during the sprint retrospective the velocity of our team is 7 rather than 9.**

**We have now managed to decrease the number of issues that prevented us of finishing with the promised user stories. Having discussed the further stories with the customer, we have decided to aim for velocity 9 as motivation and because we believe that having learnt from our previous mistakes we would be able to achieve complete success.**

**TEAM 1: THOUGHTS ON ACCEPTANCE TESTS**

(User Stories in order to be done throughout the sprints)

1. **DEVELOPER STORY**
   1. Pre-agreement ideas to show results from research?
   2. Agreed to produce a prototype with the following features:
      1. Android app
      2. Simple UI
      3. Button to click
      4. Tesco share price fetching in pennies
      5. Most Current data presented with timestamp
2. **TOTAL PORTFOLIO USER STORY**
   1. Pre-agreement ideas have been approved by the customer to be:
      1. Android app
      2. Fetching price for Friday @ 4.30PM
      3. Showing total worth of portfolio at that time
      4. Rounding the total to pounds with no pennies
      5. Showing time and date
      6. Sentence form presentation of the information
3. **LOSS/GAIN FOR A WEEK USER STORY**
   1. Pre-agreement ideas have been approved by the customer to be:
      1. Data comparison between current and Friday @4.30PM
      2. Timestamp to show time at which the data is the most accurate
      3. Showing total worth of the portfolio?
      4. Showing the lost or gained value with red/green color coding
      5. Presenting the data in sentence form
      6. Values rounded to pounds with no pennies
4. **TOTAL FOR SETS OF SHARES USER STORY**
   1. Would it be refreshing constantly or just on button click?
   2. Pre-agreement ideas have been approved by the customer to be:
      1. Live data fetched when accessed
      2. Refreshed on load from menu (not constantly)
      3. Using last trade price of the shares
      4. Presented in table of the form

* Column 1: Name of company
* Column 2: Number of shares
* Column 3: Current total price of shares in pounds rounded up to two decimal places

1. **BEST / WORST PERFORMING USER STORY**
   1. Pre-agreement ideas have been approved by the customer to be:
      1. Best & worst presented on one menu button click
      2. Comparison as of previous Friday @4.30PM to now ( show date of the Friday)
      3. Timestamp to show the time at which the data is most accurate
      4. Sentence form presentation with red / green color coding
      5. Best/worst based on percentage increase/decrease
      6. Display the percentage increase/decrease for more clarity
2. **PLUMMET / ROCKET OF SHARE PRICES** 
   1. Pre-agreement ideas:
      1. Show the most current data on price
      2. Red/green color coding
      3. Alert issued on start-up of the app
      4. Sentence form including:

* company name
* current price
* percentage rise/fall
  + 1. Data comparison as of the previous evening close

1. **ALERT ON RUN OF SHARES USER STORY**
   1. Pre-agreement ideas:
      1. Show the most current data on volume of shares in company
      2. Red/green color coding
      3. Alert issued on start-up of the app
      4. Sentence form including:

* company name
* current volume
* percentage run
  + 1. Data comparison as of the previous evening close

1. **WEEKLY HIGH/LOW FOR EACH SHARE USER STORY** 
   1. Pre-agreement ideas:
      1. Data taken since the previous Friday’s close up till now
      2. Table view of the data including:

* Column 1: Name of company
* Column 2: Weekly High
* Column 3: Weekly Low
* Column 4: Current price of a share unit
  + 1. Red / green color coding
    2. Prices rounded to second decimal place in pounds

1. **RISE & FALL OF SHARE PRICES DAILY USER STORY**
   1. Pre-agreement ideas:
      1. Data taken since the previous day’s close up till now
      2. Table view of the data including:

* Column 1: Name of company
* Column 2: Number of Shares
* Column 3: Current price of a share unit
  + 1. Red / green color coding
    2. Prices rounded to second decimal place in pounds

**TEAM 1: CUSTOMER COMMUNICATION**

**TEAM 1: TEAM COMMUNICATION & PROBLEM SOLVING**